Contents

Preface ............................................................................................................................................................ xxvii
  Audience ........................................................................................................................................................ xxvii
  Downloading Oracle Communications Documentation ................................................................. xxvii
  Documentation Accessibility ................................................................................................................ xxvii
  Document Revision History ................................................................................................................ xxvii

Part I  Basic BRM System Administration

1  Starting and Stopping the BRM System
  About Starting and Stopping BRM Components ............................................................................... 1-1
    Choosing the User Name for BRM Components ............................................................................. 1-1
    Starting Several BRM Components in Sequence ......................................................................... 1-1
    Stopping Several BRM Components in Sequence ....................................................................... 1-3
  Starting a Component by Using the pin_ctl Utility ......................................................................... 1-3
    Starting BRM Components with pin_ctl ......................................................................................... 1-4
      Starting Optional BRM Components with pin_ctl ................................................................. 1-4
      Starting pin_radiusd with pin_ctl ............................................................................................. 1-4
    Starting Pipeline Manager Components with pin_ctl ............................................................ 1-4
    Getting Diagnostic Data When You Start a Component ............................................................ 1-5
    Halting and Restarting a Component with One Command .......................................................... 1-5
    Starting a Component and Clearing the Log File ........................................................................ 1-5
    Starting a Base Set of BRM Components ....................................................................................... 1-6
  Starting BRM Components Automatically ....................................................................................... 1-6
  Starting Multiple Families of BRM Components on the Same Computer ...................................... 1-7
  Confirming That a BRM Component Started Successfully ............................................................ 1-7
  Stopping a BRM Component by Using the pin_ctl Utility ............................................................. 1-7
    Getting Diagnostic Data When You Stop a Component ............................................................... 1-8
  Stopping a Process by Using Commands ......................................................................................... 1-8
  Stopping a Base Set of BRM Components ....................................................................................... 1-8
  Starting and Stopping Pipeline Manager Manually ........................................................................ 1-9
    Starting Pipeline Manager ............................................................................................................. 1-9
    Stopping Pipeline Manager .......................................................................................................... 1-10
    Starting and Stopping Individual Pipelines .................................................................................. 1-10
    Restarting Pipeline Manager After an Abnormal Shutdown .......................................................... 1-11
    Using Perl Scripts to Start and Stop Pipeline Manager .............................................................. 1-11
Using Logs to Monitor Components ................................................................. 2-30
  Types of Log Files .................................................................................. 2-30
    Log Files for System Components .................................................. 2-30
    Log Files for Applications ............................................................... 2-30
    Log Files for Client Applications ................................................... 2-31
  Location of Log Files .......................................................................... 2-31
    Default Log File Locations ............................................................. 2-31
    Changing the Name or Location of a Log File .................................. 2-32
  Setting the Reporting Level for Logging Messages .................................. 2-32
  Getting Debugging Information from Command-Line Utilities .............. 2-33
  Dynamically Changing the CM and DM Log Levels ................................. 2-33
  Setting the Log Level for a Specific Opcode .......................................... 2-33
  Recording Opcode Calls in the CM Log File ........................................ 2-34
  About Formatting Log Files ................................................................. 2-34
  Maintaining Log Files .......................................................................... 2-35

Checking the Number and ID of a BRM Process ........................................... 2-35

Dealing with Hung and Looping Processes .................................................... 2-35
  Checking for Hung Processes ............................................................... 2-35
  Checking for Looping Processes ........................................................... 2-35
  Stopping a Hung or Looping Process .................................................... 2-36

Monitoring CM Activity .............................................................................. 2-36
  Manually Checking the Status of the CM ............................................ 2-36
  Resolving Lost TCP Connections ....................................................... 2-37
  Enabling Java PCM Clients to Use Operating System TCP/IP Keepalive Parameters .................................................. 2-37
  Setting the CM Log Time Resolution .................................................. 2-37
  Setting a Timeout Value for Requests Sent to the CM .......................... 2-38
  Configuring Multilevel CM Timeout for Client Requests ....................... 2-39
    A Short (Suspect) Timeout ............................................................... 2-39
    A Long (Failover) Timeout ............................................................. 2-39
  Getting Quality of Service Statistics from the CM ............................... 2-39
    Configuring CM QoS Statistics ....................................................... 2-40

Monitoring DM Activity .............................................................................. 2-41
  Manually Checking the Status of the DM ............................................ 2-41
  Checking the DM Status in flist Format .............................................. 2-41
    Checking the DM Status in a Report Format ................................... 2-43
  Monitoring DM Shared Memory Usage .............................................. 2-44
  Monitoring DM Transaction Queues ............................................... 2-44
  Monitoring DM Back Ends ................................................................. 2-44
  Monitoring DM Front Ends ............................................................... 2-45
  Increasing the Level of Reporting for a DM ....................................... 2-45
    Using Environment Variables to Set Debug Options ....................... 2-46
    Editing the Configuration File to Set Debug Options ....................... 2-46
    Logging the DM Process Time Information for Performance Diagnostics .................................................. 2-47
    Replacing Failed DM Child Processes ............................................ 2-47

Monitoring Pipeline Manager .................................................................... 2-47
  Monitoring Pipeline Manager Memory Usage .................................... 2-48
  Monitoring Pipeline Manager EDR Throughput .................................... 2-48
4 Implementing System Security

Using General System-Security Measures ................................................................. 4-1
Configuring Access to the BRM Database ................................................................. 4-1
Configuring Login Names and Passwords for BRM Access ........................................ 4-1
  Configuring the CM to Verify Application Logins with the Service Only ................. 4-2
  Configuring the Maximum Number of Invalid Login Attempts ............................. 4-3
  Configuring Applications to Provide Login Information ........................................... 4-3
    Login Information for Payment Tool ................................................................. 4-4
    Login Information for Optional Service Integration Components ..................... 4-4
    Login Information for Java-Based Client Applications ...................................... 4-4
    Changing Default Passwords for Java-Based Client Applications .................... 4-4
  Creating a Customer Service Representative Account ............................................ 4-5
  Changing the Login Name and Password for an Account ....................................... 4-5

Logging Customer Service Representative Activities .............................................. 4-5

Setting Up Permissions in BRM Applications .......................................................... 4-6
  About Permissioning Center .................................................................................... 4-7
    About Access to Permissioning Center ............................................................... 4-7
  About Managing Roles ............................................................................................ 4-8
    About Multiple Role Support ................................................................................ 4-8
  About Hierarchical Roles ....................................................................................... 4-8

About Managing Permissions in Permissioning Center .......................................... 4-9

Managing CSR Passwords ....................................................................................... 4-9
  Setting CSR Account Password Status .................................................................. 4-9
  Automatic Logout ..................................................................................................... 4-10
  Changing the Password ......................................................................................... 4-10
  Unlocking a Locked CSR Account ......................................................................... 4-11
  Setting the Default Password Expiry Duration ...................................................... 4-12

How Permission Settings Are Stored ....................................................................... 4-12
5 Using HP OpenView to Monitor and Control BRM

- About Using HP OpenView to Monitor BRM ................................................................. 5-1
- Hardware and Software Requirements ...................................................................... 5-1
- Configuring HP OpenView When You Use Multiple Install Directory Names .......... 5-2

5-2

- About the HP OpenView Architecture ....................................................................... 5-2
- Managing BRM Components in HP OpenView ......................................................... 5-4
- Monitoring Error Messages in Log Files by Using HP OpenView ......................... 5-4
  - Error Messages Monitored by HP OpenView ....................................................... 5-5
- Using HP OpenView to Display System Status and Topology .............................. 5-6
- Using HP OpenView to Display Performance History Data .................................. 5-7
- Using Alarms ............................................................................................................. 5-7

5-8

- Installing and Configuring the HP OpenView BRM SPI........................................ 5-8
  - Installing the BRM SPI ......................................................................................... 5-9
    - portal_spi_hpopenview Directory Contents ...................................................... 5-9
  - Uninstalling the BRM SPI .................................................................................. 5-10
  - Configuring the BRM SPI by Editing the pin_comp.values File ....................... 5-10
    - How the pin_comp.values File Is Organized ................................................... 5-10
    - Editing the pin_comp.values File .................................................................. 5-10
  - Configuring the BRM SPI .................................................................................. 5-15
  - Deploying Performance Graphs ....................................................................... 5-18
  - Collecting Performance Data ........................................................................... 5-18
  - Restarting Data Collection ............................................................................... 5-19

5-20

- Editing and Loading the Service Configuration File ................................................. 5-20
  - About the Sample Service Configuration File.................................................. 5-20

6 BRM OMF Instrumented Objects

- About the BRM MIB ................................................................................................. 6-1
- About the Top-Level Components in the MIB ....................................................... 6-1

6-1

- Transaction Manager Probes .................................................................................. 6-1
- Log File Probes .................................................................................................. 6-2

6-2

- Operations Management Framework Probes ....................................................... 6-2
  - HTTP Probes .................................................................................................. 6-2
  - SNMP Probes ................................................................................................ 6-3
  - Diagnostic Data Handler Probe ....................................................................... 6-3

6-4

- Multi-Threaded Framework .................................................................................. 6-4
  - Connection Configurations Probes ................................................................. 6-4
  - Oracle DM Server Probes ............................................................................... 6-5
7 Configuring Pipeline Manager

About Configuring Pipeline Manager ................................................................. 7-1
Using Registry Files to Configure Pipeline Manager ...................................... 7-1
   About the Registry File Structure ................................................................. 7-1
   About the Registry File Syntax .................................................................... 7-3
   About the Sample Registry Files ................................................................ 7-4
About Configuring Pipelines ........................................................................... 7-4
   About Configuring Function Modules ....................................................... 7-5
   About iScripts and iRules ............................................................................ 7-7
      About Configuring iScripts .................................................................... 7-7
      About Configuring iRules ....................................................................... 7-8
   Configuring Multiple Instances of a Pipeline .......................................... 7-8
About Configuring Multiple Instances of Sequencers, Output Streams, or System Brands ........................................ 7-9
   Configuring Multiple Instances of Sequencers, Output Streams, or System Brands ........................................ 7-11
8 Controlling Batch Operations

About the Batch Controller ................................................................. 8-1
Setting Activity Times and Triggers ...................................................... 8-1
    General Batch Controller Parameters .............................................. 8-1
        Connection Parameters ............................................................... 8-1
        Time-to-Run Parameters ............................................................ 8-2
        Log-File Parameter ................................................................. 8-2
        Timeout-Limit Parameters ......................................................... 8-2
        Example of Parameters ........................................................... 8-3
    Handler Identification ................................................................. 8-3
    Occurrence-Driven Execution ......................................................... 8-4
    Timed Execution ........................................................................... 8-5
        Metronomic Execution ............................................................... 8-5
        Scheduled Execution ............................................................... 8-7
Starting the Batch Controller ............................................................ 8-8
About SampleHandler ....................................................................... 8-8
    Copying SampleHandler ............................................................... 8-8
    Customizing SampleHandler ......................................................... 8-8
    Configuring the Batch Controller .................................................. 8-10
    Starting the New Batch Handler .................................................... 8-10

9 About Connection Pooling

Overview ........................................................................................... 9-1
Configuring the Connection Pool ....................................................... 9-1
    Infranet.properties File Connection Pool Parameters ....................... 9-2
Connection Pool Error Handling ......................................................... 9-2
Monitoring Connection Pooling Events ............................................. 9-3

10 System Administration Utilities and Scripts

    load_pin_event_record_map .......................................................... 10-2
    partition_utils ............................................................................. 10-4
    partitioning.pl ............................................................................. 10-10
    pin_clean_asos .......................................................................... 10-12
    pin_clean_rsvns .......................................................................... 10-14
    pin_ctl ...................................................................................... 10-16
    pin_db_alert.pl ........................................................................... 10-20
    pin_purge .................................................................................. 10-21
13 Configuring a High-Availability System

About Setting Up a High-Availability System ................................................................. 13-1
Configuring the BRM Database for High Availability .................................................. 13-3
  Setting Up Oracle RAC for Failover in a High-Availability System ......................... 13-4
  Configuring Oracle Database Services ................................................................. 13-5
  Defining Connections to the Oracle Database Services ........................................... 13-6
Minimizing Recovery Time of Oracle RAC Instances ............................................... 13-7
Configuring IMDB Cache Manager for High Availability ........................................... 13-8
  Creating and Configuring Active Data Stores ....................................................... 13-11
  Initializing an Active Data Store ........................................................................... 13-13
    Generating the Schema and Load SQL Files for the Active Data Store ............... 13-14
    Initializing the Data Store .................................................................................. 13-14
  Configuring the Cluster for an Active Data Store ................................................ 13-15
  Configuring Standby Data Stores ......................................................................... 13-16
  Creating Active and Standby Data Store Pairs ..................................................... 13-17
  Associating IMDB Cache DM Instances with Data Stores ...................................... 13-18
    Making a Data Store Accessible to IMDB Cache DM ....................................... 13-18
    Configuring an IMDB Cache DM Instance for a Data Store ............................. 13-19
    Configuring pin_ctl to Start and Stop IMDB Cache DM ................................... 13-19
  Configuring Connection Managers for High Availability ....................................... 13-20
    Connecting CMs to IMDB Cache DMs ............................................................... 13-21
  Registering IMDB Cache DM Instances with Oracle Clusterware ......................... 13-22
Configuring AAA Gateway Manager for High Availability ........................................ 13-25
  Restoring a High-Availability System After Failover ........................................... 13-27
  Rebalancing AAA Gateway Manager and CM Connections .................................. 13-28
  Switching Back to the Primary Oracle RAC Instance .......................................... 13-28
  Ensuring All Accounts are Billed ......................................................................... 13-29

Part III Improving Performance

14 Improving BRM Performance

  Monitoring Performance ......................................................................................... 14-1
  Improving Connection Manager Performance ...................................................... 14-1
    Increasing CM Login Performance ..................................................................... 14-2
      Using CM Proxy to Allow Unauthenticated Log On ........................................ 14-2
      Turning Off Session-Event Logging .................................................................. 14-2
      Turning Off the Checking of Logons and Passwords ....................................... 14-3
    Load Balancing CMs ......................................................................................... 14-3
    Improving Performance for Loading Large Price Lists ....................................... 14-3
    Improving Real-Time Rating Performance ...................................................... 14-4
      Changing the Precision of Rounded and Calculated Values ......................... 14-5
      Setting the Interval for Checking for Price List Changes ............................. 14-5
      Setting the Interval for Updating Zone Maps ............................................... 14-5
      Filtering the ERAs Considered During Rating and Discounting .................... 14-5
      Enabling and Disabling the Caching of Customized Products ....................... 14-7
      Configuring the Maximum Number of Products and Discounts Cached .......... 14-8
Improving the Performance of your Multithreaded Applications ........................................... 14-9
Controlling Thread Load on Your Multithreaded Application ................................................... 14-9
Monitoring the Thread Activity of Your Multithreaded Application ........................................... 14-10
Logging Non-Currency Events .................................................................................................. 14-11
Specifying the Number of Connections to CMs ....................................................................... 14-11
Setting the CM Time Interval between Opcode Requests ......................................................... 14-12

Using Connection Manager Master Processes ...................................................................... 14-12
Sample CMMP Configuration .....................................................................................................
Setting Up a CMMP .................................................................................................................. 14-14

Improving Data Manager and Queue Manager Performance .................................................. 14-14
About Queuing-Based Processes ............................................................................................... 14-14
Example of Queuing in a Client-to-CM Connection ................................................................. 14-15
Configuring DM Front Ends and Back Ends ............................................................................ 14-16
Ratio of Front Ends to Back Ends ............................................................................................. 14-17
Providing Enough Front-End Connections ............................................................................ 14-17
Determining the Required Number of Back Ends .................................................................... 14-18
Determining the Maximum Number of Back Ends Dedicated to Transactions ....................... 14-19
Setting the DM Time Interval between Opcode Requests ....................................................... 14-19
Setting DM Shared Memory Size ............................................................................................ 14-19
Determining DM Shared Memory Requirements ..................................................................... 14-20
How BRM Allocates Shared Memory for Searches .................................................................. 14-21
Shared Memory Guidelines ..................................................................................................... 14-22
Reducing Resources Used for Search Queries .......................................................................... 14-22
Load Balancing DMs ................................................................................................................. 14-23
Optimizing Memory Allocation during Database Searches ....................................................... 14-23
Improving BRM Performance During Database Searches ....................................................... 14-23
Increasing DM CPU Usage ....................................................................................................... 14-24
Examples of DM Configurations ............................................................................................... 14-24

Improving Interprocess Communication (IPC) Performance .................................................. 14-25
Sample Configuration Settings for a Multidatabase System ..................................................... 14-26

Improving Performance by Disabling Unused Features .......................................................... 14-27
Tuning Billing Performance Using Configuration File .............................................................. 14-28
Filtering Search Results ........................................................................................................... 14-29
Tuning the Number of Children for Billing Utilities ................................................................. 14-30
Tuning the Account Cache Size for Billing Utilities (fetch_size) ............................................ 14-30
Tuning the Batch Size for Billing Utilities (per_batch) ............................................................. 14-30
Tuning the Batch Size for the pin_collect Utility .................................................................. 14-31
Specifying the Number of Retries in Case of a Deadlock ......................................................... 14-31
Rearranging Accounts to Improve Billing Performance ............................................................ 14-31
Improving Performance in Retrieving Purchased Offerings For a Bill Unit ............................. 14-33
Tuning Memory for Billing ....................................................................................................... 14-34
Additional Issues Related to Billing Performance ..................................................................... 14-34
The Effect of the Number of Events on Billing ........................................................................ 14-34

Tuning Billing Performance Using Business Parameters ....................................................... 14-35
Configuring to Exclude Searches on Closed Offerings ........................................................... 14-35
Configuring to Exclude Searches on Overridden Products .................................................... 14-36

Tuning Invoicing Performance ................................................................................................. 14-37
15 Optimizing Pipeline Manager Performance

Pipeline Manager Optimization Overview .......................................................... 15-1
Key Metrics for Measuring Performance ......................................................... 15-1
About Measuring Pipeline Manager Performance ........................................... 15-2
Information Requirements .............................................................................. 15-2
Testing Requirements ..................................................................................... 15-2
Optimizing Pipeline Manager ........................................................................... 15-2
Troubleshooting Pipeline Performance ............................................................ 15-4

Optimizing Function Modules ......................................................................... 15-4

Configuring Single-Threaded or Multithreaded Operation ................................ 15-5
Reducing Startup Times with Parallel Loading .................................................. 15-5
Assigning Multiple Threads to Process Function Modules .............................. 15-6
Configuring the DAT_AccountBatch Module Database Connections and Threads .................................................. 15-7
Setting the Hash Map Size for Threads ............................................................. 15-7
Locking Objects During DAT_AccountBatch Processing ................................ 15-8
Configuring Threads for DAT_BalanceBatch Connections ............................ 15-9
16 Optimizing BRM for Prepaid and Postpaid Convergence

About Convergent BRM Systems ................................................................. 16-1
About Cache Residency ............................................................................ 16-1
How Cache Residency is Determined ................................................................. 16-2
About Setting Up Business Profiles for Cache Residency Distinction .............. 16-3
How BRM Caches Objects in a Convergent Rating System............................... 16-4
About Changing the Cache Type of an Object ................................................... 16-4
Configuring BRM for Cache Residency Distinction .......................................... 16-6
   Enabling Cache Residency Distinction ........................................................ 16-6
   Assigning Objects a Cache Residency Value ................................................. 16-7
   Setting Up Cache Residency Validation Rules ............................................. 16-7
   Changing the Default Business Profile ....................................................... 16-8
   Overriding the Default Business Profile ..................................................... 16-10
About Selective Account Loading ...................................................................... 16-10
   Configuring Pipeline Manager for Selective Account Loading .................... 16-10
      Enabling Selective Account Loading ..................................................... 16-10
   Configuring Pipeline Manager to Process Prepaid CDRs ............................. 16-11
Customizing Cache Residency Distinction ...................................................... 16-12
   Configuring Nonreference Objects for Cache Residency Distinction ............ 16-12

Part IV Troubleshooting BRM

17 Resolving Problems in Your BRM System

General Checklist for Resolving Problems with BRM ....................................... 17-1
Using Error Logs to Troubleshoot BRM .......................................................... 17-2
   Finding Error Log Files ............................................................................. 17-2
   Understanding Error-Message Syntax ...................................................... 17-2
   Resolving Clusters of Error Messages ..................................................... 17-3
Interpreting Error Messages .......................................................................... 17-3
   Example 1: Failure of a Client Application .............................................. 17-3
   Example 2: Problem with Customer Center ............................................. 17-4
   Example 4: Getting More Information about Oracle Errors ....................... 17-5
Diagnosing Some Common Problems with BRM .............................................. 17-5
   Problems Starting BRM Components ...................................................... 17-5
      Problem: Bad Bind, Error 13 ................................................................. 17-5
      Problem: Bad Bind, Error 125 ............................................................. 17-6
      Problem: Cannot Connect to Oracle Database .................................... 17-6
      Problem: ORA-01502: Index 'PINPAPI_EVENT_ITEM_OBJ_ID' or Partition of Such Index Is in Unusable State 17-7
   Problems Stopping BRM Components ....................................................... 17-7
      Problem: No Permission to Stop the Component .................................. 17-7
      Problem: No pid File ........................................................................... 17-7
   Problems Connecting to BRM .................................................................... 17-7
      Problem: Cannot Connect to the Database ........................................... 17-8
      Problem: Cannot Connect to the CM .................................................... 17-8
      Problem: CM Cannot Connect to a DM ............................................... 17-9
   Problems with Deadlocking ........................................................................ 17-9
      Problem: BRM “Hangs” or Oracle Deadlocks ....................................... 17-9
      Problem: dm_oracle Cannot Connect to the Oracle Database ............... 17-9
18 Reference Guide to BRM Error Codes

Interpreting BRM Error Codes ................................................................. 18-1
BRM Error Locations ........................................................................... 18-1
BRM Error Classes ............................................................................. 18-2
BRM Error Codes ................................................................................ 18-3

19 Pipeline Manager Error Messages

Pipeline Framework Error Messages .................................................... 19-1
Pipeline Manager Module Error Messages ......................................... 19-13
   DAT_AccountBatch ................................................................. 19-14
   DAT_AccountRealtime ........................................................... 19-17
   DAT_BalanceBatch ................................................................. 19-17
   DAT_BalanceRealtime ............................................................. 19-19
   DAT_ConnectionManager ......................................................... 19-19
   DAT_ConnectionPool ............................................................... 19-19
   DAT_Currency ................................................................. 19-20
   DAT_Discount .................................................................. 19-20
   DAT_ExchangeRate .............................................................. 19-20
   DAT_InterConnect ................................................................. 19-21
   DAT_ItemAssign ................................................................. 19-21
   DAT_Listener .................................................................. 19-21
   DAT_ModelSelector ............................................................... 19-22
   DAT_NumberPortability ........................................................... 19-23
   DAT_PortalConfig ................................................................. 19-23
   DAT_Price .................................................................. 19-23
   DAT_Rateplan .................................................................. 19-24
   DAT_Recycle ................................................................ 19-24
   DAT_ResubmitBatch ............................................................... 19-24
   DAT_ScenarioReader .............................................................. 19-25
   DAT_USC_Map ................................................................ 19-25
   DAT_Zone .................................................................. 19-26
   EXT_InEasyDB ................................................................. 19-26
   EXT_PipelineDispatcher ............................................................ 19-26
   FCT_Account ................................................................ 19-27
Pipeline Utility Error Messages

OMF Error Messages

FCT_AccountRouter .......................................................................................................................
FCT_AggreGate..........................................................................................................................
FCT_APN_Map .............................................................................................................................
FCT_ApplyBalance .....................................................................................................................
FCT_BatchSuspense ...................................................................................................................
FCT_BillingRecord .....................................................................................................................
FCT_CallAssembling ...................................................................................................................
FCT_CarrierIcRating ....................................................................................................................
FCT_CiberOcc .............................................................................................................................
FCT_CliMapping ..........................................................................................................................
FCT_CustomerRating ....................................................................................................................
FCT_DataDump ............................................................................................................................
FCT_Discard ..................................................................................................................................
FCT_DroppedCall ..........................................................................................................................
FCT_EnglishSplitting ....................................................................................................................
FCT_ExchangeRate ......................................................................................................................
FCT_Filter_Set ..............................................................................................................................
FCT_IRules ......................................................................................................................................
FCT_IScriptPlugIn .......................................................................................................................}
FCT_ItemAssign ............................................................................................................................
FCT_MainRating ............................................................................................................................
FCT_Opcode ..................................................................................................................................
FCT_PreSuspense ..........................................................................................................................
FCT_PreRating ...............................................................................................................................
FCT_Reject ....................................................................................................................................
FCT_Rounding ..............................................................................................................................
FCT_SegZoneNoCust ......................................................................................................................
FCT_Timer ......................................................................................................................................
FCT_UoM_Map ..............................................................................................................................
FCT_UsageClassMap .....................................................................................................................
FCT_USC_Map ...............................................................................................................................}
INP_GenericStream .......................................................................................................................}
INP_Realtime ...............................................................................................................................}
INP_Recycle .................................................................................................................................}
NET_EM ........................................................................................................................................}
OUT_DB .......................................................................................................................................}
OUT_GenericStream .....................................................................................................................}
OUT_Realtime ..............................................................................................................................}

Pipeline Utility Error Messages ..................................................................................................
LoadIFWConfig .............................................................................................................................
OMF Error Messages .....................................................................................................................
Part V Managing IMDB Cache-Enabled Systems

20 Using Oracle IMDB Cache Manager

About Oracle IMDB Cache Manager ................................................................. 20-1
About Caching BRM Objects In Memory ......................................................... 20-1
About BRM Cache Groups ............................................................................. 20-1
Guidelines for Setting Cache Group Size ....................................................... 20-2
About Loading BRM Objects into IMDB Cache ............................................. 20-3
About Loading Migrated Accounts into the Cache Groups ......................... 20-3
About Managing Data in IMDB Cache ........................................................... 20-3
Managing Fast-Growing Tables ..................................................................... 20-3
About Purging Expired Reservation and Active-Session Objects ................. 20-3
About Purging Closed Bills, Items, Journals, and Expired Sub-Balances ....... 20-4
Using the IMDB Cache Grid to Partition Data ................................................ 20-4
How Accounts Are Assigned to the Logical Partition ................................... 20-6
How the CM Locates Accounts in the Logical Partition ............................... 20-7
About Setting Up the CM for Logical Partitioning .......................................... 20-7
How IMDB Cache DM Connects to IMDB Cache ........................................ 20-8
How IMDB Cache DM Handles Both IMDB Cache and BRM Database Transactions 20-8
About the AAA Flow in a BRM System with IMDB Cache Manager ......... 20-9

21 Creating and Initializing the BRM Cache Group Schema

About Creating and Initializing BRM Cache Group Schema ............................ 21-1
Creating and Initializing Your Cache Group Schema ................................. 21-1
Configuring the pin_tt_schema_gen.values File ........................................... 21-2
Defining Cache Group Information ............................................................... 21-3
Defining Cache Group Information for Special Cases .................................. 21-4
Setting On Delete Cascade ......................................................................... 21-5
Defining the Cache Group Aging Policy ......................................................... 21-5
Defining Transient Local Tables ................................................................... 21-6
Supporting IMDB Cache Data Types ............................................................. 21-6
Generating Load SQL ................................................................................... 21-6
Defining the Logical Partition Database Number ........................................ 21-7
Generating Your Schema and Load SQL Scripts .......................................... 21-7
Generating Scripts for A Basic BRM System ............................................... 21-7
Generating Scripts for A BRM System with Logical Partitioning ............... 21-8

22 Customizing BRM for In-Memory Database Cache DM

About Customizing the Cache Groups ............................................................. 22-1
Customizing the Default BRM Cache Groups ................................................ 22-1
About Extending Tables in the Default BRM Cache Groups ....................... 22-2
Creating a Custom Cache Group .................................................................... 22-2
Creating Custom Storable Classes ................................................................. 22-3
Assigning Custom Objects a Residency Value ............................................. 22-3
About Extending Event Type Storable Classes ............................................. 22-3
PIN_GET_SCHEMA_NO ................................................................................. 22-4
# Partitioning and Managing BRM Database Tables

## 23 Partitioning Database Tables

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>About Partitioning</td>
<td>23-1</td>
</tr>
<tr>
<td>Partitioning Overview</td>
<td>23-3</td>
</tr>
<tr>
<td>About Partitioning Schemes</td>
<td>23-4</td>
</tr>
<tr>
<td>About Nonpurgeable Events</td>
<td>23-5</td>
</tr>
<tr>
<td>About Objects Stored in partition_last and partition_last_pin</td>
<td>23-5</td>
</tr>
<tr>
<td>About the Default Partitioning Scheme</td>
<td>23-5</td>
</tr>
<tr>
<td>If You Did Not Create Default Partitions</td>
<td>23-6</td>
</tr>
<tr>
<td>Upgrade Partitioning Scheme</td>
<td>23-6</td>
</tr>
<tr>
<td>About Partitions for Delayed Events</td>
<td>23-6</td>
</tr>
<tr>
<td>Overview of Partition Schemes</td>
<td>23-7</td>
</tr>
<tr>
<td>About Managing Partitions</td>
<td>23-9</td>
</tr>
<tr>
<td>About Purging Objects</td>
<td>23-10</td>
</tr>
<tr>
<td>About Removing Partitions to Purge Objects</td>
<td>23-10</td>
</tr>
<tr>
<td>About Purging Events While Maintaining Partitions</td>
<td>23-11</td>
</tr>
<tr>
<td>About Running the partition_utils Utility</td>
<td>23-11</td>
</tr>
<tr>
<td>Running the partition Utils Utility in Test Mode</td>
<td>23-12</td>
</tr>
<tr>
<td>Configuring a Database Connection</td>
<td>23-13</td>
</tr>
<tr>
<td>Improving Performance When Using partition_utils</td>
<td>23-13</td>
</tr>
<tr>
<td>Restarting partition_utils</td>
<td>23-13</td>
</tr>
<tr>
<td>Adding Partitions</td>
<td>23-13</td>
</tr>
<tr>
<td>Enabling Delayed-Event Partitions</td>
<td>23-16</td>
</tr>
<tr>
<td>Disabling Delayed-Event Partitioning</td>
<td>23-17</td>
</tr>
<tr>
<td>Updating Partitions</td>
<td>23-17</td>
</tr>
<tr>
<td>Purging Objects by Removing Partitions</td>
<td>23-18</td>
</tr>
<tr>
<td>Purging Event Objects without Removing Partitions</td>
<td>23-19</td>
</tr>
<tr>
<td>Finding the Maximum POID for a Date</td>
<td>23-20</td>
</tr>
<tr>
<td>Customizing Partition Limitations</td>
<td>23-20</td>
</tr>
<tr>
<td>Customizing the List of Events Stored in partition_historic</td>
<td>23-21</td>
</tr>
</tbody>
</table>

## 24 Changing from a Nonpartitioned to a Partitioned Database

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>About Upgrading from a Nonpartitioned to a Partitioned Database</td>
<td>24-1</td>
</tr>
<tr>
<td>Software Required for Event Table Partitioning</td>
<td>24-1</td>
</tr>
<tr>
<td>Upgrading to a Partitioned Database</td>
<td>24-2</td>
</tr>
<tr>
<td>Increasing Disk Space for Event Tables and Indexes</td>
<td>24-2</td>
</tr>
<tr>
<td>Installing the Upgrade Package</td>
<td>24-2</td>
</tr>
<tr>
<td>(Optional) Reconfiguring the Parameters in partition.cfg</td>
<td>24-4</td>
</tr>
<tr>
<td>Merging the pin_setup.values File</td>
<td>24-4</td>
</tr>
<tr>
<td>Backing Up Your BRM Database</td>
<td>24-4</td>
</tr>
<tr>
<td>Running the Upgrade Scripts</td>
<td>24-4</td>
</tr>
<tr>
<td>Adding Purgeable Partitions to Event Tables</td>
<td>24-4</td>
</tr>
<tr>
<td>Restarting BRM</td>
<td>24-4</td>
</tr>
<tr>
<td>About the Upgrade Scripts and Files</td>
<td>24-5</td>
</tr>
</tbody>
</table>
25 About Purging Data

**About Purging Event Objects** ........................................................................................................... 25-1
Categories of BRM Event Objects ........................................................................................................... 25-1
  Event Objects That Have a Balance Impact ................................................................................................ 25-1
  Event Objects That Do Not Have a Balance Impact ................................................................................ 25-2
  Event Objects Stored in partition_historic ................................................................................................. 25-2
Impact of Purging Event Objects .............................................................................................................. 25-3
  Billing Event Objects ............................................................................................................................... 25-3
  Accounts Receivable Event Objects ...................................................................................................... 25-8
  Delayed Event Objects ............................................................................................................................. 25-9
  Group Event Objects ............................................................................................................................... 25-9
  Sharing Group Event Objects ............................................................................................................... 25-10
  Session Event Objects ............................................................................................................................. 25-10
  Auditing Event Objects ............................................................................................................................. 25-10
About Purging Account Sub-Balances ...................................................................................................... 25-11

26 Generating Virtual Columns on Event Tables

**About Generating Virtual Columns on Event Tables** ........................................................................... 26-1
**Generating Virtual Columns on Event Tables** .................................................................................. 26-2
  Viewing Tasks for Generating Virtual Columns .................................................................................. 26-3
  Viewing and then Running Virtual-Column Tasks .............................................................................. 26-4
  Viewing Virtual-Column Task Details ................................................................................................. 26-5
**Exporting a BRM Schema with Virtual Columns** ............................................................................. 26-5

Part VII Managing a Multidatabase System

27 Managing a Multidatabase or Multischema System

**About Multidatabase Systems** ............................................................................................................. 27-1
  Using Pipeline Manager with Multiple Databases ............................................................................... 27-2
Converting a Single-Database System to a Multidatabase System .......................................................... 27-2
Preparing to Manage a Multidatabase System ....................................................................................... 27-2
Adding Databases to a Multidatabase System .......................................................................................... 27-2
  Configuring the pin_multidb.conf File on the Primary Installation Machine ...................................... 27-3
Setting Database Status ............................................................................................................................ 27-4
Setting Database Priorities ....................................................................................................................... 27-5
Changing Refresh Intervals for BRM Groups ............................................................................................ 27-6
  Changing the Refresh Interval ................................................................................................................ 27-6
Forcing a Database Refresh ........................................................................................................................ 27-7
Creating Custom Tables for Replication .................................................................................................. 27-7
  Creating Custom Tables You Want Replicated to the Secondary Databases ........................................ 27-8
  Creating Custom Tables That Are Available to All Databases ............................................................ 27-8
Synchronizing Database Data Dictionaries ............................................................................................... 27-9
Synchronizing the Database/uniqueness Objects ...................................................................................... 27-9
Changing the Interval for Checking New Accounts ................................................................................ 27-10
Changing the Interval for Updating the Distribution Table ..................................................................... 27-11
28 Multidatabase Utilities

load_config_dist ................................................................. 28-2
load_pin_uniqueness .......................................................... 28-3
pin_config_distribution ....................................................... 28-4
pin_confirm_logins ............................................................. 28-5
pin_mta_monitor ............................................................... 28-6
pin_multidb ......................................................................... 28-8

Part VIII Migrating Accounts from One Database to Another

29 Understanding Account Migration Manager

About AMM ............................................................................ 29-1
When to Redistribute Database Loads ................................. 29-1
Scheduling Account Migration ............................................ 29-1
About Migrating Accounts When the Pipeline Manager is Online 29-1
How AMM Works ............................................................... 29-2
  Deleting Migrated Objects ............................................... 29-2
About Migrating Hierarchical, Sponsorship, and Resource Sharing Accounts .... 29-2
  About Searching for Member and Non-Group Member Accounts ................... 29-3
  About Account Groups ..................................................... 29-3
  About Migrating Account Groups ....................................... 29-4
AMM Process Overview .................................................... 29-4
  About the Account Search Configuration File ........................ 29-5
  About the pin_amt Utility .................................................. 29-5
  About the AMM Controller ............................................... 29-6
    How AMM Controller Processes Batches with Non-Group Members ............... 29-6
    How AMM Controller Processes Batches with Account Group Members .......... 29-6
  About the AMM Mover ...................................................... 29-7
About Distributed Transactions .......................................... 29-8
Account Migration Restrictions .......................................... 29-8
  Account Activity Prevented during Account Migration ......................... 29-9
  Do Not Rerate Events during Account Migration ................................... 29-9
  Do Not Alter Account Group Members ....................................... 29-9
  Migration Prevented during Account Activity ...................................... 29-9
  Brand and Remittance Accounts Not Migrated ................................... 29-9
  Unique POIDs Required across All Databases .................................... 29-9
  Some Client Applications May Fail during Account Migration ................ 29-10
  AMM Does Not Support Some BRM Components ................................. 29-10
  Using BRM Reports after Migration ......................................... 29-10
About Using Multiple AMM Controllers ............................. 29-11
Account Migration Performance .......................................... 29-11
About AMM Job Management Tables ................................. 29-11
About Job Status Flags ........................................................ 29-12
30 Installing and Configuring Account Migration Manager

System Requirements ........................................................................................................ 30-1
   Software Requirements ................................................................................................. 30-1
Installing AMM .................................................................................................................. 30-2
   Configuring All Primary and Secondary Database Machines ...................................... 30-2
   Installing the AMM Software on the Primary Installation Machine ......................... 30-4
   Configuring your Oracle DM to Check for Invalid Objects ........................................ 30-6
Connecting AMM to Your Databases ................................................................................ 30-6
   Sample Infranet.properties File .................................................................................. 30-9
   Configuring the JDBC Driver Jar File for BRM ............................................................ 30-10
   About Batch Status Flags ........................................................................................... 30-13
   About Group Status Flags ........................................................................................... 30-13
What’s Next? ..................................................................................................................... 30-10
Configuring AMM for Additional Databases .................................................................... 30-10
Configuring AMM for New Custom Tables ..................................................................... 30-11
Tuning Your Database for Optimal Account Migration Performance ................................. 30-11

31 Migrating Accounts with the Pipeline Manager Running

About Migrating Accounts When Pipeline Manager Is Online ........................................... 31-1
   How AMM Interacts with Your Pipelines during Account Migration ............................ 31-2
   About Waiting before Migrating Accounts .................................................................. 31-3
   About Starting Multiple Jobs Concurrently ................................................................. 31-3
   About Notifying the Pipelines About Account Migration ............................................. 31-3
   About AMM Business Events ....................................................................................... 31-4
   About Sending AMM Business Events to the Pipelines .............................................. 31-4
   About Notifying AMM About EDR Processing ............................................................. 31-5
   About Acknowledgment Events .................................................................................... 31-5
   About Sending Acknowledgments to AMM ................................................................. 31-6
   About the Account Router Instance of the Pipeline Manager ...................................... 31-6
   About Suspending Call Records ................................................................................... 31-7
   About Reprocessing Suspended Call Records ............................................................. 31-8
Configuring Your System to Migrate Accounts When the Pipeline Manager Is Running ... 31-9
   Configuring Your Account-Router Pipeline Manager ..................................................... 31-9
   Configuring Your Routing Pipeline .............................................................................. 31-9
   Configuring Your Pre-Recycling Pipeline .................................................................... 31-10
   Configuring Your Re-Suspending Pipeline ................................................................... 31-11
   Configuring the Data Pool ............................................................................................ 31-12
   Configuring BRM to Handle Suspended EDRs ............................................................. 31-12
   Configuring AMM to Send Business Events to Your Pipelines .................................... 31-13
   Connecting AMM to the Primary CM ......................................................................... 31-13
   Configuring Account Synchronization ......................................................................... 31-13
   Configuring Your Pipelines to Dequeue AMM Business Events .................................. 31-15
   Configuring Your Pipelines to Send Acknowledgments to AMM ................................. 31-15
   Creating the Acknowledgment Queue .......................................................................... 31-16
   Connecting AMM Directly to Your Acknowledgment Queue ...................................... 31-16
   Configuring Your Pipelines to Send Acknowledgment Events ..................................... 31-16
32 Using Account Migration Manager

Overview of Account Migration Tasks ................................................................. 32-1
Creating the Account Search Configuration File .................................................. 32-1
  Sample Account Search Configuration File ......................................................... 32-3
Submitting the Account Search File .................................................................. 32-4
Enabling Migration Jobs in the Queue ............................................................... 32-4
Starting the AMM Controller ............................................................................ 32-4
Monitoring the AMM Controller ...................................................................... 32-4
  Checking the AMM Controller Status .............................................................. 32-4
  Checking the AMM Controller Log File ........................................................... 32-5
  Monitoring the AMM Controller in Real Time ................................................. 32-5
Monitoring Account Migration ........................................................................ 32-5
  Monitoring Job Status ...................................................................................... 32-5
  Checking Job Details ....................................................................................... 32-6
  Checking Account Group Details .................................................................... 32-7
Handling Account Migration Failures .............................................................. 32-7
  Finding Debugging Information ....................................................................... 32-7
  Reprocessing Failed Batches .......................................................................... 32-8
Purging Migrated Objects from the Source Database ..................................... 32-8
Deleting Jobs from the Source Database .......................................................... 32-8
Stopping the AMM Controller .......................................................................... 32-9
Pausing and Resuming Account Migration ...................................................... 32-9
Automating Account Migration ........................................................................ 32-9

33 Modifying Applications to Work with AMM

Modifying Custom Client Applications for AMM ............................................. 33-1
Modifying Custom BRM Reports for AMM ....................................................... 33-2
AMM Return Codes and Messages .................................................................. 33-2

34 Modifying the Account Migration Manager

Creating Custom Account Search Criteria ......................................................... 34-1
  Creating a Search Template ............................................................................ 34-1
  Sample Search Template ................................................................................. 34-2
Adding New Entries to the Account Search Configuration File .................... 34-2
  Sample Account Search Configuration File ..................................................... 34-2
Implementing and Compiling the Conversion Interface .................................. 34-3
  Sample Class Implementing Conversion Interface ........................................ 34-3
Verifying Your Search Criteria ......................................................................... 34-3
  Verifying That the Search Criteria Creates Valid SQL Statements ............. 34-4
  Verifying That the Search Criteria Finds Correct Accounts ....................... 34-4

35 AMT ER Diagram

36 Account Migration Manager Utilities

pin_amt .............................................................................................................. 36-2
## Part IX  Configuration File Reference

### 37 Business Logic pin.conf Reference

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accounts Receivable pin.conf Entries</td>
<td>37-1</td>
</tr>
<tr>
<td>Billing pin.conf Entries</td>
<td>37-3</td>
</tr>
<tr>
<td>Customer Management pin.conf Entries</td>
<td>37-6</td>
</tr>
<tr>
<td>Discounting pin.conf Entries</td>
<td>37-7</td>
</tr>
<tr>
<td>General Ledger pin.conf Entries</td>
<td>37-9</td>
</tr>
<tr>
<td>Invoicing pin.conf Entries</td>
<td>37-9</td>
</tr>
<tr>
<td>Payments pin.conf Entries</td>
<td>37-9</td>
</tr>
<tr>
<td>Pricing and Rating pin.conf Entries</td>
<td>37-11</td>
</tr>
<tr>
<td>Registration pin.conf Entries</td>
<td>37-14</td>
</tr>
<tr>
<td>Revenue Assurance pin.conf Entries</td>
<td>37-16</td>
</tr>
<tr>
<td>Services Framework pin.conf Entries</td>
<td>37-17</td>
</tr>
<tr>
<td>Tax Calculation pin.conf Entries</td>
<td>37-19</td>
</tr>
</tbody>
</table>

### 38 System Administration pin.conf Reference

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connection Manager (CM) pin.conf Entries</td>
<td>38-1</td>
</tr>
<tr>
<td>Data Manager (DM) pin.conf Entries</td>
<td>38-3</td>
</tr>
<tr>
<td>EAI Manager pin.conf Entries</td>
<td>38-6</td>
</tr>
<tr>
<td>Multithreaded Application (MTA) Framework pin.conf Entries</td>
<td>38-7</td>
</tr>
</tbody>
</table>

### 39 business_params Reference

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accounts Receivable business_params Entries</td>
<td>39-1</td>
</tr>
<tr>
<td>Billing business_params Entries</td>
<td>39-2</td>
</tr>
<tr>
<td>Customer Management business_params Entries</td>
<td>39-5</td>
</tr>
<tr>
<td>General Ledger business_params Entries</td>
<td>39-6</td>
</tr>
<tr>
<td>Invoicing business_params Entries</td>
<td>39-6</td>
</tr>
<tr>
<td>Pricing and Rating business_params Entries</td>
<td>39-7</td>
</tr>
<tr>
<td>Selective Account Loading business_params Entries</td>
<td>39-8</td>
</tr>
<tr>
<td>Subscription business_params Entries</td>
<td>39-8</td>
</tr>
<tr>
<td>System Administration business_params Entries</td>
<td>39-9</td>
</tr>
</tbody>
</table>
This book describes system administration tasks for Oracle Communications Billing and Revenue Management (BRM).

**Audience**

This document is intended for system administrators who maintain and manage the BRM system.

**Downloading Oracle Communications Documentation**

Product documentation is located on Oracle Technology Network:

http://docs.oracle.com

Additional Oracle Communications documentation is available from Oracle E-Delivery:

http://edelivery.oracle.com

**Documentation Accessibility**

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at


**Access to Oracle Support**

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

**Document Revision History**

The following table lists the revision history for this book.

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>E25102-01</td>
<td>March 2013</td>
<td>Initial release.</td>
</tr>
<tr>
<td>Version</td>
<td>Date</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>---------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| E25102-02   | July 2013     | Documentation updates for BRM 7.4 Patch Set 18.  
  - Made minor formatting and text changes.  
  - Added an entry for the NonCurrencyResourceJournaling business parameter in "Billing business_params Entries". |
| E25102-03   | October 2013  | Documentation updates for BRM 7.4 Patch Set 19.  
  - Made minor formatting and text changes.  
  - Added an entry for SplitSponsorItemsByMember in "Billing business_params Entries". |
  - Added an entry for the UseActualBilledTimeForGLReport business parameter in "General Ledger business_params Entries". |
  - Updated the following sections:  
    - Suspense Management Center Permission Types  
    - Setting Up Email Alert Notifications |
Part I

Basic BRM System Administration

Part I describes basic Oracle Communications Billing and Revenue Management (BRM) system administration tasks. It contains the following chapters:

- Starting and Stopping the BRM System
- Monitoring and Maintaining Your BRM System
- Using Configuration Files to Connect and Configure Components
- Implementing System Security
- Using HP OpenView to Monitor and Control BRM
- BRM OMF Instrumented Objects
- Configuring Pipeline Manager
- Controlling Batch Operations
- About Connection Pooling
- System Administration Utilities and Scripts
- SNMP Utilities
Starting and Stopping the BRM System

This chapter explains how to start and stop your Oracle Communications Billing and Revenue Management (BRM) system.

Before you read this document, you should be familiar with how BRM works. See "BRM System Architecture" in BRM Concepts.

For information about using HP OpenView to start and stop BRM components, see "Managing BRM Components in HP OpenView".

About Starting and Stopping BRM Components

You start and stop BRM components by starting and stopping the corresponding process for that component.

You can start and stop BRM components by using the following methods:

- HP OpenView. See "Managing BRM Components in HP OpenView".
- The pin_ctl utility. See "Starting a Component by Using the pin_ctl Utility".
- Pipeline Manager ifw command.

Choosing the User Name for BRM Components

You can start BRM components as user root, user pin, or any other name you choose. If you start both the BRM database and the BRM components with a user name other than root, you have better control over security and administrators that do not have superuser permissions.

**Note:** If you use a port number less than 1000 for a component (1023 for the cm_proxy component), you must start that component as the user root. If you use a port number greater than 1024, you do not have to start the component as the user root.

Starting Several BRM Components in Sequence

You must start BRM components in a specific order, starting with the data tier and finishing with the application tier.

The tiers are structured as shown in Figure 1–1:
To start multiple components:

1. Start the BRM database.
   The BRM database usually starts automatically when the computer is started. For information on stopping and starting the database, see the documentation for your database server.

2. Start the Data Manager (DM) for your database (`dm_oracle`).

   **Important:** In multidatabase systems, you must start all secondary DMs before you start the primary DM.

3. Start any other DMs, such as these:
   - `dm_fusa` for BRM-initiated payment processing
   
   **Note:** Start the Paymentech credit card simulator before starting DM FUSA. See “Running the Paymentech Simulators” in BRM Configuring and Collecting Payments.

4. Start the daemon or process for any optional features such as these:
   - Mailer
   - Popper
   - RADIUS
   - Web interface
   - Tax

5. Start the Connection Manager Master Processes (CMMPs) if your system uses them.
Starting a Component by Using the pin_ctl Utility

6. Start CM Proxy if your system uses it.
7. Start the CMs.
8. Start BRM clients and other programs, such as optional service integration components.

Tip: Create a custom start script that includes required `pin_ctl start` `component` commands and `start_component` scripts. Be sure to start components in the proper order.

Stopping Several BRM Components in Sequence

You must stop BRM components in a specific order, starting with the application client tier and finishing with the data server tier.

1. Stop the BRM client applications and optional service integration components.
2. If your system uses any of these optional features, stop each daemon or process:
   ■ Mailer
   ■ Popper
   ■ RADIUS
   ■ Web interface
   ■ Tax
3. Stop the CMs.
4. Stop CM proxy if your system uses it.
5. Stop the Connection Manager Master Processes (CMMPs) if your system uses them.
6. Stop all but the database DMs, such as `dm_vertex` for tax calculation
7. Stop the Oracle DM for your database (`dm_oracle`).
8. Close all database sessions, such as SQL*Plus sessions for Oracle.
9. Stop the BRM database.

For information on stopping and starting the database, see the documentation for your Oracle database server.

Starting a Component by Using the pin_ctl Utility

You can use the `pin_ctl` utility to perform the following:

■ Start BRM components. See "Starting BRM Components with pin_ctl".
■ Start Pipeline Manager components. See "Starting Pipeline Manager Components with pin_ctl".
■ Retrieve diagnostic data when you start a component. See "Getting Diagnostic Data When You Start a Component".
■ Restart a component. See "Halting and Restarting a Component with One Command".
■ Start a component and clear the log file. See "Starting a Component and Clearing the Log File".
Starting a Component by Using the pin_ctl Utility

- Start a base set of BRM components. See "Starting a Base Set of BRM Components".

Starting BRM Components with pin_ctl

To start a BRM component by using the pin_ctl utility:

1. Go to the BRM_Home/bin directory.
2. Run the pin_ctl utility with the start action:
   ```
   pin_ctl start component
   ```
   where component is the component you want to start. For a list of valid component values, see "pin_ctl".
   For example:
   ```
   pin_ctl start dm_oracle
   ```

Starting Optional BRM Components with pin_ctl

By default, the pin_ctl utility is not configured to start BRM optional components, such as Synchronization Queue Manager DM (dm_aq) and Taxware DM (dm_taxware).

To start an optional BRM component by using the pin_ctl utility:

1. Add the optional component to the BRM_Home/bin/pin_ctl.conf configuration file. See "Customizing pin_ctl.conf for Starting and Stopping Optional Components".
2. Run the pin_ctl utility with the start action:
   ```
   pin_ctl start component
   ```
   where component is the optional component you added in step 1.
   For example, to start dm_aq:
   ```
   pin_ctl start dm_aq
   ```

Starting pin_radiusd with pin_ctl

To start pin_radiusd by using the pin_ctl utility:

1. Edit the BRM_Home/bin/pin_ctl.conf configuration file. See "Configuring the pin_ctl Utility".
2. In the startup configuration section of the pin_ctl.conf file, add the following lines:
   ```
   settings radius pin_Home_dir:BRM_Home radius:app
   start_radius cpidproc:radius: cport:Port_number
   ```
3. Copy the pin_radiusd executable from the BRM_Home/bin directory to the BRM_Home/apps/radius directory.
4. Go to the BRM_Home/bin directory.
5. Run the pin_ctl utility with the start action:
   ```
   pin_ctl start radius
   ```

Starting Pipeline Manager Components with pin_ctl

To start a BRM component by using the pin_ctl utility:
1. Edit the `BRM_Home/bin/pin_ctl.conf` configuration file. See "Configuring the pin_ctl Utility".

2. Copy the `Pipeline_Home/bin/ifw` binary file to `Pipeline_Home/bin/component`, where component is the pipeline component name.

3. Go to the `BRM_Home/bin` directory.

4. Run the `pin_ctl` utility with the `start` action:
   
   ```
   pin_ctl start component
   ```
   
   where component is the component you want to start. For a list of valid component values, see "pin_ctl".

---

### Getting Diagnostic Data When You Start a Component

To get diagnostic data about a BRM component by using the `pin_ctl` utility:

1. Go to the `BRM_Home/bin` directory.

2. Run the `pin_ctl` utility with the `start`, `-collectdata`, and component:
   
   ```
   pin_ctl start -collectdata component
   ```
   
   where component is the component you want to get data about. For a list of valid component values, see "pin_ctl".
   
   For example:
   
   ```
   pin_ctl start -collectdata dm_oracle
   ```

---

### Halting and Restarting a Component with One Command

To halt and restart a BRM component by using the `pin_ctl` utility:

1. Go to the `BRM_Home/bin` directory.

2. Run the `pin_ctl` utility with the `restart` action:
   
   ```
   pin_ctl restart component
   ```
   
   where component is the component you want to halt and restart. For a list of valid component values, see "pin_ctl".
   
   For example:
   
   ```
   pin_ctl restart dm_oracle
   ```

---

### Starting a Component and Clearing the Log File

You can clear the log file for a component and then start the component.

---

**Note:** If the component is already running, the command just clears the log file and the component continues running.

---

To start a BRM component and clear the contents of its log file by using the `pin_ctl` utility:

1. Go to the `BRM_Home/bin` directory.

2. Run the `pin_ctl` utility with the `cstart` action:
Starting BRM Components Automatically

```
  pin_ctl cstart component
```

where `component` is the component parameter.

For example:
```
  pin_ctl cstart dm_oracle
```

Starting a Base Set of BRM Components

You can use the `pin_ctl start all` command to start a customizable set of BRM components. By default, the components are started in this order:

- Oracle Data Manager (DM)
- Email DM
- Connection Manager
- CM Master Process
- Invoice formatter

You can customize the components that the `pin_ctl start all` command starts. You can also create customized “all” commands to start a separate set of components. See "Creating a Custom ‘all’ Parameter List”.

To start a set of BRM components by using the `pin_ctl` utility:

1. Go to the `BRM_Home/bin` directory.
2. Run the `pin_ctl` utility with the `start` action and `all` as the component:
```
  pin_ctl start all
```

**Note:** You can use the `-collectdata` parameter to collect diagnostic data about all of the components. See "Getting Diagnostic Data for a Component by Using the pin_ctl Utility”.

Starting BRM Components Automatically

You can configure BRM components to start automatically when you restart a system by adding component start scripts to the operating system startup scripts, such as the `/etc/rc2` script. You can also start components automatically from `cron` jobs.

**Important:** When you set up the BRM system to start automatically, make sure that the database starts before any Data Manager that connects to the database. If the DM starts first, it reports an error when it cannot find the database. For more information on component start sequences, see “Starting Several BRM Components in Sequence”.

To add a component to the startup script:

1. As user `root`, run the installation script for the component (for example, `install_dm_fusa` or `install_cm`).

   These scripts are in `BRM_Home/bin`.

2. (Optional) To avoid problems with file security and permissions, set all component processes to start as user `pin` (or another name you choose) rather than as user...
Starting Multiple Families of BRM Components on the Same Computer

Each BRM component can be part of a family of components. Each family includes a master process and one or more child processes. When you start the master process, BRM starts child processes automatically.

You can run multiple families of a BRM process on the same computer. For example, you can start another instance of the Paymentech DM on a computer that is already running the Paymentech DM.

To run multiple families, put each family in a separate directory with its own configuration file (pin.conf or Infranet.properties). Each family’s configuration file must point to a unique port to avoid conflicts when you start the processes. Each configuration file should point to a unique pinlog file as well.

For information on configuring the port and on the location of the log file for a process, see the explanatory text in the configuration file for that process.

Confirming That a BRM Component Started Successfully

To verify that a component started successfully, perform one or more of these checks:

- Display the system on HP OpenView. See “Using HP OpenView to Display System Status and Topology”.

  Note:  HP OpenView is not supported on Linux.

- For supported components, use the pin_ctl utility with the status action. See "pin_ctl".

- Look for the startup timestamp in the .log file. For more information on BRM log files, see "Using Logs to Monitor Components".

- Confirm that the pid file for the component contains the process ID (PID).
  In general, pid files are generated in pin_log_dir folders; for example, pin_log_dir/dm_oracle.

- Use the ps command to check the component process status.

- (Solaris and Linux) You can confirm that a shared memory segment has been allocated for the component process by using the ipcs command.

  Note:  The ipcs command does not show the shared memory segment unless you run it as root or pin or you use the -m parameter.

Stopping a BRM Component by Using the pin_ctl Utility

To stop a BRM component by using the pin_ctl utility:

1. Go to the BRM_Home/bin directory.

2. Run the pin_ctl utility with the stop action:
**Stopping a Process by Using Commands**

```
pin_ctl stop component
```

where `component` is the component you want to stop. For a list of valid `component` values, see "pin_ctl".

For example:
```
pin_ctl stop dm_oracle
```

**Getting Diagnostic Data When You Stop a Component**

For information about the diagnostic data, see "Getting Diagnostic Data for a Component by Using the pin_ctl Utility".

To get diagnostic data when you stop a BRM component:

1. Go to the `BRM_Home/bin` directory.
2. Run the `pin_ctl` utility with `stop`, `-collectdata`, and `component`:

```
pin_ctl stop -collectdata component
```

where `component` is the component you want to get data about. For a list of valid `component` values, see "pin_ctl".

For example:
```
pin_ctl stop -collectdata dm_oracle
```

**Stopping a Process by Using Commands**

In addition to using the `pin_ctl` utility, you can stop a component with a direct command. Use `ps` to get the process ID (PID) of the process, then use the `kill` command to stop the process.

**Note:** Stopping the CM parent also stops the CM children. If you kill a child CM, a new opcode call starts a new child CM. This is because the parent CM is still active, and can automatically start a child CM when you run an opcode.

In rare cases, you might be left with an allocated but unused shared memory block. Use the `ipcs` command to detect an allocated block; use the `ipcrm` command to remove it.

**Stopping a Base Set of BRM Components**

You can use the `pin_ctl stop all` command to stop a customizable set of components. By default, the components are stopped in this order:

- Invoice formatter
- CM Master Process
- Connection Manager
- Email DM
- Oracle DM

You can customize the components that the `pin_ctl stop all` command stops. See "Customizing the Components Included in 'all’". You can also create customized "all"
commands to stop a separate set of components. See "Creating a Custom ‘all’ Parameter List".

To stop a set of BRM components by using the pin_ctl utility:

1. Go to the BRM_Home/bin directory.
2. Run the pin_ctl utility with the stop action and all as the component:
   
   ```bash
   pin_ctl stop all
   ```

   **Note:** You can use the -collectdata parameter to collect diagnostic data about all of the components. See "Getting Diagnostic Data for a Component by Using the pin_ctl Utility".

---

**Starting and Stopping Pipeline Manager Manually**

You can stop and start Pipeline Manager by using the command line instead of by using the pin_ctl utility or HP OpenView.

---

**Important:** If you start Pipeline Manager manually, you cannot use the pin_ctl utility or HP OpenView to control Pipeline Manager; for example, to stop it or get the status.

---

**Starting Pipeline Manager**

You start an instance of Pipeline Manager by using the following command from the Pipeline_Home directory:

```bash
bin/ifw -r RegistryFile
```

where `RegistryFile` is the name of the registry file.

---

**Important:** If Pipeline Manager cannot establish a connection with the Pipeline Manager database (most likely because the database is down), you receive an error message and the Pipeline Manager startup is canceled.

---

The general syntax for the ifw command and parameters is:

```bash
ifw -r RegistryFile | -h | -v [-r RegistryFile]
```

where:

- `-r RegistryFile`
  Starts Pipeline Manager with the specified registry file.
- `-h`
  Displays the syntax and parameters.
- `-v [-r RegistryFile]`
  Displays the version of Pipeline Manager. If you use the `-r` parameter, it also displays the version and name of data and function modules. For example:

  ```bash
  ifw -v -r conf/wireless.reg
  ```
Pipeline Manager displays **Ready for processing** when startup procedures have completed.

### Stopping Pipeline Manager

**Important:** If you use HTTP or SNMP monitoring in a Pipeline Manager instance, you need to stop all monitoring requests to Pipeline Manager before you stop it. To stop the monitoring requests, stop the master SNMP agent. You can use the kill command, for example:

```plaintext
kill -9 master_agent_pid
```

You stop Pipeline Manager by using the following semaphore entry:

```plaintext
ifw.Active = FALSE
```

For information on semaphores and how to create semaphore files, see "Updating Configuration Settings During Runtime by Using Semaphore Files".

### Starting and Stopping Individual Pipelines

When you start Pipeline Manager, the Controller starts all pipelines. However, you can stop or restart individual pipelines by using semaphores.

**Important:** When a pipeline cannot establish a connection with the Pipeline Manager database (most likely because the database is down), you receive an error message and the pipeline startup is canceled.

To start an individual pipeline, use the following semaphore entry:

```plaintext
ifw.Pipelines.PipelineName.Active = True
```

where `PipelineName` is the name of the pipeline.

To stop an individual pipeline, use the following semaphore entry:

```plaintext
ifw.Pipelines.PipelineName.Active = False
```

where `PipelineName` is the name of the pipeline.

If files are added to the input directory after a pipeline is stopped and before it is restarted, the files are processed in order based on their last modified timestamp. For more information about input processing, see "Configuring EDR Input Processing" in BRM Configuring Pipeline Rating and Discounting.

For information on how to create semaphore files, see "Updating Configuration Settings During Runtime by Using Semaphore Files".

**Tip:** Pipeline Manager includes a set of Perl scripts, and associated semaphore files, that you can use to start and stop various types of pipelines and perform other system administration tasks. See "Using Perl Scripts to Start and Stop Pipeline Manager".
Restarting Pipeline Manager After an Abnormal Shutdown

Some modules track data in data files. In the event of an abnormal shutdown, you need to delete the data files that were in progress when the shut-down occurred.

See the following topics in *BRM Configuring Pipeline Rating and Discounting*:
- "Managing the Call Assembling Data Files"
- "Managing FCT_DuplicateCheck Data Files"

Using Perl Scripts to Start and Stop Pipeline Manager

Pipeline Manager includes a set of Perl scripts, and associated semaphore files, that you can use to start and stop various types of pipelines and perform other system administration tasks.

Table 1–1 describes the files and scripts used for starting and stopping pipelines:

<table>
<thead>
<tr>
<th>Semaphore and Perl script file names</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>start_all_pipeline.reg</td>
<td>Starts all pipelines configured in the <code>start_all_pipeline.reg</code> semaphore file. By default, this list includes these pipelines:</td>
</tr>
<tr>
<td>start_all_pipeline.pl</td>
<td>- ALL_RATE</td>
</tr>
<tr>
<td></td>
<td>- PRE_RECYCLE</td>
</tr>
<tr>
<td></td>
<td>- PRE_PROCESS</td>
</tr>
<tr>
<td></td>
<td>- ALL_BCKOUT</td>
</tr>
<tr>
<td></td>
<td>- ALL_RERATE</td>
</tr>
<tr>
<td></td>
<td>If you add custom pipelines, add their names to the semaphore file according to the default examples. Important:</td>
</tr>
<tr>
<td></td>
<td>- Do not run rating pipelines (ALL_RATE, PRE_RECYCLE, and PRE_PROCESS) at the same time that you run the rerating pipeline (ALL_RERATE). Edit the script to specify which pipeline to run.</td>
</tr>
<tr>
<td></td>
<td>- Before running ALL_RERATE, ensure that the backout pipeline (ALL_BCKOUT) has processed all EDRs that were extracted by the Event Extraction tool.</td>
</tr>
<tr>
<td>start_all_rate.reg</td>
<td>Starts the ALL_RATE pipeline.</td>
</tr>
<tr>
<td>start_all_rate.pl</td>
<td></td>
</tr>
<tr>
<td>start_DiscountPipeline.reg</td>
<td>Starts the discount pipeline (DiscountPipeline).</td>
</tr>
<tr>
<td>start_DiscountPipeline.pl</td>
<td></td>
</tr>
<tr>
<td>start_main_stop_rerating.reg</td>
<td>Starts these pipelines:</td>
</tr>
<tr>
<td>start_main_stop_rerating.pl</td>
<td>- PRE_PROCESS</td>
</tr>
<tr>
<td></td>
<td>- PRE_RECYCLE</td>
</tr>
<tr>
<td></td>
<td>- ALL_RATE</td>
</tr>
<tr>
<td></td>
<td>Stops these pipelines:</td>
</tr>
<tr>
<td></td>
<td>- ALL_BCKOUT</td>
</tr>
<tr>
<td></td>
<td>- ALL_RERATE</td>
</tr>
<tr>
<td>start_pre_process.reg</td>
<td>Starts the preprocessing pipeline (PRE_PROCESS).</td>
</tr>
<tr>
<td>start_pre_process.pl</td>
<td></td>
</tr>
</tbody>
</table>
Table 1–1 (Cont.) Scripts for Starting and Stopping Pipelines

<table>
<thead>
<tr>
<th>Semaphore and Perl script file names</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>start_pre_recycle.reg start_pre_recycle.pl</td>
<td>Starts the prerecycle pipeline (PRE_RECYCLE).</td>
</tr>
<tr>
<td>start_RealtimePipelineGPRS.reg start_RealtimePipelineGPRS.pl</td>
<td>Starts the real-time GPRS pipeline (RealtimePipelineGPRS).</td>
</tr>
<tr>
<td>start_RealtimePipelineGSM.reg start_RealtimePipelineGSM.pl</td>
<td>Starts the real-time GSM pipeline (RealtimePipelineGSM).</td>
</tr>
<tr>
<td>start_RealtimePipelineZone.reg start_RealtimePipelineZone.pl</td>
<td>Starts the real-time zoning pipeline (RealtimePipelineZone).</td>
</tr>
<tr>
<td>start_recycle.reg start_recycle.pl</td>
<td>Starts recycling rejected CDRs.</td>
</tr>
</tbody>
</table>
| start_rerating_stop_main.reg start_rerating_stop_main.pl | Stops these pipelines:  
  - ALL_RATE  
  - PRE_RECYCLE  
  - PRE_PROCESS  
  Starts these pipelines:  
  - ALL_BCKOUT  
  - ALL_RERATE |
| stop_all_pipeline.reg stop_all_pipeline.pl | Stops all the pipelines configured in the stop_all_pipeline.reg semaphore file. By default, this list includes these pipelines:  
  - ALL_RATE  
  - PRE_RECYCLE  
  - PRE_PROCESS  
  - ALL_BCKOUT  
  - ALL_RERATE  
If you add custom pipelines, add their names to the semaphore file according to the default examples. |
| stop_all_rate.reg stop_all_rate.pl | Stops the ALL_RATE pipeline. |
| stop_DiscountPipeline.reg stop_DiscountPipeline.pl | Stops the discount pipeline (DiscountPipeline). |
| stop_pre_process.reg stop_pre_process.pl | Stops the preprocessing pipeline (PRE_PROCESS). |
| stop_pre_recycle.reg stop_pre_recycle.pl | Stops the prerecycle pipeline (PRE_RECYCLE). |
| stop_RealtimePipelineGPRS.reg stop_RealtimePipelineGPRS.pl | Stops the real-time pipeline for GPRS (RealtimePipelineGPRS). |
| stop_RealtimePipelineGSM.reg stop_RealtimePipelineGSM.pl | Stops the real-time pipeline for GSM (RealtimePipelineGSM). |
| stop_RealtimePipelineZone.reg stop_RealtimePipelineZone.pl | Stops the real-time pipeline for zoning (RealtimePipelineZone). |
Starting and Stopping Oracle IMDB Cache DM

Before starting or stopping Oracle In-Memory Database (IMDB) Cache DM, ensure that the TIMESTEN_HOME environment variable is set to the directory in which you installed the Oracle IMDB Cache database.

Starting IMDB Cache DM

To start Oracle IMDB Cache DM:
1. Go to the BRM_home/bin directory.
2. Run the following command:
   
   pin_ctl start dm_tt

Stopping IMDB Cache DM

To stop Oracle IMDB Cache DM:
1. Go to the BRM_home/bin directory.
2. Run the following command:

   pin_ctl stop dm_tt
Monitoring and Maintaining Your BRM System

This chapter provides information and guidelines to help you manage the day-to-day operation of your Oracle Communications Billing and Revenue Management (BRM) system.

About Monitoring BRM

You use the following tools to monitor BRM components:

- HP OpenView with the BRM 7.4 Certified Smart Plug-in (the BRM_SPI_For_HPOpenView package). This BRM integration allows you to manage and control your distributed BRM system from a central graphical user interface. See "Using HP OpenView to Monitor and Control BRM".

- The pin_ctl utility. Use this utility to start and stop BRM components and to get diagnostic data. See "Using the pin_ctl Utility to Monitor BRM".

- The pin_db_alert.pl utility. Use this utility to monitor key performance indicators (KPIs), which are metrics you use to quantify the health of your database and to alert you to potential risks. See "Using the pin_db_alert Utility to Monitor Key Performance Indicators".

- Operations Management Framework (OMF) HTTP and SNMP protocols. These protocols provide access to data collected by OMF probes. See:
  - Using the SNMP Instrumentation Protocol to Monitor and Control BRM Components
  - Using the HTTP Instrumentation Protocol to Read OMF Instrumentation Data

In addition, you can use component-specific diagnostic tools such as:

- Pipeline Manager Diagnostic Data Handler. See "Using the Diagnostic Data Handler to Get OMF Diagnostic Data".

- Log files. See "Using Logs to Monitor Components".

- Connection Manager (CM) quality of service (QoS) statistics. See "Getting Quality of Service Statistics from the CM".

- Operating system commands. See:
  - Manually Checking the Status of the CM
  - Manually Checking the Status of the DM

Table 2–1 provides an overview of the system monitoring tools:
Components Monitored and Controlled by HP OpenView and the pin_ctl Utility

You can use HP OpenView and the pin_ctl utility to monitor the following BRM components:

- Connection Manager
- CM Master Process (CMMP)
- Connection Manager Proxy (cm_proxy)
- Data Managers:
  - Oracle Data Manager
  - Email Data Manager
  - EAI Data Manager
  - Paymentech Data Manager
  - Account Synchronization Data Manager
  - Invoice Data Manager
- EAI Java Server
- Invoice Formatter
- Paymentech Answer Simulator
- Pipeline Manager, including:
  - Real-time pipeline
  - Batch pipeline
  - AAA Gateway Manager
- Batch Controller

### Table 2–1  BRM System Monitoring Tools

<table>
<thead>
<tr>
<th>Monitoring Tool</th>
<th>Functions</th>
<th>Component</th>
</tr>
</thead>
<tbody>
<tr>
<td>HP OpenView</td>
<td>Monitor component status.</td>
<td>All BRM system components</td>
</tr>
<tr>
<td></td>
<td>Monitor log files for errors.</td>
<td>See &quot;Components Monitored and Controlled by HP OpenView and the pin_ctl Utility&quot;.</td>
</tr>
<tr>
<td></td>
<td>Stop and start components.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Get performance alerts.</td>
<td></td>
</tr>
<tr>
<td>pin_ctl utility</td>
<td>Stop and start components.</td>
<td>All system components</td>
</tr>
<tr>
<td></td>
<td>Get diagnostic data.</td>
<td>See &quot;Components Monitored and Controlled by HP OpenView and the pin_ctl Utility&quot;.</td>
</tr>
<tr>
<td></td>
<td>Clear log files.</td>
<td></td>
</tr>
<tr>
<td>pin_db_alert.pl utility</td>
<td>Monitor key performance indicators.</td>
<td>Oracle databases</td>
</tr>
<tr>
<td>Diagnostic Data Handler</td>
<td>Get application diagnostic data.</td>
<td>Pipeline Manager</td>
</tr>
<tr>
<td>Log files</td>
<td>Get status and error messages.</td>
<td>All system components</td>
</tr>
<tr>
<td>QoS statistics</td>
<td>Get QoS statistics.</td>
<td>CM</td>
</tr>
<tr>
<td>HTTP and SNMP system monitoring</td>
<td>Get instrumentation data from probes.</td>
<td>Pipeline Manager</td>
</tr>
<tr>
<td></td>
<td>Set configuration values.</td>
<td>AAA Gateway</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Real-time pipeline</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;BRM OMF Instrumented Objects&quot;.</td>
</tr>
</tbody>
</table>

You can use HP OpenView and the pin_ctl utility to monitor the following BRM components:
■ System Manager
■ Node Manager

About Data Collected by OMF
You can use OMF to get instrumentation data from Pipeline Manager. See "About Operations Management Framework" and "BRM OMF Instrumented Objects".

You can use two methods to get data:
■ SNMP. See "Using the SNMP Instrumentation Protocol to Monitor and Control BRM Components".
■ HTTP. See "Using the HTTP Instrumentation Protocol to Read OMF Instrumentation Data".

Using the pin_ctl Utility to Monitor BRM
You can perform the following monitoring tasks by using the pin_ctl utility:
■ Use the status command to get the current status of the component. See "Getting the Status of a Component by Using the pin_ctl Utility".
■ Use the clear command to delete log entries associated with the component (not the file). See "Clearing Log Files for a Component by Using the pin_ctl Utility".
■ Use the -collectdata parameter to get diagnostic data when starting, stopping, or checking the status of a component. See "Getting Diagnostic Data for a Component by Using the pin_ctl Utility".

You also use the pin_ctl utility to start, stop, halt, and restart system components. For information about starting and stopping BRM by using the pin_ctl utility, see "Starting and Stopping the BRM System".

For more information, see "pin_ctl".

Setting Up the pin_ctl Utility
Install the pin_ctl utility executable on any system that runs a BRM component.

Each instance of the pin_ctl utility is configured by a pin_ctl.conf file that contains data about the BRM components running on the system. See "Configuring the pin_ctl Utility".

Important: (AIX only) To start more than one pipeline process in the same machine, you must assign a different IFW_EVENTHANDLER port for each pipeline process. For example, if you are starting rtp, aaa, and bre in the same AIX machine:

```
rtp env_platform:common env_variable:IFW_EVENTHANDLER_PORT env_val:XXXX1
aaa env_platform:common env_variable:IFW_EVENTHANDLER_PORT env_val:XXXX2
bre env_platform:common env_variable:IFW_EVENTHANDLER_PORT env_val:XXXX3
```

To run the pin_ctl utility, set the PERL5LIB environment variable to point to the third-party application’s install directory. To do so, perform one of the following:
■ Add the following paths to the PERL5LIB environment variable for the root account on each managed node:
  - `BRM_Home/ThirdPartyApps/tools/PerlLib`
  - `BRM_Home/bin`

■ Before you deploy the `call_pin_ctl` script in `BRM_SPI_install_directory/bin`, add the following paths to the PERL5LIB variable in the script:
  - `BRM_Home/ThirdPartyApps/tools/PerlLib`
  - `BRM_Home/bin`

Getting the Status of a Component by Using the pin_ctl Utility

You can get the status of a component at any time.
To get the current status of a component by using the `pin_ctl` utility:
1. Go to the `BRM_Home/bin` directory.
2. Run the following command:
   ```bash
   pin_ctl status component
   ```
   where `component` is the component for which you want the status. For a list of valid component values, see "pin_ctl".
   For example:
   ```bash
   pin_ctl status dm_oracle
   ```
   You can use the `-collectdata` parameter to get diagnostic data when checking the status of a component:
   ```bash
   pin_ctl status -collectdata component
   ```
   See "Getting Diagnostic Data for a Component by Using the pin_ctl Utility".

Clearing Log Files for a Component by Using the pin_ctl Utility

To clear a component’s log file by using the `pin_ctl` utility:
1. Go to the `BRM_Home/bin` directory.
2. Run the following command:
   ```bash
   pin_ctl clear component
   ```
   where `component` is the component whose log file you want to clear. For a list of valid component values, see "pin_ctl".
   For example:
   ```bash
   pin_ctl clear dm_oracle
   ```
   You can also clear log files when you start a component by using the `cstart` command. See "Starting a Component and Clearing the Log File".

Getting Diagnostic Data for a Component by Using the pin_ctl Utility

You can use the `pin_ctl` utility to get diagnostic data about a component at the following times:
Before startup by using the `start` command. The data is collected before the component is started.

Before shutdown by using the `stop` command. The data is collected after the component is stopped.

While it is running when you use the `status` command.

When you run the `restart` and `cstart` commands.

**Note:** If you collect data during the `stop all` or `status all` commands, data is collected for all components before the command is carried out. For example, if you stop all components, data is collected about all the components, and then they are stopped.

The diagnostic data is written to a file in the component’s log directory. The file name is `component.diag.log` (for example, `cm.diag.log`).

When a new file is created, BRM renames the existing file to `component.diag.log.YYYYMMDDhhmmss` (for example, `dm_oracle.diag.log.20060918094046`).

To get diagnostic data about a component by using the `pin_ctl` utility:

1. Go to the `BRM_Home/bin` directory.
2. Run the following command:

   ```
   pin_ctl action -collectdata component
   ```

   where:
   - `action` specifies the action to be executed (for example, `start`), during which you want to collect diagnostic data.
   - `component` is the component for which you want diagnostic data. For a list of valid component values, see "pin_ctl".

   For example:

   ```
   pin_ctl start -collectdata dm_oracle
   ```

**Diagnostic Data Collected by the pin_ctl Utility**

- Date and time the data was collected.
- Extended information from the system (for example, system, node, release, and kernel ID).
- Environment variables for the current terminal session.
- System limits for the current terminal session.
- Memory information (for example, available memory).
- Storage device information (for example, available disk space).
- Patch level on the system.
- Kernel parameters.
- Network status showing all sockets, routing table entries, and interfaces.
- Network status summary.
- Inter-process communication facilities status.
Using the pin_ctl Utility to Monitor BRM

- NFS statistics.
- Duration of time that the system has been up.
- All active components.
- All active users.

Configuring the pin_ctl Utility

You can configure the pin_ctl utility by editing the pin_ctl.conf file. See the following topics:

- Customizing the Components Included in 'all'
- Customizing pin_ctl.conf for Starting and Stopping Optional Components
- Creating a Custom 'all' Parameter List
- Customizing the Components List
- Customizing the pin_ctl Startup Configuration
- Customizing the pin_ctl Utility Environment Variables
- Setting the pin_ctl Utility Log Level
- Configuring the Start and Stop Validation Settings for pin_ctl
- Customizing snmpset and snmpget Actions
- Using Custom pin_ctl Configuration Files

The pin_ctl.conf file is in BRM_Home/bin.

Customizing the Components Included in 'all'

You can customize the components that are included in the pin_ctl all component.

1. Open the pin_ctl.conf file in BRM_Home/bin.

2. Find the following lines in the file:

```
# List of services to be part of all [Optional].
#    Mention the service names separated by a space.
# '==' should be used to create an alias for 'all'.
#     For example, all=my_all
# all=my_all dm_oracle dm_email cm cmmp formatter
```

3. After all, enter each component that you want to start with the all command:

```
all component1 component2 component3 ...
```

   where componentX is the component you want to add. For a list of valid component values, see "pin_ctl".

   **Important:** Make sure the components are in the order in which you want them started. The order is reversed when the components are stopped.

4. Save and close the file.
Customizing pin_ctl.conf for Starting and Stopping Optional Components

The default pin_ctl.conf file is configured to start BRM system components only. To configure pin_ctl.conf to start an optional component, such as Synchronization Queue DM (dm_aq) or Taxware DM (dm_taxware), you must:

1. Open the pin_ctl.conf file in BRM_Home/bin.
2. Add the following line to the components list:

   

   ```
   start_sequence service_name [=alias_name]:java|:app|:pipeline|->dependent_service
   ```

   

   where:
   
   ■ `start_sequence` is the start and stop sequence number. This determines the order in which components are started or stopped.
   
   ■ `service_name` is the name of the optional component.
   
   ■ `=alias_name` indicates that `service_name` is different from the standard service name. For example:

   ```
   cm_1=cm
   cm_2=cm
   ```

   where `cm_1` and `cm_2` are `cm` services.
   
   ■ `:java` indicates that the component is Java-based.
   
   ■ `:app` indicates that the component executable is located in the BRM_Home/apps directory.
   
   ■ `:pipeline` identifies the component as pipeline.
   
   ■ `->dependent_service` specifies one or more components that `service_name` is dependent on. This indicates that `dependent_service` must start before `service_name` is started.

   For example, to add `dm_aq` to the components list:

   ```
   4 dm_aq
   ```

3. Add the following line to the startup configuration section of the file:

   ```
   start_component cpidproc:searchpattern:pidvarname cport:port_number [testnap:directory_name]
   ```

   where:
   
   ■ `start_component` is the name of the start command for the optional component, such as `start_dm_aq` or `start_dm_taxware`. It must be unique; if not, the last parsed definition is used.
   
   ■ `cpidproc:searchpattern` is a simple process name matching filter.
   
   ■ `pidvarname` is a partial match for the `pidfile` variable from `${program_name}`. If you enter nothing (which is recommended), the default is PIDS, which matches CMPID in $PIN_LOG/cm/cm.pid.
   
   ■ `cport:port_number` is the component port number.
   
   ■ `testnap:directory_name` runs the testnap utility in the specified directory. The directory is relative to BRM_Home/sys.

   For example, to enter a startup configuration for `dm_aq`:
start_dm_aq cpidproc:dm_aq: cport:--DM_AQ_PORT__

4. Save and close the file.

Creating a Custom 'all' Parameter List

You can create aliases for custom lists of components that are controlled by the pin_ctl utility all component. For example, if you define an alias named my_all, you can start a custom group of components by running:

pin_ctl start my_all

1. Open the pin_ctl.conf file in BRM_Home/bin.

2. Find the following lines in the file:

```plaintext
# List of services to be part of all [Optional].
#       Mention the service names separated by a space.
#       '=' should be used to create an alias for 'all'.
#       For example, all=my_all
# all=my_all dm_oracle dm_email cm cmmp formatter
```

3. Add the following line at the end of the section:

```plaintext
all=alias component1 component2 ...
```

where:

- **alias** specifies the name of your customized all command. For example, my_all.
- **componentX** is the component you want to add. For a list of valid component values, see "pin_ctl".

```
Important: Make sure the components are in the order in which you want them started. The order is reversed when the components are stopped by using the custom all command. Separate component names by using a space.
```

4. Save and close the file.

Customizing the Components List

The components list in the pin_ctl.conf file lists the BRM system components. For example:

```plaintext
1 dm_oracle
1 dm_email
1 dm_fusa
1 dm_invoice
...
4 rtp:pipeline
4 aaa:pipeline
4 bre:pipeline
```

If you have a high-availability system that includes duplicate instances of components, you can edit the pin_ctl.conf file to customize the components list. For example:

```plaintext
1 dmo1=dm_oracle
```
1 dmo2=dm_oracle
1 dm_eai_1=dm_eai
1 dm_eai_2=dm_eai
1 dm_ifw_sync_1=dm_ifw_sync
1 dm_ifw_sync_2=dm_ifw_sync
2 cm_1=cm->dm_oracle
2 cm_2=cm->dm_oracle
3 cm_proxy_1=cm_proxy
3 cm_proxy_2=cm_proxy
3 cmmp_1=cmmp
3 cmmp_2=cmmp
3 rtp_1=rtp:pipeline
3 rtp_2=rtp:pipeline
3 aaa_1=aaa:pipeline
34 aaa_2=aaa:pipeline

To customize the component list:
1. Open the pin_ctl.conf file in BRM_Home/bin.
2. Find the following lines in the file:
   
   # The format of entry for each service is ,
   # start_sequence service_name [=<alias_name>|:java|:app|-><list of services 
depends on>]
   #
   # The start sequence is a mandatory field, which gives sequence to start/stop
   # [Mandatory].
   # Sequence is a numerical value, and starts from 1. The service should be
   # specified
   # in the assending order based on the sequence number.
   # Mention the service name. This service_name is mandatory field [Mandatory].
   # NOTE: Start sequence and Service name should be separated by a space.
   #
   # '=' should be used if service name is different with standard service names
   #[Optional].
   # For example, cm2=cm
   # Here, cm2 is the service which is of cm category.
   # This is useful when multiple CMs/DMs are installed.
   # :app should be used if its located in BRM_Home/apps directory [Optional].
   # :java should be used if its a java based service [optional].
   # -> should be used if the current service has any dependencies [Optional].
   # This is generally useful in WINDOWS.
   # :pipeline should be used if it is Pipeline service [Optional].

3. Add the following line for each component in your system:

   start_sequence service_name [=alias_name|:java|:app|:pipeline|->dependent_ 
service]

   where:
   - start_sequence is the start/stop sequence number.
   - service_name is the component name.
   - =alias name indicates that service_name is different from the standard service 
   name. For example:

     cm_1=cm

     cm_2=cm

     where cm_1 and cm_2 are cm services.
Using the pin_ctl Utility to Monitor BRM

- \texttt{:java} indicates that the component is Java-based.
- \texttt{:app} indicates that the component executable is located in the \texttt{BRM\_Home/apps} directory.
- \texttt{:pipeline} identifies the component as pipeline.
- \texttt{->dependent\_service} specifies one or more components that \texttt{service\_name} is dependent on. This indicates that \texttt{dependent\_service} must start before \texttt{service\_name} is started.

4. Save and close the file.

Customizing the pin_ctl Startup Configuration

The \texttt{pin\_ctl.conf} file includes startup configurations for system components. For example:

\begin{verbatim}
start_cm     cpidproc:cm: cport:2224 testnap:test
\end{verbatim}

These configurations are created automatically during installation, but you can change them. For example, if you use a high-availability system with duplicate processes, you should change the component names. In the following example, the Oracle DM name in the component list is \texttt{dmo1}, so the startup configuration has been changed to match:

\begin{verbatim}
start_dmo1 cpidproc:dmo1: cport:12432
\end{verbatim}

1. Open the \texttt{pin\_ctl.conf} file in \texttt{BRM\_Home/bin}.
2. Edit the file.

   The syntax is:

   \begin{verbatim}
   start_component cpidproc:searchpattern:pidvarname cport:port_number
   \end{verbatim}
   \begin{verbatim}
   [testnap:directory_name]
   \end{verbatim}

   where:

   - \texttt{start\_component} is the name of the start command. It must be unique; if not, the last parsed definition is used.
   - \texttt{cpidproc:searchpattern} is a simple process name matching filter.
   - \texttt{pidvarname} is a partial match for the \texttt{pidfile} variable from \texttt{{$[program\_name]}}. If you enter nothing (which is recommended), the default is \texttt{PIDS}, which matches \texttt{CMPID} in \texttt{${PIN\_LOG/cm/cm.pid}$}.
   - \texttt{cport:port\_number} is the component port number. This value is entered automatically during installation.
   - \texttt{testnap:directory\_name} runs the \texttt{testnap} utility in the specified directory. The directory is relative to \texttt{BRM\_Home/sys}.

3. Save and close the file.

Customizing the pin_ctl Utility Environment Variables

Some BRM components need environment variables set before starting. You can edit the \texttt{pin\_ctl.conf} file to change the environment variables if yours are different from the default settings.

1. Open the \texttt{pin\_ctl.conf} file in \texttt{BRM\_Home/bin}.
2. To define environment variables for BRM components, find the following lines in the file:
3. Add the following line for each BRM component that requires an environment variable:

```
component env_platform:operating_system env_variable:environment_variable env_val:value
```

where:

- **component** is the BRM component that uses the environment variable (for example, `cm`). Use `common` to apply the environment variable to the entire system. For a list of component values, see "component Parameter".
- **operating_system** can be `hpux_ia64`, `linux`, `aix`, `solaris`, or `common`.
- **environment_variable** specifies the name of the environment variable to set before starting **component**.
- **value** specifies the environment variable value.

For example, the following line sets the NLS_LANG environment variable before starting any BRM component:

```
common env_platform:common env_variable:NLS_LANG env_val:AMERICAN_AMERICA.UTF8
```

4. To define environment variables for pipeline registry files, find the following lines:

```
# registry details for pipeline services
aaa env_platform:common env_variable:AAA_REGISTRY env_val:$IFW_HOME/conf/diameter_charge.reg
rtp env_platform:common env_variable:RTP_REGISTRY env_val:$IFW_HOME/conf/wirelessRealtime.reg
bre env_platform:common env_variable:BRE_REGISTRY env_val:$IFW_HOME/conf/wireless.reg
```

5. Add the following line for each pipeline component that uses a registry file:

```
component env_platform:common env_variable:registry_variable env_val:$IFW_HOME/registry_file
```

where:

- **component** is the pipeline component name. For a list of valid values, see "component Parameter".
- **registry_variable** is the environment variable to set before starting **component**.
  
  The syntax for pipeline registry environment variables is `_REGISTRY`. For example, use `AAA_REGISTRY` to specify the AAA Gateway registry file.

- **registry_file** is the path and file name for the pipeline registry file.

For example:

```
aaa env_platform:common env_variable:AAA_REGISTRY env_val:$IFW_HOME/conf/diameter_charge.reg
```

6. Save and close the file.
Setting the CTL_SNMP_PATH variable
You can set the CTL_SNMP_PATH variable to one of the following:

- \textit{BRM\_Home/bin}. For example:
  
  \begin{verbatim}
  common env_platform:common env_variable:CTL_SNMP_PATH env_val:BRM\_Home/bin
  \end{verbatim}

- The path of the SNMP third-party software. For example:
  
  \begin{verbatim}
  common env_platform:common env_variable:CTL_SNMP_PATH env_val:/home2/mydir/opt/snmp/bin
  \end{verbatim}

Setting the pin\_ctl Utility Log Level
To set the log level for the pin\_ctl utility:

1. Open the pin\_ctl.conf file in \textit{BRM\_Home/bin}.
2. Edit the file.
   
The syntax is:
   
   \begin{verbatim}
   Control_script_log loglevel:level logfile:log_file
   \end{verbatim}

   where:

   - \textit{level} is the log level, which can be:
     
     \begin{itemize}
     \item \textbf{none}: no logging
     \item \textbf{error}: log error messages only (default)
     \item \textbf{warning}: log error messages and warnings
     \item \textbf{debug}: log error messages, warnings, and debugging messages
     \end{itemize}

   - \textit{log\_file} is the name of the log file.

   For example:
   
   \begin{verbatim}
   Control_script_log loglevel: error logfile: pin\_ctl.log
   \end{verbatim}

   \textbf{Note: } Instead of always getting debugging information, you can use the \texttt{pin\_ctl -debug} parameter to get debugging information whenever you run the \texttt{pin\_ctl} utility. For example:
   
   \begin{verbatim}
   pin\_ctl -debug start dm\_oracle
   \end{verbatim}

3. Save and close the file.

Configuring the Start and Stop Validation Settings for pin\_ctl
You can configure the validations \texttt{pin\_ctl} performs when starting and stopping components, including:

- How long the utility waits before checking whether an action is complete.
- The maximum number of times the utility checks whether an action is complete.
- The home directory for the specified component:
  
  - For BRM processes, this overrides the \textit{BRM\_Home} value for the specified component.
  
  - For pipeline processes, this overrides the \textit{IFW\_Home} value for the specified component. This is used as the relative path for entries in the registry file. For
example, if a registry entry specifies \temp, the pipeline process uses IFW_Home/temp.

- The home log directory for the specified component. This overrides the $PIN_LOG value for the specified component.

To specify the validation settings used when pin_ctl starts and stops components:

1. Open the pin_ctl.conf file in BRM_Home/bin.
2. Find the following lines in the file:

```plaintext
# This sections will be used to have different settings for each service like
# 1. waittime -- number of seconds to be waited
# 2. iterations -- Number of times to be checked
# 3. pin_Home_dir -- BRM_Home path
# 4. pin_log_dir -- PIN_LOG path
# All these are optional, if these are not set then default values will be used.
```

3. Add the following line for each component that you want to override the default values:

```plaintext
settings component waittime:wait iterations:value pin_Home_dir:path pin_log_dir:logpath
```

where:

- **component** is the BRM component. For a list of valid values, see "component Parameter".
- **wait** is the number of seconds to wait before checking whether an action is complete. The default is 5.
- **value** is the maximum number of times to check whether an action is complete. The default is 5.
- **path** is the home directory. This overrides the BRM_Home value for BRM processes and the IFW_Home value for pipeline processes.
- **logpath** is the BRM log file home. The default is the value set in the $PIN_LOG environment variable. You must change this only if you use a different directory than the default directory.

For example:

```plaintext
settings dm_oracle waittime:5 iterations:5 pin_Home_dir:BRM_Home pin_log_dir:$PIN_LOG
```

4. Save and close the file.

**Customizing snmpset and snmpget Actions**

You can run `snmpset` and `snmpget` commands from the pin_ctl utility. You can edit the pin_ctl.conf file to add `snmpset` and `snmpget` actions.

1. Open the pin_ctl.conf file in BRM_Home/bin.
2. Edit the file.

   The syntax is:

   ```plaintext
   snmp_command servicename probe registry_entry base_OID
   ```

   where:
Using the pin_db_alert Utility to Monitor Key Performance Indicators

- `snmp_command` is either `snmpset` or `snmpget`.
- `servicename` is the name of the component. Use the names defined for the `pin_ctl` utility. See "pin_ctl".
- `probe` is the name of the probe that receives the `snmpset` command. For information about probes, see "BRM OMF Instrumented Objects".
- `registry_entry` is the registry entry that corresponds to the probe.
- `base_OID` is the base process ID (OID) from the BRM Management Information Base (MIB) file. See "BRM OMF Instrumented Objects".

3. Save and close the file.

   You can now use the `pin_ctl` utility `snmpset` or `snmpget` action by using the probe name. For example:

   ```bash
   pin_ctl snmpset probe_name component
   ```

Using Custom pin_ctl Configuration Files

You can create custom `pin_ctl` configuration files to run different configurations of the same system.

1. Create a custom configuration file in `BRM_Home/bin`. You can copy and rename the `pin_ctl.conf` file.

2. Use the `-c file_name` parameter when you run the `pin_ctl` utility. For example:

   ```bash
   pin_ctl cstart all -c pin_ctl_batch.conf
   ```

Using the pin_db_alert Utility to Monitor Key Performance Indicators

KPIs are metrics you use to quantify the health of your database and to alert you when potential issues exist. They identify database tables that must be archived or purged and indexes, triggers, and stored procedures that are missing or invalid.

KPIs are monitored when you run the `pin_db_alert.pl` utility. Generally you set up a `cron` job to run the utility periodically to monitor the health of your database. For more information, see "Running the pin_db_alert.pl Utility".

Each KPI is identified by an ID that associates a component being monitored to a corresponding validation value. For example, you can monitor the size of an audit table with a size threshold that monitors the number of rows in that audit table. When the threshold value is reached, the results are returned and an alert notification can be sent, warning you of the component’s condition.

The component and validation functionality for each KPI comprises:

- A data extraction module, which queries the database for the KPI data and writes the results to an output file.
- A validation module, which compares the query results to validation parameters defined in a configuration file and writes the validation status to an output file.

After the validation results are written to the output files, a decision module (`DecisionUtility.pm`) evaluates each KPI result and determines whether to generate email alert notifications based on the KPI result status. For more information, see "About KPI Status and Alert Notifications".
KPI Default Behavior

Table 2–2 contains a list of supported KPIs and provides the default behavior of their data and validation modules.

Table 2–2  Supported KPIs and Default Behavior

<table>
<thead>
<tr>
<th>KPI ID</th>
<th>Default Behavior</th>
</tr>
</thead>
</table>
| AuditHistoryAge | The *auditAge* module calculates the age of the audit tables listed in the *pin_db_alert.conf* file’s DATA_PLUGINS entry and DEFAULT_AUDIT_TABLES entry. It writes the results to the *auditAge_AuditHistoryAge.out* file.  
The *auditAge_validation* module uses threshold values in the *auditAge* validation configuration file to determine which audit tables in the results file are at the threshold, and writes them to the *auditAge_validation_AuditHistoryAge.out* file.  
For information on changing the default age thresholds, see “Monitoring the Age of Audit Tables”. |
| AuditTableSize | The *auditSize* module calculates the number of rows present in the audit tables listed in the *pin_db_alert.conf* file’s DATA_PLUGINS entry and DEFAULT_AUDIT_TABLES entry and writes the results to the *auditSize_AuditTableSize.out* file.  
The *auditSize_validation* module uses the threshold values in the *auditSize* validation configuration file to determine which audit tables in the results file are at the threshold and writes them to the *auditSize_validation_AuditTableSize.out* file.  
For information on changing the default size thresholds, see “Monitoring the Size of Audit Tables”. |
| OldestEventAge | The *eventData* module calculates the age of the oldest event in the *event_t* table, as well as the records in the tables defined in the *pin_db_alert.conf* file’s DATA_PLUGINS entry, and writes the results to the *eventData_OldestEventAge.out* file.  
The *eventData_validation* module uses the threshold values in the *eventData* validation configuration file to determine which entries in the results file are at the threshold, and writes them to the *eventData_validation_OldestEventAge.out* file.  
For information on changing the default event age, see “Monitoring the Age of Events”. |
### About KPI Status and Alert Notifications

When the `pin_db_alert.pl` utility runs, it returns a PASS or FAILURE status for each configured KPI, which includes a severity level for the status. The following severity levels listed in Table 2–3 are possible for any KPI:

<table>
<thead>
<tr>
<th>KPI ID</th>
<th>Default Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTIVETRIGGERS</td>
<td>The <code>triggersList</code> module retrieves a list of active triggers in the BRM system and writes their names and status (ENABLED or DISABLED) to the <code>triggersList_ACTIVETRIGGERS.out</code> file. The <code>triggersList_validation</code> module compares the list of active triggers in the <code>triggersList</code> validation configuration file to the triggers in the results file and writes missing triggers to the <code>triggersList_validation_ACTIVETRIGGERS.out</code> file. <strong>Important:</strong> If you installed optional managers that use unique triggers or if you created custom triggers, you must add them to the <code>triggersList</code> validation configuration file to monitor their status. See &quot;Monitoring Active Triggers&quot;.</td>
</tr>
<tr>
<td>INDEXES</td>
<td>The <code>indexList</code> module retrieves a list of unique indexes in the BRM system and writes the index names and uniqueness values to the <code>indexList_INDEXES.out</code> file. The table name and column name for each index is also listed. The <code>indexList_validation</code> module compares the list of indexes in the <code>indexList</code> validation configuration file to the indexes in the results file and writes missing or invalid indexes to the <code>indexList_validation_INDEXES.out</code> file. <strong>Important:</strong> If you installed optional managers that use unique indexes or if you created custom indexes, you must add them to the <code>indexList</code> validation configuration file to monitor their status. See &quot;Monitoring Indexes&quot;.</td>
</tr>
<tr>
<td>PROCEDURES</td>
<td>The <code>proceduresList</code> module retrieves a list of stored procedures in the BRM system and writes the stored procedure names and status (VALID or INVALID) to the <code>proceduresList_PROCEDURES.out</code> file. This enables Pipeline Manager to compile data in parallel and to restore it from the precompiled data files file. The <code>proceduresList_validation</code> module compares the list of stored procedures in the <code>proceduresList</code> validation configuration file to the procedures in the results file and writes missing procedures to the <code>proceduresList_validation_PROCEDURES.out</code> file. <strong>Important:</strong> If you installed optional managers that use unique stored procedures or if you created custom stored procedures, you must add them to the <code>proceduresList</code> validation configuration file to monitor their status. See &quot;Monitoring Stored Procedures&quot;.</td>
</tr>
</tbody>
</table>

You can enable email alerts to notify a list of people about the validation results. For more information, see "Setting Up Email Alert Notifications".
You can configure the `pin_db_alert.pl` utility to send email notifications to alert a list of people when a KPI is at a specified severity level. For more information, see "Setting Up Email Alert Notifications".

### About Monitoring KPIs

To monitor KPIs, first you configure the KPI data entries in the `pin_db_alert.pl` utility’s configuration file, and then you set up the validation thresholds in each validation module’s configuration file.

The `pin_db_alert.pl` utility’s configuration file contains entries for all KPIs; therefore, Oracle recommends that you configure this file for all KPIs before you set up the validation thresholds for each individual KPI.

**Important:** If you do not define KPI validation thresholds, the validation process will not occur; therefore, any alert notifications you configured will not be sent.

For more information, see "Setting Up KPI Monitoring".

### Setting Up KPI Monitoring

The default configuration for monitoring KPIs is defined in the `pin_db_alert.pl` utility’s configuration file (`BRM_Home/diagnostics/pin_db_alert/pin_db_alert.conf`).
To edit this file, open it with a text editor and perform the following tasks as necessary. For more information, see the comments in the `pin_db_alert.conf` file.

- In the KPI_IDS entry, specify the KPI ID for each KPI to monitor.
  
  By default, all KPIs are listed; therefore, if you do not want to monitor one, remove it from the default list. For a list of KPI IDs, see "KPI Default Behavior".

- In the DATA_PlUGINS entry, specify the data module and desired values for each KPI listed in the KPI_IDS entry. See "Specifying Which KPI Data Is Extracted".

---

**Important:** In the sample `pin_db_alert.conf` file, values are provided for the AuditHistoryAge and AuditTableSize KPIs; however, the OldestEventAge KPI does not contain any values. You must provide your own values. See "Monitoring the Age of Events".

---

- In the VALIDATION_PlUGINS entry, specify the validation module for each KPI listed in the KPI_IDS entry.

---

**Important:** Make sure the validation modules are listed in the same order as their associated data modules in the DATA_PlUGINS entry.

---

- In the STATUS entry, configure the alert notifications. Specify the status and severity, and list the email addresses that get notified by the status/severity combination. For more information, see "Setting Up Email Alert Notifications".

- In the DEFAULT_AUDIT_TABLES entry, specify which audit tables to monitor by default. These audit tables are monitored in addition to any tables you list as values in the DATA_PlUGINS entry for the auditAge and auditSize modules.

- In the DB_USER and DB_PASSWD entries, specify the database user ID and encrypted password that are listed in the sm_id and sm_pw entries in the Data Manager (DM) `pin.conf` file. For more information, see "Enabling Database Access".

### Specifying Which KPI Data Is Extracted

To specify which data is extracted from the database during KPI monitoring:

1. Open the `pin_db_alert.pl` utility’s configuration file (`BRM_Home/diagnostics/pin_db_alert/pin_db_alert.conf`) with a text editor.

2. In the DATA_PLUGINs entry, specify the data module and desired values for each KPI in the KPI_IDS entry:

   - **To extract data for the auditAge data module:**

     Specify the audit table names to monitor using the following syntax, separating each audit table name by a space:

     ```
     @DATA_PLUGINs =("auditAge Audit_table_name Audit_table_name");
     ```

     **Note:** These tables are in addition to audit tables you have listed in the DEFAULT_AUDIT_TABLES entry.

   - **To extract data for the auditSize data module:**
Specify the audit table names to monitor using the following syntax, separating each audit table name by a space.

\[@DATA\_PLUGINS = \(\text{"auditSize Audit_table_name Audit_table_name\text{"}}\);\]

**Note:** These tables are in addition to audit tables you have listed in the \text{DEFAULT\_AUDIT\_TABLES} entry.

- **To extract data for the eventData module:**

  Specify the events to monitor using the following syntax:

  \[@DATA\_PLUGINS = \(\text{"eventData Table_name:Column_name:Operator:Column_value\text{"}}\);\]

  where:
  - \text{Table_name} is the name of the table that contains the event data.
  - \text{Column_name} is the name of the table column that contains the event data.
  - \text{Operator} is any standard SQL operator.
  - \text{Column_value} is the POID of the event.

  For example:

  \[@DATA\_PLUGINS = \(\text{"eventData event_t:account_obj_id0:=:21950\text{"}}\);\]

  **Note:** You can add any number of values for the \text{eventData} module, separated by spaces; however, you can specify only one operator per table. If the operator or syntax is incorrect, the table is not validated, and an error is written to the data extraction output file.

- **To extract data for the triggersList, proceduresList, and indexList modules:**

  The \text{triggersList}, \text{proceduresList}, and \text{indexList} modules take no values. To extract data for these modules, list them in the \text{DATA\_PLUGINS} entry using the following syntax:

  \[@DATA\_PLUGINS = \(\text{"triggersList","proceduresList","indexList"}\);\]

  Enclose the entire \text{DATA\_PLUGINS} value string with parentheses ( ) and separate each data value string with commas. For example:

  \[@DATA\_PLUGINS = \(\text{"auditSize au_service_t au_product_t au_account_t au_rate_t","eventData event_t:account_obj_id0:=:21956 account_t:poid_id0:=:21956","auditAge au_service_t au_product_t","triggersList","proceduresList","indexList"}\);\]

3. Save and close the file.

**Setting Up Email Alert Notifications**

To configure the \text{pin\_db\_alert.pl} utility to send email notifications when a KPI validation returns a specified result/severity combination:

1. Open the \text{pin\_db\_alert.pl} utility’s configuration file (\text{BRM\_Home/diagnostics/pin\_db\_alert.conf}) with a text editor.

2. Edit the \text{STATUS} entry using the following syntax:
Using the pin_db_alert Utility to Monitor Key Performance Indicators

'Error:MAIL_ALERT:Notification_list'

where:

■  *Error* is a combination of the status and severity, separated by a dot (.). The following values are valid:

FAIL.CRITICAL
FAIL.MAJOR
FAIL.MINOR
FAIL.WARNING
PASS.WARNING
PASS.NORMAL

■  *Notification_list* is a comma-separated list of email addresses to which the validation results are sent. You can have any number of email addresses for any error.

Be sure to enclose each status string in single quotation marks (').

For example:

@STATUS=('FAIL.CRITICAL:MAIL_ALERT:IT@example.com', 'FAIL.MINOR:MAIL_ALERT:john_smith@example.com, sysadm@example.com');

---

**Note:** You cannot configure email alerts for a specific KPI.

---

3. Save and close the file.

**Enabling Database Access**

The pin_db_alert.pl utility requires the database user name and password to query the database for KPIs.

1. Open the pin_db_alert.pl utility’s configuration file (`BRM_Home/diagnostics/pin_db_alert/pin_db_alert.conf`).

2. In the DB_USER and DB_PASSWD entries, specify the database user ID and encrypted password, respectively.

---

**Important:** These must be the same database user ID and password specified in the `sm_id` and `sm_pw` entries in the DM `pin.conf` file.

---

Use the following syntax:

```
DB_USER="User_ID";
DB_PASSWD="Encrypted_passwd";
```

For example:

```
DB_USER="brm123";
DB_PASSWD="&aes|0D5E11BFDD97D2769D9B0DB8BD1BBF7EE03F1642861DFA57502C7FB85A654267";
```

3. Save and close the file.
Monitoring the Size of Audit Tables
To monitor the size of audit tables:

1. If necessary, specify the `auditSize` module values in the DATA_PLUGINS entry of the `pin_db_alert.pl` utility’s configuration file. See "Setting Up KPI Monitoring".

2. Open the `auditSize` validation configuration file (`BRM_Home/diagnostics/pin_db_alert/auditSize_validation_AuditTableSize.conf`) with a text editor.
   - To change a size threshold for an existing table, change the number of rows specified in the AUDIT_SIZE_THRESHOLD value for that table.
   - To add an audit table, add a new AUDIT_SIZE_THRESHOLD entry for that table.
   - To omit an audit table from the validation process, either delete the AUDIT_SIZE_THRESHOLD entry for that table or comment out the entry.

For details on how to configure the AUDIT_SIZE_THRESHOLD entry, see the comments in the `AuditTableSize` configuration file.

3. Save the file.

Monitoring the Age of Audit Tables
To monitor the age of audit tables:

1. If necessary, specify the `auditAge` module values in the DATA_PLUGINS entry of the `pin_db_alert.pl` utility’s configuration file. See "Setting Up KPI Monitoring".

2. Open the `auditAge` validation configuration file (`BRM_Home/diagnostics/pin_db_alert/auditAge_validation_AuditHistoryAge.conf`) with a text editor.
   - To change an age threshold for a table, change the number of days specified in the AUDIT_AGE_THRESHOLD value for that table.
   - To add an audit table, add a new AUDIT_AGE_THRESHOLD entry.
   - To omit an audit table from the validation process, either delete the AUDIT_AGE_THRESHOLD entry for that table or comment out the entry.

For details on how to configure the AUDIT_AGE_THRESHOLD entry, see the comments in the `AuditHistoryAge` configuration file.

3. Save the file.

Monitoring the Age of Events
To monitor the age of events:

1. If necessary, configure the `eventData` module values in the DATA_PLUGINS entry of the `pin_db_alert.pl` configuration file (`BRM_Home/diagnostics/pin_db_alert.conf`). See "Specifying Which KPI Data Is Extracted".

---

**Note:** You can add any number of arguments for the `eventData` module; however, you can specify only one operator per table. If the operator or syntax is incorrect, the table is not validated, and an error is written to the data extraction output file.
2. Open the `eventData` validation configuration file (`BRM_Home/diagnostics/pin_db_alert/eventData_validation_OldestEventAge.conf`) with a text editor.
   - To change an age threshold, change the number of days specified in the `OLDEST_THRESHOLD` value for the table.
   - To add a table to monitor, add a new `OLDEST_THRESHOLD` entry for the table.
   - To omit a table from the validation process, either delete the `OLDEST_THRESHOLD` entry for that table or comment it out.

For details on how to configure the `OLDEST_THRESHOLD` entry, see the comments in the `OldestEventAge` configuration file.

3. Save the file.

**Monitoring Active Triggers**
To monitor a trigger for an optional manager or customization that is not part of BRM:

1. If necessary, specify the `triggersList` module in the `DATA_PLUGINS` entry in the `pin_db_alert.pl` utility’s configuration file. See "Setting Up KPI Monitoring".

2. Open the `ACTIVETRIGGERS` validation configuration file (`BRM_Home/diagnostics/pin_db_alert/triggersList_validation_ACTIVETRIGGERS.conf`) with a text editor.

3. Add a new entry for the trigger using the following syntax:
   ```
   ENABLED trigger_name
   ```

4. Save the file.

5. Restart the Connection Manager (CM).

**Monitoring Indexes**
To monitor an index for an optional manager or customization that is not part of BRM:

1. If necessary, specify the `indexList` module in the `DATA_PLUGINS` entry in the `pin_db_alert.pl` utility’s configuration file. See "Setting Up KPI Monitoring".

2. Open the `BRM_Home/diagnostics/pin_db_alert/indexList_validation_INDEXES.conf` file.

3. Add a new entry for the index using the following syntax:
   ```
   table_name  column_name  index_name  UNIQUE
   ```

   **Note:** To add a composite index, add each column name as a separate entry, in the order of the columns in the index. For example:
   ```
   ACCOUNT_NAMEINFO_T  OBJ_ID  I_ACCOUNT_NAMEINFO__I  UNIQUE
   ACCOUNT_NAMEINFO_T  REC_ID  I_ACCOUNT_NAMEINFO__I  UNIQUE
   ACCOUNT_T  ACCOUNT_NO  I_ACCOUNT_NO__ID  UNIQUE
   ```

4. Save the file.

**Monitoring Stored Procedures**
To monitor a stored procedure for an optional manager or customization that is not part of BRM:
1. If necessary, specify the proceduresList module in the DATA_PLUGINS entry in the pin_db_alert.pl utility's configuration file. See "Setting Up KPI Monitoring".

2. Open the PROCEDURES validation configuration file (BRM_Home/diagnostics/pin_db_alert/proceduresList_validation_PROCEDURES.conf file) with a text editor.

3. Add a new entry for the stored procedure using the following syntax:

   procedure_name VALID

4. Save the file.

**Running the pin_db_alert.pl Utility**

Run the pin_db_alert.pl utility periodically to monitor the health of your database. The cron command is the typical way to do this.

---

**Note:** You can also run the pin_db_alert.pl utility manually at the command line (for example, after system upgrades).

---

Use a cron job with a crontab entry to run the pin_db_alert.pl utility at a specified time. The following crontab entry runs the utility at 1:00 a.m. on a quarterly basis:

```
0 1 * */3 * BRM_Home/bin/pin_db_alert.pl &
```

**Defining Custom KPIs**

You can define custom KPIs (for example, to monitor the integrity of customer subscriber information after system upgrades):

- Define a new KPI called SubscriberInformation to monitor the consistency of subscriber data over a period of time. This KPI must include a data module that retrieves the subscriber information and a validation module that verifies this data.

- Create a configuration file for the KPI validation module and specify the relevant threshold information.

- Add the new KPI information to the pin_db_alert.conf file. For information on the entries in this file, see "Setting Up KPI Monitoring".

**Collecting Diagnostic Information by Using RDA**

Remote Diagnostic Agent (RDA) is an Oracle standard tool used to collect diagnostic data from your system applications environment.

---

**Note:** RDA replaces the Support Informer utility. Support Informer is obsolete and no longer supported. However, Support Informer libraries continue to be packaged with BRM. The libraries are accessed by the RDA profile named SupportInformer74 at run time.

---

Use RDA to collect information about your BRM system. When you submit a service request (SR) to Oracle Technical Support, you must also provide an RDA output file. The RDA output file provides a comprehensive view of your system configuration and
Collecting Diagnostic Information by Using RDA

contains diagnostic data used by Oracle Technical Support to diagnose problems. This minimizes the number of requests from Oracle Technical Support for additional information, which can reduce the service request resolution time.

You can use RDA to collect BRM and Pipeline Manager diagnostic information. The information collected from BRM includes:

- **Component log files**
  - RDA collects component log data from the component `.pinlog`, `.log`, and `Infranet.properties` files. For example, RDA collects the log data for BRM invoice formatter from `formatter.pinlog`, `formatter.log`, and `Infranet.properties`.

- **Application log files**
  - RDA collects application log data from the application `.pinlog`, `.log`, and `Infranet.properties` files. For example, RDA collects the log data for Batch Controller from `batch_controller.pinlog`, `BatchController.log`, and `Infranet.properties`.

- **Configuration files**
  - RDA collects configuration data from the `pin.conf` file. For example, RDA collects CMMP configuration data from the CMMP `pin.conf` file.

- **Other files**
  - RDA collects installation and version details from the `vpd.properties` and `pinrev.dat` files.

The information collected from Pipeline Manager includes:

- **Configuration files**
  - RDA collects the pipeline configuration data from the `.reg` (registry) and `.dsc` (description) files. For example, RDA collects the configuration data for wireless from the `wireless.reg` and `containerDesc.dsc` files.

- **Log files**
  - RDA collects pipeline log data from the process log, pipeline log, and stream log files. For example, RDA collects the log data for wireless from the `processWIRELESS.log` file, the `log_streamRT1.log` file, etc.

- **Other files**
  - RDA collects pipeline installation and version details from the `vpd.properties` and `piperev.dat` files.

To find BRM component information, RDA looks in the following directories:

- **BRM_Home/sys**
- **BRM_Home/apps**

To find Pipeline Manager information, RDA looks at the registry files.

A complete overview of RDA is provided in the Remote Diagnostic Agent (RDA) 4 - Getting Started document. See “Viewing RDA Documentation”.

RDA 4.21 collects the following customer-specific information:

- Company name
- Contact person
- Contact email
Remote Diagnostic Agent (RDA) 4.21 is included in BRM 7.4 Patch Set 7 and supported by subsequent releases.

**Caution:** When you run `rda.sh`, the script returns the "Perl not found in the PATH" error and the command fails. To work around this issue, remove the `.config` file (hidden file) in the RDA directory. Oracle recommends that you do not use shell script for RDA 4.21.

If you are using a version of the Third-Party software package that is prior to the version provided by BRM 7.4 Patch Set 7, you can do any one of the following:

- Uninstall the previous installation and install the BRM Patch Set 7 Third-Party software installation package.
- Overwrite the previous installation with the BRM Patch Set 7 Third-Party software installation package. In this case, both RDA 4.15 and RDA 4.21 will be available.

### Installing Remote Diagnostic Agent

RDA is included in the Third-Party package along with Perl and Java Runtime Environment. It automatically gets installed when you install the Third-Party package, in the directory you choose to install the Third-Party software. For more information, see "Installing the Third-Party Software" in *BRM Installation Guide*.

**Note:**

- RDA is not supported on Windows.
- RDA collects diagnostic and configuration data for all BRM and Pipeline Manager components and applications only from the server on which RDA is running. To collect data for BRM or Pipeline Manager components and databases on other servers, install and run RDA on the other servers.

To determine whether RDA is installed on a server, run the following command:

```bash
>perl rda.pl -cv
```

If RDA is installed on the server without any error, the following message is displayed: "No issues found in the RDA installation."

RDA includes a profile named `SupportInformer74`, which runs the following modules:

- **S380BRM**
  Collects Oracle Communications BRM information.
- **S105PROF**
  Collects the user profile data.
- **S110PERF**
  Collects performance information.
■ **S100OS**

Collects operating system information.

---

**Note:** In addition to the preceding modules, the RDA **SupportInformer74** profile runs other modules, such as INI, CFG, END, RDSP, and LOAD.

---

**Running Remote Diagnostic Agent**

To run RDA:

1. Go to the directory where you installed the Third-Party package and source the `source.me` file:
   
   Bash shell:
   ```bash
   source source.me.sh
   ```
   
   C shell:
   ```csh
   source source.me.csh
   ```

2. To collect BRM system information, verify that the PIN_HOME environment variable is set to the BRM installation directory. By default, it is `/opt/portal`.

3. To collect pipeline log files, verify that the INT_HOME environment variable is set to the Pipeline Manager installation directory. By default, it is `/opt/ifw`.

4. To run RDA, you must first perform an initial setup and then run data collection. To perform the initial setup, run the following command:
   ```perl
   perl rda.pl -S
   ```

5. Run one or more of the following commands:

   ■ To identify the list of modules:
     ```perl
     perl rda.pl -L m
     ```

   ■ To identify the list of profiles:
     ```perl
     perl rda.pl -L p
     ```

   ■ To identify the list of modules for the available profiles:
     ```perl
     perl rda.pl -x profiles
     ```

   ■ To get online documentation about the BRM module:
     ```perl
     perl rda.pl -M BRMr7.def
     ```

   ■ To perform BRM data collection using default values:
     ```perl
     perl rda.pl -v
     ```

***Important:** To collect database-specific data, you must run the command as a SYSDBA because DBA privileges are required to collect the database tables data.
When you run RDA, it prompts for information to determine what data to collect and for which products. You can choose to accept the default values or change them based on your BRM and Pipeline Manager installations and system configuration. RDA saves all your responses to the `/rda/setup.cfg` file.

**Note:** You can change the default location of `setup.cfg` file, if required.

For example, to initialize data collection and to generate the output files, RDA prompts for the following setup information:

**S000INI:** Initializes the Data Collection

Enter the prefix for all the files generated
Hit 'Return' to accept the default (RDA)

Enter the directory used for all the files generated
Hit 'Return' to accept the default (/rda/output)

Do you want to keep report packages from previous runs (Y/N)?
Hit 'Return' to accept the default (N)

Is a fresh collection done as default action (Y/N)?
Hit 'Return' to accept the default (Y)

Enter the Oracle Home to be used for data analysis
Hit 'Return' to accept the default

Enter the domain name of this server
Hit 'Return' to accept the default (portal.com)

If your database is running on the same server as RDA, RDA prompts for the following database information:

**S200DB:** Controls RDBMS Data Collection

Enter the Oracle SID to be analyzed
Hit 'Return' to accept the default (PortalDB)

Enter the location of the spfile or the INIT.ORA (including the directory and file name)

To collect BRM and Pipeline Manager system information, RDA prompts for the following BRM information:

**S380BRM:** Collects BRM Software Information

Should RDA collect BRM Software information (Y/N)?
Hit 'Return' to accept the default (Y)

Should RDA collect BRM based system information (Y/N)?
Hit 'Return' to accept the default (Y)

Enter a pipeline registry file to analyze or . to terminate the list
Collecting Diagnostic Information by Using RDA

Prompts are displayed that apply to other Oracle products. For these cases, choose the default value.

You can also run RDA in non-interactive mode by using command-line options:

Syntax:
```
    rda.pl -v -d -S -C -R -P -p profile_name [-db_version]
```

- `-v`: Set verbose mode
- `-d`: Set debug mode
- `-S`: Set up specified modules
- `-C`: Collect diagnostic information
- `-R`: Generate specified reports
- `-P`: Package the reports
- `-p profile_name [-db_version]`: Specify the setup profile and the database version. The database version is used only to collect database-specific data.

To collect BRM diagnostic data, run the following command:
```
    perl rda.pl -vdSCRP -p SupportInformer74
```

To collect BRM- and database-specific data, run the following command:
```
    perl rda.pl -vdSCR -p SupportInformer74-DB11g
```

**Note:** The database version in the SupportInformer74 profile depends on the version of the database installed for BRM. BRM supports Oracle Database 9i, 10g, and 11g.

The final output is packaged in an archive located in the output directory chosen during RDA setup. RDA output is not encrypted and can be viewed by anyone using any Web browser.

For information on reporting RDA problems, see "Reporting Problems".

### Viewing RDA Documentation

To view the RDA documentation, including the *Getting Started*, *FAQ*, and *Troubleshooting* guides:

1. Go to My Oracle Support (support.oracle.com).
2. In the Search Knowledge Base field, enter 330364.1 and click the Global Search icon.
3. In the search results, click the Remote Diagnostic Agent (RDA) - Main Man Page link.
   The Remote Diagnostic Agent (RDA) - Main Man Page appears.
4. In the RDA Main Links section of the page, click the link for the appropriate guide.
Dumping Business Parameters in XML Format

To dump BRM business parameters (/config/business_params objects) in XML format, use the `pin_cfg_bpdump` utility. For more information about business parameters, see "Using /config/business_params Objects" in BRM Developer’s Guide and "business_params Reference".

You can use the output as input to another application or utility, such as a diagnostic application. You can also direct the XML output to a file. For example, to direct the output to a file called `myfile.xml` in the same directory in which the utility is run, enter the following command:

```
pin_cfg_bpdump > myfile.xml
```

For each `/config/business_params` object, the utility outputs a `<RESULTS>` element that supplies identifying information about the object. The `<RESULTS>` elements include a `<PARAMS>` element for each parameter they include. A `<PARAMS>` element provides the parameter description, name, type, and value.

The following example shows output for the subscription business parameter object with three parameters:

```
<RESULTS elem="0">
  <POID>0.0.0.1 /config/business_params 8526 0</POID>
  <CREATED_T>1213082439</CREATED_T>
  <MOD_T>1213082439</MOD_T>
  <READ_ACCESS>G</READ_ACCESS>
  <WRITE_ACCESS>S</WRITE_ACCESS>
  <ACCOUNT_OBJ>0.0.0.1 /account 1 0</ACCOUNT_OBJ>
  <DESCR>Business logic parameters for Subscription</DESCR>
  <PROGRAM_NAME>subscription</PROGRAM_NAME>
  <VALUE />
  <VERSION />

  - <PARAMS elem="0">
    <DESCR>Parameter to enable contract days counter feature. This needs to be set to 1 if the accounts contain the resource contract days counter</DESCR>
    <PARAM_NAME>discount_based_on_contract_days_feature</PARAM_NAME>
    <PARAM_TYPE>1</PARAM_TYPE>
    <PARAM_VALUE>0</PARAM_VALUE>
  </PARAMS>

  - <PARAMS elem="1">
    <DESCR>Parameter to enable or disable best pricing feature. Enabling this feature will be effective only if license is loaded for best pricing. 1 means enabled</DESCR>
    <PARAM_NAME>best_pricing</PARAM_NAME>
    <PARAM_TYPE>1</PARAM_TYPE>
    <PARAM_VALUE>0</PARAM_VALUE>
  </PARAMS>

  - <PARAMS elem="2">
    <DESCR>Threshold of the number of offerings below which poids of offerings retrieved in PCM_OP_SUBSCRIPTION_GET_PURCHASED_OFFERINGS with a database search are cached for use in subsequent calls to the opcode in the same transaction. If the number of offerings retrieved is above this threshold, then the use of the cache could become inefficient</DESCR>
    <PARAM_NAME>get_offers_from_cache_threshold</PARAM_NAME>
    <PARAM_TYPE>1</PARAM_TYPE>
  </PARAMS>
</RESULTS>
```
To dump business parameters by using the `pin_cfg_bpdump` utility:

1. Go to the `BRM_Home/diagnostics/pin_cfg_bpdump` directory.
2. Run the following command:
   ```
   pin_cfg_bpdump
   ```
   To direct the output to a file, use the following syntax:
   ```
   pin_cfg_bpdump > file_name
   ```
   where `file_name` is the name of a file in the same directory in which the utility is run.

### Using Logs to Monitor Components

BRM records system activity in log files. One log file is generated for each component or application. Review these files daily to monitor your system and detect and diagnose system problems. You can also:

- Use HP OpenView to monitor log files for error messages. See "Monitoring Error Messages in Log Files by Using HP OpenView".
- Write scripts to look for certain conditions such as types or numbers of errors.
- Record opcode calls in the CM log file. See "Recording Opcode Calls in the CM Log File".

For information about understanding errors, see "Using Error Logs to Troubleshoot BRM".

For information about Pipeline Manager log files, see "About Pipeline Manager Log Files".

### Types of Log Files

BRM generates log files for system components, applications, and client applications.

#### Log Files for System Components

For system processes (or threads) such as CMs and DMs, BRM uses two types of log files:

- Those that record normal startup activity are named `program.log` (for example, `cm.log`, `js.log`, and `dm.log`).
- Those that record activity, such as error conditions, while the system is running. These pinlogs are named `program.pinlog` (for example, `cm.pinlog`, `js.pinlog`, and `dm_oracle.pinlog`).

#### Log Files for Applications

For BRM applications, log files are named `program.pinlog` (for example, `pin_billd.pinlog`). If an application is missing a configuration file (`pin.conf`) or if the application fails before it can read the configuration file, it records errors in the `default.pinlog` log file.
Log Files for Client Applications
BRM Java-based applications, such as Customer Center and Configuration Center, by default do not use log files. However, you can enable error logging by adding entries to the `Infranet.properties` file that provide configuration information when the application starts. For information about the `Infranet.properties` file, see "Setting Global Options" in BRM Developer’s Guide.

For Payment Tool, the log file `default.pinlog` is located in the same directory as the executable file.

Location of Log Files
The following are the minimum BRM log files:

- `cm.log`
- `cm.pinlog`
- `dm.log`
- `dm_oracle.pinlog`
- `pin_billd.pinlog`

Depending on what applications are running, your installation might also have one or more of these log files:

- `dm_email.log`
- `dm_email.pinlog`
- `pin_invoice_gen.log`
- `dm_fusa.log`
- `dm_fusa.pinlog`

Your customizations or special applications might generate their own log files.

You may want to increase the logging level to 2 (see "Setting the Reporting Level for Logging Messages") and have your notification script detect and act on warning messages. Log files should be archived weekly to a safe storage area.

Tip: You can write a script to compress the log files and then erase the originals. BRM automatically re-creates new empty log files as required.

Default Log File Locations
Log files for system components are stored in `BRM_Home/sys/component`. For example, the CM log file is in `BRM_Home/sys/cm`.

The `pinlog` files are stored in `/var/portal/7.4/component`. For example, the CM `pinlog` file is in `/var/portal/7.4/cm`.

For an application or client application log file, the default location is the directory from which the program was started.
You can leave log files in their default locations or move them.

**Changing the Name or Location of a Log File**

To change the name or location of the `pinlog` file for a component or application:

1. Open the configuration file (`pin.conf` or `Infranet.properties`) for the component or application. See “Locations of Configuration and Properties Files”.

2. Change the relevant entry:
   - `logfile`: Applications
   - `cm_logfile`: CM
   - `dm_logfile`: DM

3. Enter the desired name and directory for the log file.

4. Save and close the file.

5. Stop and restart the component or application. See “Starting and Stopping the BRM System”.

---

**Tip:** You can change the name of the default application’s log file by using the PIN_ERR_SET_LOGFILE function (see BRM Developer’s Reference).

---

**Note:** For Payment Tool, you cannot change the name of the log file. For Java-based BRM client applications, use an `Infranet.properties` file to specify the name and location of a log file.

---

**Setting the Reporting Level for Logging Messages**

By default, BRM components report error messages, and BRM applications report both error and warning messages. You can set BRM to report debugging messages or to not report errors. The four levels of error reporting are:

- **0** = no logging.
- **1** = (default) log error messages only.
- **2** = log error messages and warnings.
- **3** = log error, warning, and debugging messages.

---

**Important:** To avoid performance degradation, use only level 3 logging for debugging.

---

To change the severity level for logging:

1. Open the configuration file (`pin.conf` or `properties`) for the component or application. See “Locations of Configuration and Properties Files”.

2. Edit the `loglevel` entry. The notes in the configuration file define the options.

3. Save and close the file.

4. Stop and restart the component or application. See “Starting and Stopping the BRM System”.

Getting Debugging Information from Command-Line Utilities

Most BRM utilities use the following command-line parameters:

- **-d**: Set the log level to debug and outputs debug information into the log file. If not set, only error-level information is output. Use this parameter if no errors were reported, but the command was not successful (for example, if a `/config` object was not loaded).
- **-v**: Displays information about successful or failed processing as the utility runs.

**Note:** This parameter is always used in conjunction with other parameters and commands. It is not position dependent. For example, you can enter `-v` at the beginning or end of a command to initiate verbose display. To redirect the output to a log file, use the following syntax with the `-v` parameter. Replace `filename.log` with the name of the log file:

```
command any_other_parameter -v > filename.log
```

Dynamically Changing the CM and DM Log Levels

You can dynamically change the log level of the CM and DM without stopping and restarting them.

To change the log levels dynamically:

1. Pass the log level for the CM and the debug flags for the DM in the input flist to the `PCM_OP_INFMGR_SET_LOGLEVEL` opcode.

**Note:** You change the log levels of the CM and DM at the same time.

You can check the current log levels by calling the `PCM_OP_INFMGR_GET_LOGLEVEL` opcode.

2. Call the opcode by using `testnap`.

All the new processes after this opcode call will use the new CM log levels and DM debug flags.

Setting the Log Level for a Specific Opcode

You can record debug-level information for a specified opcode without having to reset the default system log level. This enables you to monitor the activity of a specific opcode (and any opcode it calls) without impacting system performance.

When you enable opcode logging, the logging level is increased to debug level 3 for the specified opcode only; all other opcodes are logged at the level specified in the CM `pin.conf` file.

You can define how many times during a CM session the debug-level reporting occurs for the specified opcode before the default reporting level is restored. This enables you
to increase the logging level without having to stop and restart the CM to reset it to the default level.

1. Open the CM pin.conf file in BRM_Home/sys/cm.
2. Set the pinlog_debug_opcode entry:
   
   ```
   cm pinlog_debug_opcode opcode
   ```
   
   where opcode is the opcode name or opcode number.

   **Note:** If this entry is not set, BRM uses the loglevel entry in the CM pin.conf file to determine the log level.

3. Set the pinlog_debug_op_count entry:
   
   ```
   cm pinlog_debug_op_count number
   ```
   
   where number is the number of times the opcode is recorded at the debug level before the default log level is restored.
4. Save and close the file.
5. Restart the CM. See "Starting and Stopping the BRM System".

For information on setting the system log level, see "Setting the Reporting Level for Logging Messages".

### Recording Opcode Calls in the CM Log File

You use the enable_pcm_op_call_stack and max_pcm_op_call_stack_entries CM pin.conf entries to record opcodes in the CM log file.

When enable_pcm_op_call_stack is enabled, the opcodes that are called by BRM clients are recorded in the CM log file.

See "Connection Manager (CM) pin.conf Entries".

### About Formatting Log Files

You can format a log file to improve readability and traceability of errors by using the splitPinlog script. This script splits a log file into multiple files, one for each combination of process ID (PID) and thread ID (TID) based on the information in the header of the pinlog entries.

To format a log file:

1. Go to the BRM_Home/bin directory.
2. Run the following Perl script:
   
   ```
   splitPinlog original_pinlog_file
   ```
   
   The Perl script creates a file with the name `original_pinlog_file.pid.tid.pinlog`.

   For example, running the command:
   
   ```
   splitPinlog cm.pinlog
   ```
   
   results in these file names:
   
   ```
   cm.pinlog.342353.12.pinlog
   ```
Maintaining Log Files

Large log files degrade system performance. Check the sizes of log files periodically and delete or archive large files. When you delete or rename a log file, a new empty file is created as soon as a new log entry is created and either a maximum of four hours have elapsed or the application is stopped and restarted. Be especially vigilant when using new custom applications, which commonly makes log files grow quickly.

Checking the Number and ID of a BRM Process

You can check the number of processes running for the CM or a DM. The number should match the number specified in the configuration file (pin.conf) for that component. If not, the processes either did not start or have stopped. You can also look at the process ID (PID) for each process.

Enter the following command:

```
ps -ef | grep process
```

The system shows each process and its ID.

For example, to show the processes running for the Paymentech DM, enter the following command:

```
ps -ef | grep dm_fusa
```

Dealing with Hung and Looping Processes

A hung process does not respond in a normal fashion.

A looping process uses CPU cycles without doing any useful work.

Checking for Hung Processes

If the CM does not respond to a login attempt, one of the processes in the system might be hung. Check the status of the CM. See “Monitoring CM Activity”. The CM should show a new connection. If the CM report shows that the CM is “waiting on DM,” the DM might be hung. See “Manually Checking the Status of the DM”. You can check the database by verifying that it responds to manual SQL commands.

Checking for Looping Processes

If the CPU time for a process is increasing and is out of proportion to the rest of the processes, this might be a looping process. To check the CPU time used by a process, enter the following command twice, separated by a 10- to 30-second interval (or as much as several minutes on a lightly loaded system):

```
ps -ef | grep process
```
Stopping a Hung or Looping Process

Enter the following command to stop a hung or looping process:

```
kill -ABRT process_id
```

BRM stops the process and writes a core image file of the process. If you contact Oracle Technical Support about this problem, send the core file along with the relevant log files. (See "Getting Help with BRM Problems").

Monitoring CM Activity

You can check the CM's status and resolve lost TCP connections.

Manually Checking the Status of the CM

You can monitor the operation of the CM by checking the status at regular intervals and comparing the results with what you expect.

To check the status of the CM:

1. Find the process ID (PID) of the master CM process by looking in the pid file for the CM in BRM_Home/sys/cm.
2. Enter the following command:

```
kill -USR1 PID_of_CM
```

BRM displays a report on the CM, which shows information about the master CM such as the version and the number of children. If there are CM children, the rest of the reports consist of a single line for each child showing the state, the IP address and port for the application, and the IP address and port of the current DM connection.

Table 2–4 describes the state values:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Reading from (or waiting to read from) the application</td>
</tr>
<tr>
<td>2</td>
<td>Starting to process the operation</td>
</tr>
<tr>
<td>3</td>
<td>Facilities Module processing in progress (if going to FM)</td>
</tr>
<tr>
<td>4</td>
<td>Facilities Module processing done, sending response</td>
</tr>
<tr>
<td>5</td>
<td>Finding DM address (if going to DM)</td>
</tr>
<tr>
<td>6</td>
<td>Sending operation to DM</td>
</tr>
<tr>
<td>7</td>
<td>Waiting on DM</td>
</tr>
<tr>
<td>8</td>
<td>Forwarding DM response to application</td>
</tr>
<tr>
<td>9</td>
<td>Cleaning up after the operation</td>
</tr>
<tr>
<td>10</td>
<td>Shutting down the child CM</td>
</tr>
</tbody>
</table>

Note: Before you stop a hung or looping DM or CM process, check its status at least twice at 30-second intervals (or up to several minutes on a lightly loaded system). For more information, see "Monitoring DM Activity" or "Monitoring CM Activity".
Resolving Lost TCP Connections

BRM recognizes when an application closes a TCP connection. If the computer running the client application fails, however, the application might not close the TCP socket.

In the `pin.conf` files for the CM and the Connection Manager Master Process (CMMMP), the `keepalive` entry specifies whether to monitor the TCP connection.

```
Note: This entry should be set to avoid sockets not being closed properly due to network problems or hardware crashes.
```

The CM monitors the TCP connections by using the standard TCP keepalive feature. This lets you detect lost connections and clean up the CM and DM.

With the keepalive feature turned on, BRM uses the system’s keepalive APIs to detect a lost connection and to try to reconnect, before closing the socket.

For more information about TCP keepalive options, see the TCP and keepalive documentation for your operating system.

Enabling Java PCM Clients to Use Operating System TCP/IP Keepalive Parameters

If a connection for a Java PCM client is not in use for some time, a BAD_READ error may result. If this becomes a recurring problem, you can enable the client to use the underlying operating system TCP/IP keepalive parameters such as keepalive time, keepalive interval, and keepalive retry.

To enable Java PCM clients to use operating system TCP/IP keepalive parameters:

1. Open the `Infranet.properties` file of the Java PCM client.
2. Add the following entry:
```
infranet.pcp.socket.keepalive.enabled=true
```

- **true** enables Java PCM clients to use operating system TCP/IP keepalive parameters.
- **false** prevents Java PCM clients from using operating system TCP/IP keepalive parameters.

By default, BRM prevents Java PCM clients from using operating system TCP/IP keepalive parameters.

3. Save and close the file.

Setting the CM Log Time Resolution

By default, the time resolution in CM log files is in seconds. If you need a higher resolution to help diagnose performance issues, change the resolution to milliseconds.

---

**Table 2–4 (Cont.) CM State Values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>Starting the child CM</td>
</tr>
</tbody>
</table>
To set the CM log time resolution:

1. Open the CM `pin.conf` file in `BRM_Home/sys/cm`.
2. Change the value of the `cm_logformat` entry from 0 to 1, where 0 sets the log time resolution to seconds and 1 sets the log time resolution to milliseconds.
3. Save and close the file.
4. Stop and restart the CM. See "Starting and Stopping the BRM System".

**Setting a Timeout Value for Requests Sent to the CM**

BRM client applications process requests in a synchronous mode; that is, they wait for a response from the CM before sending the next request. Therefore, if there is an error on the server side, the client application has to wait indefinitely. To prevent this problem, you can set a timeout value for requests sent to the CM. If the CM does not respond within the time specified, the PCP connection layer returns an error message to the client application and closes the connection.

To specify a timeout value, configure your client applications as follows:

- For BRM client applications that use a configuration (`pin.conf`) file:
  1. Open the `pin.conf` file in a text editor.
     
     By default, the `pin.conf` file is in `BRM_Home/apps/application_name`, where `application_name` is the name of the application, such as `pin_billd`.
  2. Add the following entry to the file:
     
     ```
     - nap pcm_timeout_in_msecs milliseconds
     ```
     
     where `milliseconds` is the number of milliseconds to wait before returning an error message and closing the connection.
  3. Save and close the file.

- For BRM client applications that use the `Infranet.properties` file:
  1. Open the `Infranet.properties` file in a text editor.
     
     By default, the `Infranet.properties` file is in `C:/Program Files/Common Files/Portal Software`.
  2. Add the following entry to the file:
     
     ```
     infranet.PcmTimeoutInMsecs=milliseconds
     ```
     
     where `milliseconds` is the number of milliseconds to wait before returning an error message and closing the connection.
  3. Save and close the file.

---

**Note:** The timeout value specified in the configuration or `Infranet.properties` file is used for all open connections. If a timeout value is set for a connection in the application itself, that value overrides the value in the configuration or properties file entry.

---

For information on setting timeout values for each connection in your custom C and Java client applications, see "Implementing Timeout for Requests in Your Application" and "Specifying a Timeout Value for Requests" in *BRM Developer's Guide*. 
Configuring Multilevel CM Timeout for Client Requests

You can configure the CM to use two timeouts for handling client requests:

- A short (suspect) timeout
- A long (failover) timeout

A Short (Suspect) Timeout

When this timeout period expires, the request for the DM connection is placed in a suspect state, the current transaction is stopped, and the request is returned to the client with the PIN_ERR_TIMEOUT and PIN_ERRCLASS_SYSTEM_SUSPECT errors.

To configure the suspect timeout:

1. Open the CM configuration file (`BRM_Home/sys/cm/pin.conf`) in a text editor.
2. Add the following entry to the file:
   
   ```
   pcm_suspect_timeout_in_msecs = milliseconds
   ```
   
   where `milliseconds` is the number of milliseconds in the suspect timeout period.

   **Note:** The value of this entry must be smaller than the value of the `pcm_timeout_in_msecs` entry.

3. Save and close the file.

A Long (Failover) Timeout

When this timeout period expires, the CM returns a PIN_ERR_TIMEOUT error to the client. In a high-availability system with multiple DMs configured, the CM connects to the secondary DM to process the requests.

To configure the failover timeout:

1. Open the CM configuration file (`BRM_Home/sys/cm/pin.conf`) in a text editor.
2. Add the following entry to the file:
   
   ```
   pcm_timeout_in_msecs milliseconds
   ```
   
   where `milliseconds` is the number of milliseconds in the failover timeout period.

   **Note:** The value of this entry should be larger than the value of the `pcm_suspect_timeout_in_msecs` entry.

3. Save and close the file.

Getting Quality of Service Statistics from the CM

You can collect statistics about CM opcode performance (for example, the number of times an opcode is called or the number of times an opcode returns an error). You can collect statistics on a per-opcode basis. The statistics are written to the CM log file whenever a client connection closes. You can enable and disable this feature by modifying the CM `pin.conf` file.
To measure latency for an opcode, you can specify up to seven maximum latency times, with each latency time period representing a QoS bucket. For example, if you specify latencies of 10, 20, and 100, the buckets are:

- 0-10 milliseconds: QoS bucket 1
- 10-20 milliseconds: QoS bucket 2
- 20-100 milliseconds: QoS bucket 3
- Greater than 100 milliseconds: QoS bucket 4

The QoS buckets are defined as follows:

- QoS bucket 1: less than or equal to QoS time 1
- QoS bucket 2: greater than QoS time 1 and less than or equal to QoS time 2
- QoS bucket 3: greater than QoS time 2 and less than or equal to QoS time 3
- QoS bucket 4: greater than QoS time 3

The information listed in Table 2–5 is collected per opcode:

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Opcode</td>
<td>The opcode that this information pertains to.</td>
</tr>
<tr>
<td>Interval timestamp</td>
<td>The starting timestamp of this interval.</td>
</tr>
<tr>
<td>Total opcode call count</td>
<td>The number of times this opcode has been called in this time interval.</td>
</tr>
<tr>
<td>Total error count</td>
<td>The number of times this opcode has returned an error in this time interval.</td>
</tr>
<tr>
<td>Minimum latency</td>
<td>The fastest elapsed time that this opcode took to complete without returning an error.</td>
</tr>
<tr>
<td>Timestamp of minimum latency</td>
<td>The timestamp when the minimum latency occurred.</td>
</tr>
<tr>
<td>Maximum latency</td>
<td>The slowest elapsed time that this opcode took to complete without returning an error.</td>
</tr>
<tr>
<td>Timestamp of maximum latency</td>
<td>The timestamp when the maximum latency occurred.</td>
</tr>
<tr>
<td>Total latency</td>
<td>Total latency of all successful calls to this opcode, not including the calls that returned an error.</td>
</tr>
<tr>
<td>Input flist of maximum latency</td>
<td>The input flist that was used when the maximum latency occurred.</td>
</tr>
<tr>
<td>QoS bucket count</td>
<td>The number of active QoS buckets for this opcode.</td>
</tr>
<tr>
<td>QoS bucket 1 counts</td>
<td>The number of times that the latency of a successful call to the opcode falls into each bucket. For example, 10 in bucket 1, 12 in bucket 2, and so forth.</td>
</tr>
<tr>
<td>QoS bucket times2 count</td>
<td>The maximum time in nanoseconds for each QoS bucket.</td>
</tr>
<tr>
<td>Timestamp of first received opcode</td>
<td>The timestamp when the first opcode was received.</td>
</tr>
<tr>
<td>Timestamp of last received opcode</td>
<td>The timestamp when the latest opcode was received.</td>
</tr>
</tbody>
</table>

### Configuring CM QoS Statistics

To enable or disable the collection of opcode QoS statistics:

1. Open the CM `pin.conf` file in `BRM_Home/sys/cm`.
2. Change the value of the `cm_opcode_stats` entry.

   The syntax is:
   
   ```
   cm cm_opcode_stats opcode QoS_1 [, QoS_2, ... QoS_7]
   ```
where *opcode* can be an opcode name or opcode number.

For example, to use an opcode name and four buckets, enter the following:

```
- cm cm_opcode_stats PCM_OP_CUST_COMMIT_CUSTOMER 10, 20, 30
```

For example, to use an opcode number and four buckets, enter the following:

```
- cm cm_opcode_stats 63 10, 20, 30
```

---

**Note:** If the entry does not exist, you can add it anywhere in the file.

3. Save and close the file.
4. Stop and restart the CM. See "Starting and Stopping the BRM System".

---

### Monitoring DM Activity

You can check the status of a Data Manager (DM) at regular intervals to monitor resource usage. You can also make inferences about the operation of the DM by checking the status at intervals and comparing the results with what you expect.

#### Manually Checking the Status of the DM

You can check and view the status of the DM in flist format and in a report format.

- Checking the DM Status in flist Format
- Checking the DM Status in a Report Format

#### Checking the DM Status in flist Format

To check the status of a DM in flist format:

1. Go to the `BRM_Home/sys/test` directory.
2. Enter the following commands:

```
testnap
robj - database_number /status_dm 1
```

where `database_number` is the database number of the DM for which you want the status.

BRM displays the status of the DM in flist format.

*Table 2–6* describes the fields in `/status_dm`.

---

<table>
<thead>
<tr>
<th>/status_dm field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIN_FLD_DM_BIGSIZE</td>
<td>Specifies the size, in kilobytes, of the big part of the DM shared memory.</td>
</tr>
<tr>
<td>PIN_FLD_SM_PASSTHRU_NAME</td>
<td>Specifies the current value of the <code>dm_sm_pass_thru_obj</code> entry in the DM <code>pin.conf</code> file.</td>
</tr>
<tr>
<td>PIN_FLD_SM_SHMSIZE</td>
<td>Specifies the maximum shared memory size, in kilobytes, for a custom DM.</td>
</tr>
<tr>
<td></td>
<td><em>Note:</em> Ignore this field if your system uses Oracle DM.</td>
</tr>
</tbody>
</table>

---

Monitoring and Maintaining Your BRM System 2-41
Table 2–6  (Cont.) /status_dm Object Fields

<table>
<thead>
<tr>
<th>/status_dm field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIN_FLD_TRANS_OP_QUEUED</td>
<td>Specifies the number of transactions currently queued. This is an instantaneous counter.</td>
</tr>
<tr>
<td>PIN_FLD_DM_BACKEND</td>
<td>Array that defines the DM back end.</td>
</tr>
<tr>
<td>PIN_FLD_FLAGS</td>
<td>Specifies the internal state of the DM back end. These states are used for the internal working of the Oracle DM.</td>
</tr>
<tr>
<td></td>
<td>- DMSHM_ALIVE 0x00001000</td>
</tr>
<tr>
<td></td>
<td>- DMSHM_DYING 0x00002000</td>
</tr>
<tr>
<td></td>
<td>- DMSHM_DEAD 0x00004000</td>
</tr>
<tr>
<td></td>
<td>- DMSHM_INITING 0x00008000</td>
</tr>
<tr>
<td></td>
<td>- DMSHM_RESTARTING 0x00010000</td>
</tr>
<tr>
<td>PIN_FLD_TATTLE_TALE</td>
<td>This flag is reset each time you retrieve the DM status report. This allows you to see what happened since the last DM report.</td>
</tr>
<tr>
<td></td>
<td>- DM_TATTLE_SELECT 0x0001</td>
</tr>
<tr>
<td></td>
<td>- DM_TATTLE_CMD 0x0002</td>
</tr>
<tr>
<td></td>
<td>- DM_TATTLE_IO_IN 0x0004</td>
</tr>
<tr>
<td></td>
<td>- DM_TATTLE_IO_OUT 0x0008</td>
</tr>
<tr>
<td></td>
<td>- DM_TATTLE_ALL (DM_TATTLE_SELECT</td>
</tr>
<tr>
<td>PIN_FLD_DM_FRONTEND</td>
<td>Array that defines the DM front end.</td>
</tr>
<tr>
<td>PIN_FLD_FLAGS</td>
<td>Specifies the internal state of the DM front end. These states are used for the internal working of the Oracle DM.</td>
</tr>
<tr>
<td></td>
<td>- DMSHM_ALIVE 0x00001000</td>
</tr>
<tr>
<td></td>
<td>- DMSHM_DYING 0x00002000</td>
</tr>
<tr>
<td></td>
<td>- DMSHM_DEAD 0x00004000</td>
</tr>
<tr>
<td></td>
<td>- DMSHM_INITING 0x00008000</td>
</tr>
<tr>
<td></td>
<td>- DMSHM_RESTARTING 0x00010000</td>
</tr>
<tr>
<td>PIN_FLD_TATTLE_TALE</td>
<td>This flag is reset each time you retrieve the DM status report. This allows you to see what happened since the last DM report.</td>
</tr>
<tr>
<td></td>
<td>- DM_TATTLE_SELECT 0x0001</td>
</tr>
<tr>
<td></td>
<td>- DM_TATTLE_CMD 0x0002</td>
</tr>
<tr>
<td></td>
<td>- DM_TATTLE_IO_IN 0x0004</td>
</tr>
<tr>
<td></td>
<td>- DM_TATTLE_IO_OUT 0x0008</td>
</tr>
<tr>
<td></td>
<td>- DM_TATTLE_ALL (DM_TATTLE_SELECT</td>
</tr>
<tr>
<td>PIN_FLD_CONNECTS</td>
<td>Specifies the number of concurrent connections the front end has received. This is an instantaneous counter.</td>
</tr>
<tr>
<td>PIN_FLD_HIWAT</td>
<td>Specifies the maximum number of concurrent connections the front end received during the life of the DM. This is the maximum value reached by PIN_FLD_CONNECTS for this front end</td>
</tr>
<tr>
<td>PIN_FLD_DM_FE_CONNECT</td>
<td>Array that defines the front-end connection.</td>
</tr>
</tbody>
</table>
Checking the DM Status in a Report Format

To check the status of the DM in a report format:

1. Find the process ID (PID) of the master DM process by looking in the *pid* file for the DM in `BRM_Home/sys/dm_oracle`.

2. Enter the following command:

   ```
   kill -USR1 PID_of_DM
   ```

   where *PID_of_DM* is the process ID of the master DM process.

BRM displays the status of the DM in the `dm_oracle.log` file. The log file shows information about the DM, such as the PID, memory usage, transaction queue, and information about the back ends and the front ends.
Monitoring DM Shared Memory Usage

You can check shared memory usage by looking in the master overview section of the DM report. The number of used and free heap blocks (# used and # free) shows memory usage, expressed in 8-KB blocks. To prevent failures associated with insufficient memory, verify that # free is a relatively large number. If # free is a small portion of # used, you should increase the size of the shared memory area. Otherwise, operations might fail, returning PIN_ERR_NO_MEM.

Monitoring DM Transaction Queues

To check the status of transactions, look in the master overview section of the DM report. The trans_op_cnt entry shows the number of transactions currently being processed, and the trans_op_queued entry shows the number waiting to be processed. For applications that require rapid response times, you can adjust the load on the system to keep to a minimum the number of transactions waiting to be processed. See "Improving Data Manager and Queue Manager Performance".

Monitoring DM Back Ends

You can use the back-end report to identify each back-end process ID (PID), the back-end status, and the number of operations processed. A value of 0x1000 (4096) for FLAGS shows that the back end is active. The report also gives information on resource usage.

The second flag value is reset each time the DM status report is received. Therefore, you can tell what has happened (at least once) since the last DM report by a flag bit being clear.

Table 2–7 shows the flag bit values:

<table>
<thead>
<tr>
<th>Value</th>
<th>Flag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x8</td>
<td>IO output</td>
<td>Never cleared for back ends.</td>
</tr>
<tr>
<td>0x4</td>
<td>IO input</td>
<td>Cleared when the back end starts an operation.</td>
</tr>
<tr>
<td>0x2</td>
<td>CMD</td>
<td>Cleared when the back end is given a command or transaction.</td>
</tr>
<tr>
<td>0x1</td>
<td>SELECT</td>
<td>Cleared when the back end wakes up using select(2).</td>
</tr>
</tbody>
</table>

On a quiet back end, the second flag value stays at f. The counters of most interest are those that keep track of the total number of operations and total transactions.

As shown in Table 2–8, the back-end state values are a bit mask flag:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x1</td>
<td>Busy; currently doing an operation.</td>
</tr>
<tr>
<td>0x2</td>
<td>Locked to a transaction.</td>
</tr>
</tbody>
</table>

The back-end index and operation may be left over from the previous operation and may be no longer valid. The used field indicates memory usage. When idle, one 8-KB chunk is normally used. During an operation or transaction, this amount varies.
Monitoring DM Front Ends

You can use the front-end report to identify each front-end process ID (PID), the front-end status, and the number of operations processed. A value of 0x1000 (4096) for FLAGS shows that the front end is active.

For each connection, the report also gives a snapshot of the connection status. When idle, the state values should each be 0 (zero).

Table 2–9 describes the front-end state values:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Waiting to receive an operation from the CM.</td>
</tr>
<tr>
<td>1</td>
<td>Receiving from the CM.</td>
</tr>
<tr>
<td>2</td>
<td>Sent an operation to be processed, waiting for back end.</td>
</tr>
<tr>
<td>3</td>
<td>The operation is done.</td>
</tr>
<tr>
<td>4</td>
<td>Sending a response to the CM.</td>
</tr>
</tbody>
</table>

The front-end flags are the same as the back-end flags, except that the front ends clear the IO output value when they send a reply back to the CM. The information in the connection report is a snapshot of the connection status.

Increasing the Level of Reporting for a DM

By default, DMs report errors and warnings. You can have a DM report debugging messages as well.

You can specify which debugging messages you want written to the log. There are three settings to control which debugging information is logged:

- DM_DEBUG variables control the logging of opcode-processing debug messages.
- DM_DEBUG2 variables control the logging of data dictionary processing debug messages.
- DM_DEBUG3 variables debug the SQL statements produced by different parts of the DM.

The BRM_Home/include/dm_debug.h file contains definitions of the flags you can set. You specify which individual flags you want to enable for each setting by summing the values of the flags and including the sum in an environment variable or in the DM configuration (pin.conf) file.

For example, to log information about transaction tracing, you set DM_DEBUG to 0x70, which is the sum of the following individual flags:

DM_DEBUG_TRANS_IN_PR = 0x10
DM_DEBUG_TRANS_OUT_PR = 0x20
DM_DEBUG_TRANS_TRACE = 0x40

Depending on what information you want to log, you can include values for any combination of the three settings (DM_DEBUG, DM_DEBUG2, and DM_DEBUG3).

The way you increase the level of reporting depends on your operating system and the DM:
For all DMs other than `dm_oracle` and `dm_tax`, you can include debug statements in the DM's configuration (`pin.conf`) file. You specify each setting separately as in the following example:

- `dm dm_debug 0xFFF003FF`
- `dm dm_debug2 0x10`
- `dm dm_debug3 0x10`

See "Editing the Configuration File to Set Debug Options".

For `dm_oracle` and `dm_tax`, you must specify the debugging information as environment variables. You set a separate environment variable for each debug setting.

See "Using Environment Variables to Set Debug Options".

You can dynamically change the debugging level without stopping the DM. For more information, see "Dynamically Changing the CM and DM Log Levels".

**Using Environment Variables to Set Debug Options**

To set debug options for `dm_oracle` and `dm_tax`:

1. Stop the DM. See "Starting and Stopping the BRM System".
2. In the environment from which the DM starts, set the environment variable for debugging. For example:
   - C-shell:
     ```bash
     setenv DM_DEBUG3 0xFFFF003F
     ```
   - Korn shell:
     ```bash
     DM_DEBUG3=0xFFFF003F
     export DM_DEBUG3
     ```
3. Start the DM. See "Starting and Stopping the BRM System".
4. Run the DM operations for which you want debugging information.
5. Stop the DM.
6. Open the log file for the DM (for example, `dm_oracle.pinlog`) and review the messages.
7. Return DM logging to its normal level. Otherwise, subsequent DM activity will generate large log files.

**Editing the Configuration File to Set Debug Options**

To set debug logging options for all DMs except `dm_oracle` and `dm_tax`:

1. Stop the DM. See "Starting and Stopping the BRM System".
2. Open the configuration file (`pin.conf`) for this DM. See "Locations of Configuration and Properties Files".
3. Edit the three debugging entries to set the level of debugging reporting.
4. Save and close the file.
5. Start the DM. See "Starting and Stopping the BRM System".
6. Run the DM operations for which you want debugging information.
7. Stop the DM.
8. Open the log file for the DM (for example, dm_fusa.pinlog) and review the messages.

9. Return DM logging to its normal level by commenting out the debugging entries in the configuration file. Otherwise, subsequent DM activity will generate large log files.

**Logging the DM Process Time Information for Performance Diagnostics**

To diagnose performance problems with the DM process, you can configure the DM to log the time it takes to process each opcode. You can use this information to determine the time the DM spends on its internal operations and the time it spends on the database operations.

Before the DM starts processing an opcode, it logs the current time. Then for each SQL statement that the DM sends to the database for the opcode, it logs the following information:

- Session ID
- Statement ID
- Time taken by the database to process the SQL statement

To log the timing information for the SQL statement, set the DM_DEBUG3 flag to 0x00010000, which corresponds to the DM_DEBUG3_TIME_INFO variable defined in the BRM_Home/include/dm_debug.h file.

You can also dynamically set or change this variable when the DM is running. See "Dynamically Changing the CM and DM Log Levels".

**Replacing Failed DM Child Processes**

All DMs, such as Paymentech DM, Email DM, and Taxware DM, are set to automatically replace child processes that have stopped. This feature prevents the system from losing DM processes as a result of transient failures over time. For initial testing, or if you have recurring errors that would cause a “fork and die” endless loop (in an Oracle database, for example), you can tell the DM to not replace failed child processes:

1. Open the configuration file (pin.conf) for this DM. See "Locations of Configuration and Properties Files".
2. Change the value of the dm_restart_children entry to 0.
3. Save and close the file.
4. Stop and restart the DM. See "Starting and Stopping the BRM System".

When a child process stops and is replaced, BRM notes the event in the error log file for the DM.

---

**Note:** BRM does not automatically replace child processes that are hung. See "Dealing with Hung and Looping Processes".

---

**Monitoring Pipeline Manager**

For information about improving Pipeline Manager performance, see "Optimizing Pipeline Manager Performance".
Monitoring Pipeline Manager Memory Usage

You can use the MemoryMonitor module to monitor Pipeline Manager memory during startup and while it is processing files. You set a threshold for the amount or percentage of memory that determines when Pipeline Manager should issue a warning or gracefully shut down. You can set the thresholds as a percentage or as kilobytes or megabytes.

For example, if you set **ShutdownFreeMemLimit** to 50 and **ScaleUnit** to M, Pipeline Manager shuts down gracefully when the remaining free system memory reaches 50 MB. If you set **WarningFreeMemLimit** to 10 and **ScaleUnit** to P, Pipeline Manager logs a warning when the remaining free system memory reaches 10 percent.

See "Memory Monitor" in *BRM Configuring Pipeline Rating and Discounting*.

Monitoring Pipeline Manager EDR Throughput

You can monitor the following statistics for each pipeline:

- Number of event data records (EDRs) since startup.
- Accumulated EDR processing time since startup.
- Total number of EDRs since startup, independent of any transaction. This number is incremented after every processed EDR.
- Total number of EDRs after the transaction ended. This number is not incremented until the current transaction has ended.
- The real-time EDR count increments after each EDR is processed, while the transaction count increments EDR count only after transaction/file processing is ended.
- Number of transactions since startup.
- EDRs per second (throughput). This data includes the timestamp of when the measurement was taken.

You can use the Operations Management Framework (OMF) HTTP and SNMP protocols to access the data. See "Pipeline Statistics Probes".

You can use HP OpenView alarms to monitor and report on EDR throughput. See "Using Alarms".

Getting Recent Pipeline Log File Entries

You can display recent log file entries in the OMF HTTP server. See "Using the HTTP Instrumentation Protocol to Read OMF Instrumentation Data". The entries are also included in the Diagnostic Data Handler output file. See "Using the Diagnostic Data Handler to Get OMF Diagnostic Data".

The log messages are stored in a circular buffer that stores the last 1000 log messages. See "Log File Probes".

You can change the number of error messages stored in the buffer. To do so, edit the **CircularBufferSize** registry entry in the ITO section.

For example:

```plaintext
ProcessLog
{
    ModuleName = LOG
    Module
    {
```
IT0
{
    LogLevel = Debug
...
    CircularBufferSize = 100
}

Managing IMDB Cache DM

This section provides information for monitoring and maintaining IMDB Cache DM.

Monitoring IMDB Cache DM

In addition to the pin_ctl utility, you can use the following features to monitor the IMDB Cache DM:

- Core dumps. See "Generating the IMDB Cache DM Core Dump".
- Log files. See "Troubleshooting IMDB Cache DM Errors".
- Opcode latency statistics. See "Getting Opcode Statistics from IMDB Cache DM".

Generating the IMDB Cache DM Core Dump

To generate the IMDB Cache DM core dump:

1. Go to the system where IMDB Cache DM is started.
2. Enter the following command:
   ```
   setenv sbUtDumpCore 1
   ```

Troubleshooting IMDB Cache DM Errors

By default, IMDB Cache DM reports errors and warnings in the dm_tt.pinlog file. Additionally, you can use environment variables to set debug options to report debugging information in the log file.

IMDB Cache DM logs system activities in the BRM_Home/sys/dm_tt/dm_tt.pinlog file. Any error in the IMDB Cache DM is reported in the dm_tt.pinlog file, and the error number is returned to the Connection Manager (CM). You should monitor this log file daily to detect and diagnose system problems. You might want to create a script file to periodically scan the log file and notify you if it detects any error messages.

Getting Opcode Statistics from IMDB Cache DM

You can collect statistics about opcode performance from IMDB Cache DM. IMDB Cache DM prints the opcode stack with details about the total time spent at Oracle IMDB Cache and at the BRM database. This data can be used to compare opcode performance and for debugging purposes. For example, if the database operation is taking more time, check the database statistics to ensure the database is running optimally.

To get opcode statistics from IMDB Cache DM, set the following entry in the IMDB Cache DM and Oracle DM pin.conf files:

```
- dm enable_pcm_op_call_stack 1
```

The opcode stack is printed for the whole transaction after the transaction is committed or aborted.

The following is a sample opcode stack output:
To customize the opcode stack size, set the following optional configuration entry in the IMDB Cache DM and Oracle DM pin.conf files.

- dm max_pcm_op_call_stack_entries Size

## Maintaining IMDB Cache DM

This section describes steps you can take to maintain the IMDB Cache DM, including:

- Handling Active IMDB Cache DM Failure
- Handling Active Node Failure
- Managing Cache Group Data

### Handling Active IMDB Cache DM Failure

In a high-availability system, when an active IMDB Cache DM fails, its associated data store is not notified of the failure, so the data store’s status remains active. This prevents the standby data store from becoming active.

Because its associated data store is still on standby, the standby DM rejects all CM requests with the PIN_ERR_NOT_ACTIVE error to indicate that it is in standby mode and not accepting requests. (The PIN_ERR_NOT_ACTIVE error is recorded as PIN_ERR_NOT_PRIMARY in the CM log file.)

Therefore, if an internal IMDB Cache DM error prevents Oracle Clusterware from restarting a DM, you must manually change the standby data store’s state to active. This enables the standby DM to switch its state to active and process the requests redirected to it by the CM.

All CM requests will fail until either the active or standby IMDB Cache DM establishes a connection with an active data store.

### Handling Active Node Failure

When Oracle IMDB Cache goes down, you have to restore the data store by detaching and reattaching the data store to the grid and then re-creating the schema and reloading the BRM objects.

To restore the data store, do the following:

1. Detach and reattach the data store to the grid. See Oracle In-Memory Database Cache User’s Guide for information on how to detach and reattach data store to a grid.
2. Using ttIsqI, run tt_schema.sql on Oracle IMDB Cache to re-create the BRM cache groups schema.
3. Using ttIsql, run the load SQL file on Oracle IMDB Cache to reload the BRM objects into the cache groups.
Managing Cache Group Data
Data in the cache groups is stored in shared memory. To avoid running out of shared memory, purge expired BRM objects from the Oracle IMDB Cache to free shared-memory space. Additionally, you can configure an aging policy for the cache groups to purge least-recently-used (LRU) objects.

See "About Managing Data in IMDB Cache" for more information about purging BRM objects from the Oracle IMDB Cache.

Monitoring Customer Center Activity
You can configure Customer Center to send flist information to a log file by using the SDK to modify the Infranet.properties file. You can use this information to monitor Customer Center activity and to resolve problems.

See "Using Customer Center SDK" in BRM Developer’s Guide.

Monitoring Hardware and Operating Systems
To monitor your system using standard tools, use monitoring utilities such as vmstat, sar, and top on UNIX, or use OS performance monitors such as Glance on HP-UX IA64 systems.

On Solaris systems, use sysdef to find information about kernel parameter settings. This is especially useful for determining if per-process shared memory, file descriptor, or thread limits are adequate. pmap is useful for separating memory usage into total, resident, shared, and private.

Checking the Version Numbers of Components
You can check the version numbers of all BRM and pipeline components installed on a machine by using the pinrev and piperev utilities. These utilities return the following information for each component, ServicePak, FeaturePak, and patch installed on a machine:

- Product name
- Version number
- Components
- Build time
- Installation time

**Tip:** Run these utilities whenever you are working with Oracle Technical Support to help them re-create your system environment and troubleshoot your problems.
Checking BRM Component Version Numbers

To check which BRM components are installed on your machine, go to the BRM_Home/bin directory and run the pinrev utility:

```
pinrev
```

BRM displays the versions of all products installed on your machine. For example:

```
PRODUCT_NAME=Vertex
VERSION=7.4
COMPONENTS=" Vertex_DM Vertex_Comtax Portal_Perl Portal_Uninstaller Shared_PCM_Perl Shared_PCM_Components ">
Build: Thu Sep 25 12:52:14 PDT 2003
====================================================================
PRODUCT_NAME=Vertex
VERSION=7.4
COMPONENTS=" Vertex_DM ">
Build: Thu Sep 25 12:52:14 PDT 2003
====================================================================
```

**Tip:** To print a report of the version information, direct the output of the `pinrev` utility to a file with a `.csv` extension, which you can open in Microsoft Excel. For example:

```
pinrev > BRM.csv
```

**Important:** The `pinrev` utility does not display information about the uninstallation of any BRM component. Only installation information is displayed.

Checking Pipeline Component Version Numbers

To check which pipeline components are installed on your machine, go to the Pipeline_Home/tools directory and run the piperev utility:

```
piperev
```

The pipeline displays the versions of all products installed on your machine. For example:

```
PRODUCT_NAME=Pipeline
VERSION=7.4
COMPONENTS= "Common files","Pipeline Framework Files","Pipeline Sample Files","Pipeline Database Scripts","Pipeline Misc Files","Pipeline Tools files","PDK files","Rogue Wave Files",
BUILD_TIME= 6-19-2009 18:46:58
INSTALLED_TIME=Wed, 24 Jun 2009 06:07:09 -0700
```

**Tip:** To print a report of the version information, direct the output of the `piperev` utility to a file with a `.csv` extension, which you can open in Microsoft Excel. For example:

```
piperev > BRM.csv
```
Using the Diagnostic Data Handler to Get OMF Diagnostic Data

Use the Diagnostic Data Handler to collect analysis data during a crash, exception, or critical error or by performing a snapshot. You can use the Diagnostic Data Handler with Pipeline Manager.

When the Diagnostic Data Handler collects data, it creates a text file that includes information obtained from instrumentation probes. The information includes:

- The stack trace of the Pipeline Manager process.
- Log file entries from the pipeline log circular buffers. See "Getting Recent Pipeline Log File Entries".
- Diagnostic and performance data collected by Operations Management Framework (OMF) probes. The relevant data includes:
  - HTTP server
  - SNMP
  - Real-time pipeline statistics
  - AAA Gateway Manager statistics
  - EDR statistics
  See "BRM OMF Instrumented Objects".

You specify the diagnostic file name by editing the `DiagnosticFileName` registry entry. If an existing diagnostic file exists, it is renamed to include the process ID and the date and time that the new file was created. For example:

`diagnostic.dat.3418.20060824_113734`

You can manually remove old files as needed. You should archive the data files regularly.

By default, diagnostic files are stored in `Pipeline_Home/log`.

The `DiagnosticDataHandler` entry is a top-level section in the registry:

```plaintext
DiagnosticDataHandler
{
    DiagnosticFilePath = ./log
    DiagnosticFileName = diagnostic.dat
}
```

For more information, see "Diagnostic Data Handler" in BRM Configuring Pipeline Rating and Discounting.

Getting a Snapshot from the Diagnostic Data Handler

You can get a snapshot by using the `snmpset` command to set the `createSnapshot` probe value to `True`. When you get a snapshot, a diagnostic file is created.

---

**Important:** The `piperev` utility does not display information about the uninstallation of any pipeline component. Only installation information is displayed.
About Operations Management Framework

Operations Management Framework (OMF) provides a framework for monitoring and controlling the BRM system. OMF is implemented for the following components:

- Pipeline Manager, including batch pipeline, real-time pipeline, and AAA Gateway pipeline.
- Multi-threaded framework (MTF).

OMF includes the following components:

- **Instrumentation Probe API.** This component includes *probes* that gather system data and can control processes.
  
  A probe is part of the code for a component. It collects and sets instrumentation data about that component. For example:
  
  - A reference object cache (ROC) synchronization probe can collect information about pre-commits, post-commits, and rollbacks.
  - A Diagnostic Data Handler probe can create a snapshot of the Pipeline Manager system.

- **Instrumentation Probe Broker.** This component provides data from the Instrumentation Probe API to the instrumentation protocols. The Instrumentation Probe Broker runs as part of a Pipeline Manager instance.

- **Instrumentation Protocol Plugins.** These components provide an interface for client tools and Web browsers. There are two protocols:
  
  - **SNMP.** This protocol uses the SNMP daemon to provide instrumentation data to client tools and HP OpenView and to diagnostic tools such as the Pipeline Manager Diagnostic Data Handler.
    
    You can use SNMP utilities to get and set instrumentation data. To do so, you find the instrumentation object IDs (OIDs) in the MIB.
    
    For more information, see "Using the SNMP Instrumentation Protocol to Monitor and Control BRM Components".
  
  - **HTTP.** This protocol uses a Web interface to provide instrumentation data to Web applications. BRM includes a default XML style sheet that you can customize to display selected information.
    
    See "Using the HTTP Instrumentation Protocol to Read OMF Instrumentation Data".

**Note:** The Instrumentation Protocol Plugins have their own probes (for example, to record the number of SNMP GET operations).

Typically, SNMP is used for monitoring the system, and the HTTP interface is for more interactive use.

Figure 2–1 shows the OMF components:
About Probe Data

Probes can handle data in the following ways:

- Attribute probe. This is the simplest form of data, consisting of a name/value pair. For example:
  
  Number of Startup Threads: 20

- Group probe. This is a list of name/value pairs about a related process. For example:
  
  Thread Info:
  Pool Type Round Robin
  Thread Count 10

- Table probe. This is a list of groups. For example:
  
  DMO Server Configuration:
  Name Host Name Port Number
  DMO Server 1 dmo1.corp.com 13093
  DMO Server 2 dmo2.corp.com 13093

- BigTable probe. This returns large amounts of data (for example, the contents of a log file).

  BigTable probes are not supported by SNMP. Therefore, you can only display data from them by using the HTTP protocol.

Using the SNMP Instrumentation Protocol to Monitor and Control BRM Components

SNMP (Simple Network Management Protocol) is a widely-used protocol for monitoring network equipment, computer equipment, and other devices.

When SNMP is configured, you can use SNMP utilities to get and set instrumented data. BRM includes the AGENT++ SNMP utilities. See "SNMP Utilities".
Important
You can use SNMP utilities other than AGENT++ (for example, NetSNMP). If you use the AGENT++ SNMP utilities, you cannot use symbolic SNMP names in SNMP commands. For example, instead of using the following command:

```plaintext
snmpWalk 10.196.129.31
portal.components.mtf.connectionConfigurations.dmoTable.dmoEntry -P20761 -S
```

Use the following command:

```plaintext
snmpWalk 10.196.129.31 1.3.6.1.4.1.3512.1.5.2.2.1 -P20761 -S
```

About the SNMP Components
The SNMP architecture consists of one master agent per machine, and multiple sub-agents. Each sub-agent reads a dynamic MIB file to support changes to the objects being monitored. The sub-agents use the AgentX protocol to communicate with the master agent. The SNMP clients use the SNMP protocol to communicate with the master agent.

Figure 2–2 shows the SNMP components.

![SNMP Components Diagram](image)

Installing SNMP
Install the BRM SNMP package before installing the BRM Third-Party Software package. The SNMP package includes the following:

- AGENT++ SNMP server and configuration files.
- AGENT++ SNMP client tools.
- The base BRM MIB file.
Configuring SNMP Components

The SNMP instrumentation protocol runs as an SNMP sub-agent. SNMP sub-agents can read from and write to instrumented objects defined in the MIB. Sub-agents connect to the master agent by using the AgentX sub-agent protocol. For information on configuring and running SNMP sub-agents, see the SNMP documentation.

To start the SNMP master agent, use the following command:

```
master_agent -l Master_agent_port -x AgentX_port &
```

where `AgentX_port` must be the same as defined in the registry file.

For more information, see the SNMP documentation.

Problems Using SNMP on Oracle Solaris 10

If you are unable to do any of the following

- Run SNMP utilities
- Start AAA Gateway Manager when `SnmpServer` entry is enabled in the registry file
- Send SNMP requests to AAA Gateway Manager for rebalancing Connection Managers

Complete the following steps to solve the problem:

1. Open the `etc/system` file using a text editor.
2. Add the following entries:
   ```
   set ip:do_tcp_fusion=0x0
   set ip:tcp_fusion_rcv_unread_min=0
   ```
3. Save and close the file.
4. Restart the system.

Enabling SNMP Instrumentation Data Collection

To enable or disable instrumentation data collection, use the `Instrumentation` section in the Pipeline Manager registry file. If you enable these entries, the SNMP sub-agent starts when Pipeline Manager starts.

**Important:** You can enable instrumentation data collection in any Pipeline Manager instance, including those for rating, running AAA Gateway, and rerating.

The `Instrumentation` section includes the following entries:

- Use the `ProbeBroker` section to point to the directory that contains probe information files. The default is `Pipeline_Home/instrumentation`.
- Use the `SnmpServer` entry to configure the SNMP protocol.
- Use the `Port` entry to define the SNMP AgentX port number.

**Important:** The port must be the port you assigned when you configured the SNMP master agent and sub-agent.
- Use the **WaitTimeout** entry to define how long to wait, in milliseconds, before reconnecting to the master agent and re-initializing the MIB.

  The default is 10 milliseconds.

- Use the **ProcessDescription** entry to provide a name for the process being monitored. You see the name when you run SNMP commands, for example, and SNMP walk. Providing different names is helpful when you run more than one of the same type of process on a single host.

  The default is:

  

  user:process_name:registry_file

  where:

  - `user` is the name of the user who ran the process.
  - `process_name` is the name of the process.
  - `registry_file` is the name of the registry file that configured the process.

  The following is a sample **Instrumentation** section:

  ```
  Instrumentation
  {
  ProbeBroker
  {
  ProbeInfoFilePath = ./instrumentation
  }
  SnmpServer
  {
  Port = 11960
  ProcessDescription = ifw
  WaitTimeout = 10
  }
  HttpServer
  {
  Port = 12019
  StyleSheetFile = ./instrumentation/portal_omf.xsl
  PortalLogoFile = ./instrumentation/portal_omf.gif
  }
  }
  ```

---

**Important:** If you use SNMP monitoring in a Pipeline Manager instance, stop all monitoring requests to Pipeline Manager before you stop it. To stop the monitoring requests, stop the master SNMP agent. You can use the `kill` command. For example:

```
kill -9 master_agent_pid
```
**About Dynamic Object IDs**

You can run multiple instances of the same type of process on a single host. Therefore, these separate processes must be identified by SNMP. To do so, BRM creates object IDs (OIDs) dynamically by using:

- The base OID from the MIB.
- A process ID from a process table.
- A registry ID from a registry table.
- For table probes only, an instance ID.

The OID for a probe uses the following format:

```
1.3.6.1.4.1.3512.1.component_id.module_id.1.probe_id.process_id.registry_id.instance_id
```

To use SNMP to access the probe value, you must find the process ID, registry ID, and instance ID. For example, to find the OID for a specific process `batchSizeLimit` entry:

1. See "BRM OMF Instrumented Objects" to find the name of the probe.
2. To find the base OID, look in the MIB file or in "BRM OMF Instrumented Objects". For example:
   
   ```
   1.3.6.1.4.1.3512.1.2.1.1.1.1
   ```

3. Use the `snmpWalk` command on the Process table to find the process ID for the component you want to find the value for.
   
   The OID for an entry in the process table is:
   
   ```
   1.3.6.1.4.1.3512.1.101.1
   ```

   The `snmpWalk` command is:

   ```
   snmpWalk host_name 1.3.6.1.4.1.3512.1.101.1 -Pport -S
   ```

   For example:

   ```
   snmpWalk frisco 1.3.6.1.4.1.3512.1.101.1 -P44293 -S
   ```

   The results show:

   ```
   1.3.6.1.4.1.3512.1.101.1.1.1 = 1
   ```

   The process ID is 1.

4. Use the `snmpWalk` command on the registry table to find the registry ID.
   
   The OID for an entry in the registry table is:
   
   ```
   1.3.6.1.4.1.3512.1.102.1
   ```

   So the `snmpWalk` command is:

   ```
   snmpWalk host_name 1.3.6.1.4.1.3512.1.102.1 -Pport -S
   ```

5. Find the instance ID.
   
   For all probes that are not in a probe table, the instance ID is 0. In this case, the MIB file shows that `batchSizeEntry` is a group probe:

   ```
   transactionManagerGroupEntry ::= SEQUENCE {
     batchSizeLimit  Integer32,
     loggingOff     DisplayString,
   }
   ```
The instance ID is 0.

The OID for this instance of `batchSizeEntry` is:

```
1.3.6.1.4.1.3512.1.2.1.1.1.1.18.0
```

You can use that OID to get or change the value. For example, to get the value:

```
snmpGet frisco 1.3.6.1.4.1.3512.1.2.1.1.1.1.18.0 -P44293
```

**Tip:** To get all available probe values in the process, you can run the `snmpWalk` command at the top level of the MIB structure. For example:

```
snmpWalk frisco 1.3.6.1.4.1.3512.1 -P44293 -S
```

For more information about using probe OIDs to get data, see "About Probe Data" and "Getting and Setting Instrumentation by Using SNMP".

### About Instance IDs for Table Probes

The instance ID for a table probe is the row number in the table. For example, the `numSnmpGetRequests` probe is part of a table that shows SNMP requests as shown in Table 2–10:

```
An `snmpWalk` command on the `numSnmpGetRequests` probe gives these results:

```
1.3.6.1.4.1.3512.1.4.2.1.1.6.5.1 = 2
1.3.6.1.4.1.3512.1.4.2.1.1.6.5.2 = 98
1.3.6.1.4.1.3512.1.4.2.1.1.6.5.3 = 6
1.3.6.1.4.1.3512.1.4.2.1.1.6.5.4 = 20
1.3.6.1.4.1.3512.1.4.2.1.1.6.5.5 = 8
1.3.6.1.4.1.3512.1.4.2.1.1.6.5.6 = 1
1.3.6.1.4.1.3512.1.4.2.1.1.6.5.7 = 6
1.3.6.1.4.1.3512.1.4.2.1.1.6.5.8 = 6
1.3.6.1.4.1.3512.1.4.2.1.1.6.5.9 = 6
1.3.6.1.4.1.3512.1.4.2.1.1.6.5.10 = 14
1.3.6.1.4.1.3512.1.4.2.1.1.6.5.11 = 12
1.3.6.1.4.1.3512.1.4.2.1.1.6.5.12 = 5
```

The last number in the OID is the instance ID, which corresponds with the table row, 1-4. The value shown in the `Number of GET Requests` column is the probe value.

### About the Process Table

The process table uses these OIDs:

```
processTable (101)
processEntry (1)
```
Using the SNMP Instrumentation Protocol to Monitor and Control BRM Components

Monitoring and Maintaining Your BRM System

processIndex (1)
processDescr (2)

- **processIndex** is the process ID assigned in the table.
- **processDescr** is the description of the process (for example, ifw). This description is defined in the Instrumentation section of the registry file. See "Enabling SNMP Instrumentation Data Collection".

For example, an SNMP walk could give these results, where the process ID is 1 and the process description is ifw:

\[
1.3.6.1.4.1.3512.1.101.1.1.1 = 1 \\
1.3.6.1.4.1.3512.1.101.1.2.1 = ifw
\]

**Note:** The last number in the OID is the row number, which is the same as the process ID.

Each process can have multiple registry settings, each of which needs its unique ID. Therefore, these registered objects are identified in the registry table.

**About the Registry Table**

The registry table uses these OIDs:

registryTable (102)
  registryEntry (1)
    registryIndex (1)
    registryName (2)

- **registryIndex** is the ID assigned in the table.
- **registryName** is the name used in the registry file (for example, ifw or ifw.Pipelines.ALL_RATE).

For example, an SNMP walk could give these results:

\[
1.3.6.1.4.1.3512.1.102.1.1.1.2 = 2 \\
1.3.6.1.4.1.3512.1.102.1.2.1.2 = ifw.SignalHandler
\]

**Note:** The last number in the OID is the row number, which is the same as the ID.

**Getting and Setting Instrumentation by Using SNMP**

To get and set instrumentation data, use the SNMP tools installed in `BRM_Home/bin`. The following SNMP utilities are included:

- snmpBulk
- snmpDiscover
- snmpGet
- snmpNext
- snmpNextAsync
- snmpPasswd
- snmpSet
Using the HTTP Instrumentation Protocol to Read OMF Instrumentation Data

- snmpWalk
- snmpWalkThreads

Sample SNMP Input and Output

This section presents sample input and output for the `snmpGet`, `snmpSet`, and `snmpWalk` commands.

**snmpGet**

In the following sample, the master agent is on balrog/28093:

```
$ snmpGet sampleserver 1.3.6.1.4.1.3512.1.1.1.1.1.36.3.0 -P12345
SNMP++ Get to sampleserver SNMPV1 Retries=1 Timeout=1000ms Community=public
Oid = 1.3.6.1.4.1.3512.1.1.1.1.1.36.3.0
Value = Startup complete
```

**snmpSet**

In the following sample, the master agent is on sampleserver/12345:

```
snmpSet sampleserver 1.3.6.1.4.1.3512.1.1.1.1.1.32.7.0 -P12345
SNMP++ Set to sampleserver SNMPV1 Retries=1 Timeout=1000ms SET-community=public
GET-community=public
Oid = 1.3.6.1.4.1.3512.1.1.1.1.1.32.7.0
Current Value = << WRITE-ONLY PROBE >>
Value Type is Octet String
Please enter new value: yes
Set Status = Success
```

MTF / Version R2 10092 stopped at 24.08.2007 13:35:04

**snmpWalk**

In the following sample, the master agent is on sampleserver/12345:

```
snmpWalk sampleserver 1.3.6.1.4.1.3512.1.1.1.1.1.1.1.35.0 -P12345 -S
SNMP++ snmpWalk to kabini2 SNMPV1 Retries=1 Timeout=1000ms Community=public
1.3.6.1.4.1.3512.1.1.1.1.1.1.1.35.0 = true
1.3.6.1.4.1.3512.1.1.1.1.1.1.1.36.2.0 = Normal
1.3.6.1.4.1.3512.1.1.1.1.1.1.1.36.3.0 = Startup complete
1.3.6.1.4.1.3512.1.1.1.1.1.1.1.36.4.0 = true
1.3.6.1.4.1.3512.1.1.1.1.1.1.1.36.5.0 = 10
1.3.6.1.4.1.3512.1.1.1.1.1.1.1.36.6.0 = 5
1.3.6.1.4.1.3512.1.1.1.1.1.1.1.36.7.0 = 5
1.3.6.1.4.1.3512.1.1.1.1.1.1.1.36.8.0 = 0
1.3.6.1.4.1.3512.1.1.1.1.1.1.1.36.9.0 = 20
1.3.6.1.4.1.3512.1.1.1.1.1.1.1.36.10.0 = DEFAULT, REALTIME, DEFAULT_INACTIVE, REALTIME_INACTIVE
End of SUBTREE Reached
Total # of Requests = 11
Total # of Objects = 10
```

Using the HTTP Instrumentation Protocol to Read OMF Instrumentation Data

To get instrumentation data from the HTTP server, you configure the HTTP server in the Pipeline Manager registry files. You then send the URL for the data you want to retrieve.
Using the HTTP Instrumentation Protocol to Read OMF Instrumentation Data

**Note:** You can read instrumentation data from the HTTP server, but you cannot set it.

**Figure 2–3** shows the HTTP display:

**Figure 2–3 HTTP Instrumentation Display**

<table>
<thead>
<tr>
<th>Last Update Requested at:</th>
<th>2006/06/19 20:43:33</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td><strong>Running</strong></td>
</tr>
</tbody>
</table>

**Instrumented objects:**

| Refresh ifw               |
| DataPool                  |
| DiagnosticDataHandler     |
| Instrumentation            |
| HttpServer                |
| ProbeBroker               |
| ImsmpServer               |
| LogMessageTable           |
| Pipelines                 |
| TestPipeline              |
| Description               |
| EdrFactory                |
| Functions                 |
| Standard                  |
| FunctionPool              |
| Input                     |
| InputModule               |
| Module                    |

**Throughput Statistics:**

- **Total EDR Count (real-time):** 120
- **Total EDR Count (after transaction ended):** 120
- **Accumulated Txn Processing Time (sec):** 3
- **Total Txn Count:** 40

**Last Throughputs:**

<table>
<thead>
<tr>
<th>Index</th>
<th>Throughput (edrs/sec)</th>
<th>Timestamp</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>16.06.2006 11:49:21</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>16.06.2006 11:49:21</td>
</tr>
</tbody>
</table>

**Enabling HTTP Display of Instrumentation Data**

To enable or disable instrumentation data collection, use the **Instrumentation** section in the Pipeline Manager registry files. If you enable these entries, the HTTP server starts in a thread when Pipeline Manager starts.

**Important:** You can enable instrumentation data collection in any Pipeline Manager instance, including those for running AAA Gateway, real-time rating, and rerating.

The **Instrumentation** section includes the following entries:

- Use the **ProbeBroker** section to point to the directory that contains probe information files. The default is Pipeline_Home/instrumentation.
- Use the **HttpServer** section to configure the HTTP protocol.
  - Use the **Port** entry to specify the port number for the HTTP server.
  - Use the **StyleSheetFile** and **PortalLogoFile** entries to specify the path to the XML style sheet and the logo displayed in the Web interface.
The following is a sample **Instrumentation** section:

```
Instrumentation
{
  ProbeBroker
  {
    ProbeInfoFilePath = ./instrumentation
  }
  SnmpServer
  {
    Port = 11960
    ProcessDescription = ifw
  }
  HttpServer
  {
    Port = 12019
    StyleSheetFile = ./instrumentation/portal_omf.xsl
    PortalLogoFile = ./instrumentation/portal_omf.gif
  }
}
```

---

**Important:** If you use HTTP monitoring in a Pipeline Manager instance, stop all monitoring requests to Pipeline Manager before you stop it. To stop the monitoring requests, stop the master SNMP agent. You can use the `kill` command. For example:

```
kill -9 master_agent_pid
```

---

**Displaying Instrumentation Data in a Web Browser**

To display instrumentation data in a Web browser, use a URL with this format:

```
http://host_name:port/registry_entry
```

- The host name and port number are those configured in the registry file.
- The registry entry is an entry from the Pipeline Manager registry files.

For example, to get all instrumentation from Pipeline Manager, use `ifw`, the top-level registry entry:

```
http://host_name:port/ifw
```

**Customizing the Data Displayed by the HTTP Instrumentation Protocol**

BRM includes a default style sheet to display HTTP instrumentation data. You can customize the style sheet and logo to display selected instrumentation data.

The default style sheet is `portal_omf.xsl`. It is installed in `BRM_Home/instrumentation`. The default logo is `portal_omf.gif`. It is installed in `BRM_Home/instrumentation`.
Using Configuration Files to Connect and Configure Components

This document provides an overview of Oracle Communications Billing and Revenue Management (BRM) configuration and properties files, including where to find them and how to edit them.

For information about a specific entry in a configuration file, look directly in the file. Each entry includes a description and guidelines for changing the values.

For information about the optimal values for tuning your BRM system, see “Improving BRM Performance”.

About Configuration and Properties Files

The primary purpose of configuration and properties files is to enable the different components of BRM to communicate with each other (see “About Connecting BRM Components”). The configuration and properties files can also include other entries that let you increase performance and implement business policies.

- Most BRM components and utilities use configuration files (`pin.conf`).
- BRM Java programs use properties files (usually `Infranet.properties`).
- BRM programs based on Perl scripts read configuration and properties information from a file named `pin.conf.pl`.

You can use any text editor to edit configuration files.

---

**Important:** Before you edit a configuration file, save a backup copy.

---

Some configuration files are write-protected. Before editing the file, remove that restriction. After you edit the file, restore the restriction.

Each configuration file includes specific, detailed information about how to edit each configuration entry in that file. To edit an entry, follow the guidelines provided for that entry.

To insert a comment, type a crosshatch (#) followed by the comment. BRM ignores all text on that line.
Configuration Entry Syntax

Any configuration entry that includes an application name, such as Infranet.pricing_tool.log.file, is specific to that application. Any other entry is a shared entry, applying to all applications. Any application-specific entry overrides a shared entry.

Syntax for Configuration Entries

Entries in a configuration file use this syntax:

```
host_name   program   keyword   values
```

where:

- **host_name** is generally a single hyphen (-), which refers to the current computer. If several computers share the same configuration file, use the name of the current computer. In this case, BRM components, such as the Connection Manager (CM) or the Data Manager (DM), use only the entries that contain the host name on which they are started. You can use the name or IP address as `host_name`.

- **program** is the name of the program to which this entry applies. The program can be:
  - The name of the application (or custom application) or Facilities Module (FM), such as `cm`, `pin_bill`, or `fm_bill`. Use a specific name when the entry applies only to a single program.
  - `nap` (Network Application Program). Use `nap` when the entry applies to general BRM applications, which use the PCM_CONNECT and PCM_CONNECT_OPEN functions.
  - Blank or a single hyphen (-). The entry applies to any BRM function, such as `pin_virtual_time`.

- **keyword** is the name of the configuration entry.

- **values** is one or more parameters specific to the configuration entry. Values are separated by spaces.

A single configuration entry resembles this example:

```
-cm userid 0.0.0.1 /service 1
```

This entry applies to the Connection Manager (cm) on the local computer (-). The entry is called `userid`, and the three values associated with that entry are `0.0.0.1`, `/service`, and `1`.

---

**Note:** Some configuration files have entries with `userid` and a database, as shown here. BRM ignores the database portion of these entries:

```
-cm userid 0.0.0.1 /service 1
```

---

Syntax for Facilities Module (FM) Entries

The CM configuration file includes entries for Facilities Module (FM) that are linked to the CM at startup. Some of these entries are for the base set of FMs that are part of the standard release; other entries are for optional BRM components. You can also add entries for custom FMs.

Configuration entries for FMs use this syntax:
- cm fm_module file_name initialization_table initialization_function tag

where:

- **file_name** is the path to the shared library file containing the functions that make up the FM. The file name has the following platform-dependent extensions:
  - .so on HP-UX IA64, Linux, and Solaris
  - .a on AIX
  
  Do not change this parameter unless you change the location of the file.

- **initialization_table** is the name of the configuration structure that maps each opcode to a function. Do not change this text for standard FMs.

- **initialization_function** is either a hyphen (-), meaning that no function is run when the CM is started, or the name of the function to be run at startup. Some FMs call optional initialization functions that you can use to configure the FM.

- **tag** identifies the FM to the CM. Each CM has an equivalent tag as part of the cm_ports configuration entry. Each FM with a matching tag is linked to that CM at startup. The default tag for the base set of FMs is *pin*. For example, you can use other tags to define separate sets of FMs for multiple CMs on the same computer.

Configuration entries for the base-rating FM resemble this example:

- cm fm_module ../../lib/fm_rate_pol.so fm_rate_pol_config - pin

The entry shows a policy FM that must always be included with its corresponding base FM.

---

**Caution:** Some FMs depend on others. Never change the order of the base set of FMs in the CM configuration file.

---

### Preparing for Platform Migration by Using Variables in pin.conf Files

You can reference certain system environment variables in pin.conf configuration files. These references can facilitate future migration of the pin.conf files to BRM implementations on other platforms.

For information about environment variables, see "BRM Environment Variables" in BRM Installation Guide.

Table 3–1 shows the environment variables that can be referenced in BRM configuration files (pin.conf):

<table>
<thead>
<tr>
<th>Environment variable</th>
<th>Reference in pin.conf files</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIN_HOME</td>
<td><code>${PIN_HOME}</code></td>
</tr>
<tr>
<td>PIN_LOG_DIR</td>
<td><code>${PIN_LOG_DIR}</code></td>
</tr>
<tr>
<td>LIBRARYEXTENSION</td>
<td><code>${LIBRARYEXTENSION}</code></td>
</tr>
<tr>
<td>LIBRARYPREFIX</td>
<td><code>${LIBRARYPREFIX}</code></td>
</tr>
</tbody>
</table>

Sample pin.conf file with environment variable references:

- - pin_virtual_time `${PIN_HOME}`/lib/pin_virtual_time_file
- fm_rate tax_supplier_map `${PIN_HOME}`/sys/cm/tax_supplier.map
- cm fm_module `${PIN_HOME}`/lib/fm_utils/${LIBRARYEXTENSION} fm_utils_config fm_
Locations of Configuration and Properties Files

Each daemon can have its own configuration file, or two or more can share a configuration file.

File Locations

The default location for a configuration file is the directory where the system process or program runs. Typical locations are:

- Directories inside the `BRM_Home/sys` directory; for example, `BRM_Home/sys/cm/pin.conf`.
- Directories inside the `BRM_Home/apps` directory; for example, `BRM_Home/apps/pin_billd/pin.conf`.
- In the application folder; for example, `BRM_Home/Application/Home/Infranet.properties`.

Configuring a Shared `pin.conf` File

If you run several BRM applications and processes on one computer, they can share a single configuration file. To set up a shared configuration file:

1. Combine configuration entries for each BRM component into a single `pin.conf` file.
2. Save that file to the `BRM_Home` directory.
3. For each BRM component that uses the shared configuration file, move its specific configuration file to a backup location or rename the file.

When BRM starts any BRM application, component, or process, it searches for the appropriate `pin.conf` file in the following directories in the order shown:

1. The current directory.
2. The system `/etc` directory.
3. If the `PIN_HOME` environment variable is defined, the `BRM_Home/config` directory.

Guidelines for Editing Java Properties Files

Java applications get configuration information from Java properties files instead of the `pin.conf` files that are used for C applications.

The BRM installation program uses information supplied by the installer to write configuration information to the properties files.
Each properties file includes specific, detailed information about how to edit each configuration entry in that file. To edit an entry, follow the guidelines provided with that entry.

You can add comments to properties files to help others understand the purpose of your entries. To insert a comment, type a crosshatch (#) followed by the comment. BRM ignores all text on the same line after the crosshatch.

Common Properties File Entry Syntax

Connection entries, failover entries, and other entries each have their own syntax considerations.

Connection Entry

The connection entry consists of a full URL to the BRM services. It takes one of two forms, depending on the type of login setting:

- For the Type 1 login setting, which requires a password, use this format:

  \texttt{pcp://user_name:password@host_name:port/service_object}

  where:
  - \texttt{user_name} is the login name to use for connecting to BRM.
  - \texttt{password} is the password for the specified user name.
  - \texttt{host_name} is the name or IP address of the computer running the CM or Connection Manager Master Process (CMMP).
  - \texttt{port} is the TCP port number of the CM or CMMP on the host computer. The port number must match the corresponding \texttt{cm\_ports} entry in the CM or CMMP configuration file.
  - \texttt{service_object} is the service type. The trailing number, “1,” is the Portal object ID (POID) of the service.

  For example:

  \texttt{infranet.connection=pcp://root.0.0.0.1:password@hostname:11960/service/admin\_client}

- For the Type 0 login setting, which does not require a password, use this format:

  \texttt{pcp://host_name:port/database_number/service_object}

  where:
  - \texttt{host_name} is the name or IP address of the computer running the CM or Connection Manager Master Process (CMMP).
  - \texttt{port} is the TCP port number of the CM or CMMP on the host computer. The port number must match the corresponding \texttt{cm\_ports} entry in the CM or CMMP configuration file.
  - \texttt{database_number} is the database number assigned to your BRM database when the DM was installed. For example, 0.0.0.1.
  - \texttt{service_object} is the service type. The trailing number, “1,” is the Portal object ID (POID) of the service.

  For example:

  \texttt{infranet.connection=pcp://hostname:11960/0.0.0.1/service/admin\_client}
Failover Entry
A failover entry refers to an alternate CM host that an application can use to connect to BRM if the main host, specified in the connection entry, is unavailable.

For example:

```
infranet.failover.1=pcp://host_name:port
```

where:

- ```host_name``` is the name or IP address of the computer running the CM or CMMP.
- ```port``` is the TCP port number of the CM or CMMP on the host computer. The port number must match the corresponding ```cm_ports``` entry in the CM or CMMP configuration file.

**Note:** ```user_name, password, and service_object``` for the alternative hosts are the same as for the main host and are not specified in failover entries.

Other Properties Entries
The flags used in the connection entry of the main `Infranet.properties` file are used by all the other properties entries, unless they are overridden.

Other entries that override these values for all your Java applications use this form:

```
infranet.entry_specific_entries
```

The `Infranet.properties` file also contains entries specific to particular Java applications, in this form:

```
infranet.application_specific_entries
```

About Validating XML Configuration Files
After you edit the contents of an XML configuration file, a load utility typically validates the contents against the file’s schema definition. If the contents do not conform to the schema definition, the utility returns an error or the load operation fails.

XML files are not directly linked to their schema definition files. Instead, they are linked to one of the following XSD reference files:

- `BRM_Home/apps/pin_billd/business_configuration.xsd`
- `BRM_Home/sys/data/config/business_configuration.xsd`

The XSD reference file associates multiple XML files with their schema definition file. Each XML file contains a schema location pointer to the reference file, and the reference file contains a pointer to the XML file’s schema definition. Figure 3–1 shows an example.
To validate the contents of XML_1, XML_2, or XML_3 in the preceding figure, business_configuration.xsd is called rather than XSD_1, XSD_2, or XSD_3.

### About Connecting BRM Components

To allow BRM components to communicate with each other, you use entries in configuration or properties files. The basic connection entries in the files identify the host names and port numbers of each component.

These connection entries are set when you install BRM and when you install each client application. You can change them if you change your configuration. Depending on how you install BRM, you might have to change some entries to connect BRM components. See "Reconfiguring a BRM Component".

Figure 3–2 shows how entries in configuration files link components.
In this figure, the client application is a utility that uses a configuration file entry to point to the CM. (A Java client has a similar entry in its properties file.) This sample entry includes the CM host name, **CM_System**, and port number, **11960**:

```
cm_ports ip CM_System 11960
```

The CM configuration file has corresponding entries:

- **The cm_name entry identifies the host name as CM_System**
  ```
  - cm cm_name CM_System
  ```

- **The cm_ports entry identifies the port number as 11960**:
  ```
  - cm cm_ports 11960 pin
  ```

The CM configuration file includes an entry that points to the DM. This entry includes the DM host name and port number:

```
dm_pointer 0.0.0.1 ip DM_System 11950
```
The DM configuration file includes corresponding information:

- **The `dm_name` entry** identifies the host name as `DM_System` this entry is optional; if you remove or disable it, BRM uses `gethostname` to find the IP address of the DM computer:
  - `dm dm_name DM_System`

- **The `dm_port` entry** identifies the port number as **11950**:
  - `dm dm_port 11950`

The DM configuration file specifies the database number, which is in `0.0.n.n / 0` format:
- `dm dm_db_no 0.0.0.1 / 0`

There are additional entries in configuration files, but these entries are the most basic. For more information on connection entries in configuration files, see the comments in the configuration files.

### Guidelines for Database and Port-Number Entries

The configuration and properties files specify identifying numbers for databases and DMs. The default numbers, listed in Table 3–2, are systematic to make numbers relatively easy to maintain and extend.

The DM numbers are in the form `A.B.C.D`. You can make the number assignments meaningful by using:

- **A** to separate divisions of your company, or **0** if you have none.
- **B** to distinguish different BRM installations, or **0** if you have only one.
- **C** to indicate the type of Data Manager; for example:
  - **0** for data
  - **1** for transaction processing of credit or debit card
  - **2** for email
  - **3** for taxation

  **Important:** Start numbering your custom DMs at **100**, such as **0.0.100.1**.

- **D** to indicate the instance of a particular DM type, division, or installation.

### Table 3–2 Database and Port-Number Entries

<table>
<thead>
<tr>
<th>Program</th>
<th>Database number</th>
<th>Port number</th>
</tr>
</thead>
<tbody>
<tr>
<td>The database DM (Oracle <code>dm_oracle</code>)</td>
<td>0.0.0.1</td>
<td>12950</td>
</tr>
<tr>
<td>The second database DM for a multidatabase system (<code>dm_oracle</code>)</td>
<td>0.0.0.2</td>
<td>12951</td>
</tr>
<tr>
<td>Each additional <code>dm_oracle</code> for multidatabase systems</td>
<td>0.0.0.n</td>
<td>12950 + (n - 1)</td>
</tr>
<tr>
<td>Paymentech data manager for credit card, <code>dm_fusa</code></td>
<td>0.0.1.1</td>
<td>12810</td>
</tr>
<tr>
<td>Paymentech data manager for direct debit, <code>dm_fusa</code></td>
<td>0.0.1.2</td>
<td>12810</td>
</tr>
</tbody>
</table>
Setting Data Manager Attributes

In the CM configuration file, you can set DM attributes (dm_attributes) for each DM to which you connect. For example, set the scoped and assign_account_obj options when you enable branding. You use the dm_attributes entry with these options:

- **scoped**: Enables scoping enforcement. This option is used only when you use branding. It restricts database access for separate brands. You can disable this option if branding is not used. See "Data Scoping" in BRM Developer’s Guide.

- **assign_account_obj**: Assigns an owning account reference when the object is created. When you use branding, all objects are associated with an account. You can disable this option if branding is not used. See "Changing the Owner of a Discount Sharing Group" in BRM Managing Accounts Receivable.

- **searchable**: Restricts access to certain databases in multiple database environments and indicates that the DM is a database DM. See "Running the pin_multidb.pl Script on the Primary Installation Machine" in BRM Installation Guide.

This example specifies a DM pointer in the CM pin.conf file:

```
- cm dm_pointer 0.0.0.1 ip 156.151.2.168 33950 # dm_oracle
- cm dm_attributes 0.0.0.1 scoped,assign_account_obj,searchable
- cm dm_pointer 0.0.0.2 ip 156.151.2.168 34950 # dm_oracle
- cm dm_attributes 0.0.0.2 scoped,assign_account_obj,searchable
```
Connecting a Data Manager to the BRM Database

Use the following DM `pin.conf` file entries in Table 3–3 to connect a DM to the BRM database. These entries are used by multiple DMs, such as DM Oracle and the Account Synchronization DM.

### Table 3–3  DM `pin.conf` File Entries

<table>
<thead>
<tr>
<th>Entry</th>
<th>Description</th>
</tr>
</thead>
</table>
| sm_database | Specifies the database alias name. For example, for Oracle this is the SQL*NET alias defined in the `tnsnames.ora` file. This entry was configured when you installed BRM, so you do not have to change it.  
**Note:** If you have multiple database hosts, such as an Oracle Parallel Server configuration, include a separate `sm_database` configuration entry for each host. |
| sm_id | Specifies the database user name that the DM uses to log in to the BRM database. This entry was configured when you installed BRM, but you can change it. |

Creating Configuration Files for BRM Utilities

Some BRM utilities, such as `load_tax_supplier`, require you to create a configuration file to tell the utility how to connect to the BRM system. The configuration file must be in the same directory as the utility executable file.

To create a configuration file for a utility:

1. Copy the sample configuration file in `BRM_Home/source/apps/sample`.
   
   Use this file, which contains all of the configuration entries needed for connecting to BRM, as a template for any utility configuration file.

2. Edit the configuration entries to reflect your BRM environment. Follow the guidelines in the configuration file.

3. Save the file as `pin.conf` in the directory with the utility executable file.

Table 3–4 shows the common utility `pin.conf` entries:

### Table 3–4  Common Utility `pin.conf` Entries

<table>
<thead>
<tr>
<th>Entry</th>
<th>Description</th>
</tr>
</thead>
</table>
| cm_ports | Specifies a pointer to the CM or CMMPP.  
Use a separate entry for each CM or CMMPP.  
Each entry includes three values:  
  - `protocol: ip`  
  - `host`: the name or IP address of the computer running the CM or CMMPP  
  - `port`: the port number of the CM or CMMPP on this computer  
  
  The port number should match a corresponding `cm_ports` entry with the same port number in the CM or CMMPP configuration file. The default, 11960, is a commonly specified port for the CM or CMMPP.  
  
  See "About Connecting BRM Components". |
| login_name | Specifies the login name to use when connecting to the CM. |
In addition, the following `pin.conf` entries in Table 3–5 are used by multithreaded application (MTA) utilities:

### Table 3–5  MTA Utilities pin.conf Entries

<table>
<thead>
<tr>
<th>Entry</th>
<th>Description</th>
</tr>
</thead>
</table>
| children | Specifies the number of worker threads spawned to perform the specified work. The default is 5. **Important:** This entry is mandatory.  
For more information, see:  
- Tuning the Number of Children for Billing Utilities  
- Setting the Number of Children for Invoice Utilities  
- Controlling Thread Load on Your Multithreaded Application. |
| fetch_size | Specifies the number of objects received from the database in a block and cached in system memory for processing. The default is 5000. **Important:** This entry is mandatory.  
See:  
- Tuning the Account Cache Size for Billing Utilities (fetch_size)  
- Tuning the Account Cache Size for Invoice Utilities (fetch_size) |
| hotlist | Specifies the name for the hotlist file. This parameter is available for backward compatibility. |
| logfile | Specifies the file name used to log errors.  
**Important:** This entry is mandatory.  
See "Changing the Name or Location of a Log File". |
| loglevel | Error reporting level.  
- 0: no logging  
- 1: (Default) log error messages only  
- 2: log error messages and warnings  
- 3: log error, warning, and debugging messages  
See "Setting the Reporting Level for Logging Messages". |
Reconfiguring a BRM Component

Each BRM component: CMs, DMs, and applications; gets configuration information from its configuration file, which the component reads when it starts. Changes you make to a configuration file take effect the next time you start the program.

Important: Most configuration file entries require that you restart the CM, but some do not. For information about restart requirements, see "Business Logic pin.conf Reference" and "System Administration pin.conf Reference".

To reconfigure a BRM component:

<table>
<thead>
<tr>
<th>Entry</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>max_errs</td>
<td>Specifies the maximum number of errors allowed in the application. The application stops when the number of errors exceeds this number. The default is 1.</td>
</tr>
<tr>
<td>max_time</td>
<td>Specifies the maximum time, measured from application start time, for job processing before the application exits. The default is 0, for infinite time.</td>
</tr>
<tr>
<td>monitor</td>
<td>Specifies the file used by the pin_mta_monitor utility. The default is monitor. Important: The file specified is for system use only and should not be deleted or modified.</td>
</tr>
<tr>
<td>multi_db</td>
<td>A flag that determines whether the application works with a BRM multidatabase system. The default is 0. For information, see &quot;Using Multithreaded Applications with Multiple Databases&quot; in BRM Developer’s Guide.</td>
</tr>
<tr>
<td>per_batch</td>
<td>Specifies the number of objects processed by each worker thread in batch mode. The default is 500. Important: This entry is mandatory. See &quot;Tuning the Batch Size for Billing Utilities (per_batch)&quot;.</td>
</tr>
<tr>
<td>per_step</td>
<td>Specifies the number of objects returned by each search step. The default is 1000. Important: This entry is mandatory. See &quot;Setting the Batch Size for Invoice Utilities (per_step)&quot;.</td>
</tr>
<tr>
<td>pin_virtual_time</td>
<td>Enables pin_virtual_time to advance BRM time. See &quot;pin_virtual_time&quot; in BRM Developer’s Guide.</td>
</tr>
<tr>
<td>respawn_threads</td>
<td>Re-spawns worker threads if they exit due to an error. Threads are re-spawned if necessary after every search cycle. The default is 0, for no re-spawning.</td>
</tr>
<tr>
<td>retry_mta_search</td>
<td>The number of retry attempts for main search execution in case of search error. The default is 0, for no retry.</td>
</tr>
<tr>
<td>return_worker_error</td>
<td>Specifies whether to return an error code when any thread encounters an error: ■ 0 specifies to not return an error code. ■ 1 specifies to return an error code. The default is 0.</td>
</tr>
</tbody>
</table>
1. Edit and save the configuration file (pin.conf) for the component. See "Locations of Configuration and Properties Files" and "About Configuration and Properties Files".

2. If required, stop and restart the component. See "Starting and Stopping the BRM System".

Running Non-MTA Utilities on Multidatabase Systems

Utilities that are built on the multithreaded architecture (MTA) framework can perform global searches across all databases in a multidatabase system when configured to do so. See "Using Multithreaded Applications with Multiple Databases" in BRM Developer’s Guide.

The non-MTA utilities shown below, however, can only connect to one CM and its associated database at a time:

- pin_clean (see BRM Configuring and Collecting Payments)
- pin_deposit (see BRM Configuring and Collecting Payments)
- pin_ledger_report (see BRM Collecting General Ledger Data)
- pin_mass_refund (see BRM Managing Accounts Receivable)
- pin_recover (see BRM Configuring and Collecting Payments)
- pin_refund (see BRM Managing Accounts Receivable)
- pin_config_distribution

To run these non-MTA utilities on multidatabase systems, your system must contain a CM for each database on your system. You then connect and run the utility against each CM and database pair. For example, to run a non-MTA utility on a multidatabase system:

- Connect the utility to the primary CM and run the utility against the primary database.
- Connect the utility to a secondary CM and run the utility against its associated secondary database.
- Connect the utility to another secondary CM and run the utility against its associated secondary database.

To run a non-MTA utility on a multidatabase system:

1. Connect the utility to the primary CM:
   a. On the primary CM machine, go to the BRM_Home/apps/pin_billd directory and open the pin.conf file in a text editor.
   b. Edit the following entries:
      - nap cm_ports ip PrimaryHost PrimaryPort
      - nap login_name PrimaryLoginName
      - nap login_pw PrimaryLoginPassword
      - nap login_type LoginType
      - - user_id 0.0.0.x /service/pcm_client 1
   c. Save and close the file.

2. On the primary CM machine, go to the BRM_Home/apps/pin_billd directory and run the utility.
3. Connect the utility to a secondary CM:
   a. On the secondary CM machine, go to the `BRM_Home/apps/pin_billd` directory and open the `pin.conf` file in a text editor.
   b. Edit the following entries:
      
      ```
      - nap cm_ports ip SecondaryHost SecondaryPort
      - nap login_name SecondaryLoginName
      - nap login_pw SecondaryLoginPassword
      - nap login_type LoginType
      - - user_id 0.0.0.x /service/pcm_client 1
      ```
   c. Save and close the file.

4. On the secondary CM machine, go to the `BRM_Home/apps/pin_billd` directory and run the utility.

5. Continue steps 3 and 4 for each remaining secondary database in your system.

---

**Configuring BRM by Using the pin_bus_params Utility**

As part of BRM installation, a standard group of `/config/business_params` objects are added to the BRM database. These objects contain all of the business parameters normally used by BRM. In their default state, these parameters typically disable optional features and direct BRM to use behaviors optimal for most users. However, your business may require optional features or alternative BRM behaviors. Or, you may want to add business parameters or parameter classes that aren’t part of BRM.

If so, you use the `pin_bus_params` utility to perform these tasks. This utility has two capabilities:

- **Retrieving**: You can use the utility to retrieve the contents of an existing `/config/business_params` object in your BRM installation and translate it into an XML file that you can edit.

- **Loading**: You can use the utility to load the contents of an XML file that contains business parameters into an existing `/config/business_params` object or to create entirely new objects.

You can use the XML file created during retrieval to add new parameters for existing classes or to create an entirely new class of parameters. When you use the utility to load the XML file into the `/config/business_params` object, BRM adds the new parameters to your parameter configuration, and these parameters can be called from custom code. For information on adding new parameters and classes, see “Using /config/business_params Objects” in BRM Developer’s Guide.

**Retrieving /config/business_params Objects**

When you retrieve a `/config/business_params` object with `pin_bus_params`, you use the `-r` parameter. To retrieve the object, use the following command syntax:

```
pin_bus_params [-h] -r ParameterClassTag bus_params_ParameterClassTagName.xml
```

This command retrieves the `/config/business_params` object for the specified class into the specified XML file. Consider these points when retrieving business parameters:

- To retrieve a specific parameter, you must know which parameter class it belongs to. The resulting XML file for each class is short so you can quickly locate specific parameters within the class.
Because you retrieve one parameter class at a time, you can edit parameters for a single parameter class without affecting any other parameter classes. BRM overwrites only the `/config/business_params` object whose parameter class appears in the resulting XML file so the overall BRM business parameter configuration remains stable.

To update more than one parameter class, you must perform multiple retrievals and loads, one for each class.

You can create a library of class-based business parameter files and individually reload modified versions of these files only when needed.

### Loading `/config/business_params` Objects

To load parameters into `/config/business_params` objects, use the following command syntax:

```
pin_bus_params [-h] bus_params_ParameterClassName.xml
```

This command finds the `/config/business_params` object for the parameter class in the XML file and overwrites the object with the new parameters.
Implementing System Security

This document explains how to set up permissions and passwords for your Oracle Communications Billing and Revenue Management (BRM) system.

Before you read this document, you should be familiar with how BRM works. See "BRM System Architecture" in BRM Concepts.

For information on authenticating and authorizing customers, see "Managing Customer Authentication" in BRM Managing Customers.

Using General System-Security Measures

You can use the usual database and operating-system security measures for the BRM system. For example, you can set up read/write/execute permissions and group permissions on files and programs.

As shipped, BRM uses encryption only for passwords. However, you can encrypt any string field. For more information, see "About Encrypting Information" in BRM Developer’s Guide.

Configuring Access to the BRM Database

The Data Manager (DM) configuration file (BRM_Home/sys/dm_oracle/pin.conf) specifies the user name and password for logging in to the BRM database. This is the user name and password you specified when you installed and configured the database. By default, the user name is pin and the password is pin. Because the configuration file includes this clear-text password, you should read-protect the file (for UNIX, use mode 600 or 660 with a tight group) or, if you use dm_oracle, use an encrypted password. See "About Encrypting Information" in BRM Developer’s Guide.

---

**Note:** You can automatically encrypt passwords for all BRM components (including dm_oracle) at one time by running the encryptpassword.pl script. For more information, see "About the encryptpassword.pl Script" in BRM Developer’s Guide.

---

Configuring Login Names and Passwords for BRM Access

To access the BRM database, a client application must provide:

- An account name
- The password for that account
- The service
The database number of the BRM database

With that information, the Connection Manager (CM) queries the database. If the information is authenticated, the CM grants the application access to the database.

When you install BRM, the system creates a single user account, called root and identified by the password password, with general permission to the BRM system. This account includes two services, admin_client and pcm_client.

- BRM client applications log in to the admin_client service.
- Other BRM utilities and programs, such as optional service integration components, log in to the pcm_client service.

The default login name and password provided with the system are root.0.0.0.n (where n is your database number) and password.

---

Caution: You must change the password after installing BRM to prevent unauthorized access to BRM.

---

When you set up a production BRM system, you create additional accounts; for example, one for each of your customer service representatives (CSRs), and associate one or more services with each account. You give each account a password and grant certain privileges to the account. For example, you might want to allow only some of your CSRs to handle payment disputes.

You also need to provide an account for any extended applications you use with BRM.

---

Note: You cannot change the payment method of the root account or make it a parent or child account.

---

**Configuring the CM to Verify Application Logins with the Service Only**

By default, the CM is configured to require a service, a login name, and a password. This provides secure access to BRM.

If only secure applications will connect to your CM, you can speed up the login process by configuring the CM to verify only the service but not require a login name or password.

To configure the CM to verify application logins with the service only:

1. Open the Connection Manager (CM) configuration file (BRM_Home/sys/cm/pin.conf).

2. Change the cm_login_module entry from cm_login_pw001.dll to cm_login_null.dll:

   - cm cm_login_module cm_login_null.dll

3. Stop and restart the CM. See “Starting and Stopping the BRM System”.

4. Configure the applications that connect with this CM to provide only service information at log in. In the configuration file for each application, set login_type to 0, and make sure a valid service is listed for userid.
Configuring Login Names and Passwords for BRM Access

### Configuring the Maximum Number of Invalid Login Attempts

To configure the maximum number of invalid login attempts:

1. Run the following command:
   ```
   pin_bus_params -r BusParamsActivity bus_params_act.xml
   ```
   This command creates the XML file named `bus_params_act.xml.out` in your working directory. If you do not want this file in your working directory, specify the full path as part of the file name.

2. Search the XML file for the following line:
   ```xml
   <MaxLoginAttempts>5</MaxLoginAttempts>
   ```

3. Set the value (the default is 5) to the maximum number of login attempts.

4. Save the file as `bus_params_act.xml`.

5. Use the following command to load the change into the `/config/business_params` object:
   ```
   pin_bus_params bus_params_act.xml
   ```
   Run this command from the `BRM_Home/sys/data/config` directory, which includes support files used by the utility. To run it from a different directory, see `pin_bus_params` in BRM Developer’s Guide.

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

7. Stop and restart the CM. See “Starting and Stopping the BRM System”.

For more information, see the following topics in BRM Developer’s Guide:
- Using testnap
- Reading Objects by Using Object Browser

8. For multiple databases, run the `pin_multidb` script with the `-R CONFIG` parameter. For more information on this script, see “pin_multidb”.

### Configuring Applications to Provide Login Information

BRM client applications provide login information in various ways:

- BRM Java-based applications, including Pricing Center, Customer Center, and Configuration Center, ask the user for port numbers and database names when the application starts.

- Payment Tool provides port numbers and database names in its `.ini` file.

- BRM optional service integration components, such as RADIUS Manager, provide all login information from the configuration file for that application.

To change the default login information for client applications, edit the `.ini` or configuration file or use the login dialog box.
Login Information for Payment Tool
To change the default login information for Payment Tool:

1. Open the PaymentTool.ini file, which you can find in the C:\Windows directory.
2. Edit the login entries, and save the file.

Login Information for Optional Service Integration Components
To change the default login information for optional service integration components:

1. Open the configuration file for the application, which you can find in the directory for that application.
2. Follow the guidelines in the configuration file to provide information for these configuration entries:
   - userid
   - login_type
   - login_name
   - login_pw

Login Information for Java-Based Client Applications
To change most connection information for Java-based client applications, use the login dialog box, which appears when you start the application. The application uses this default information for subsequent sessions.

Changing Default Passwords for Java-Based Client Applications
The BRM Java-based client applications’ passwords are located in their Infranet.properties files as encrypted text. To change and encrypt new passwords for an application, you need to log in to the server hosting the application and use the pin_crypt_app utility to encrypt the new password manually. For details, see "About Encrypting Passwords" and "Encrypting Passwords Manually" in BRM Developer’s Guide.

Creating a Customer Service Representative Account
You need to set up accounts for your company’s CSRs so they can create and manage customer accounts.

Before creating CSR accounts, you need to create and load a CSR plan, which defines the services available to CSRs. You do this in Pricing Center. See "Creating CSR Plans" in BRM Setting Up Pricing and Rating.

You can create CSR accounts in either Customer Center or Permissioning Center. After a CSR account exists, you can make changes to it only in Customer Center.

To create CSR accounts:

- **In Customer Center**: See “Creating a CSR account” in Customer Center Help.
- **In Permissioning Center**: In People view, choose Actions - Create New Person.
  For more information, see Permissioning Center Help. For information on password settings for CSR accounts, see “Managing CSR Passwords”.
Changing the Login Name and Password for an Account

To change the login information for an account, use Customer Center. You can log in to the account you want to change or log in to the BRM root account or another account with permission to change the password. You can change the login name and password for each service associated with the account. See Customer Center Help.

**Important:** The BRM root account comes with the default password of password. To prevent unauthorized access to your system, change this password for both services associated with the root account.

Logging Customer Service Representative Activities

The log information includes the type of activity, the date and time the CSR performed the activity, and the IP address of the client computer. This data is stored in the /user_activity storable class.

**Note:** To log CSR activities, session-event logging must be turned on. Session-event logging is on when the CM pin.conf file’s login_audit entry is set to 1 or commented out. See “Turning Off Session-Event Logging”.

You can log CSR activities only if you have included them in the pin_notify file. For more information on using the pin_notify file for event notifications, see "Merging Event Notification Lists" in BRM Developer’s Guide. Use the load_pin_notify utility to load the events specified in the pin_notify file into the BRM database (see BRM Managing Customers).

**Note:** To generate a report displaying the log data, you must write your own query and GUI custom code.

You can log the following CSR activities:

- Account creation
- Account hierarchy changes
- Bill adjustments
- Bill now operations
- Changes to billing day of month
- Changes to billing frequency
- Changes to credit card information
- Creation of charge, discount, monitor, and profile sharing groups
- Creation of deferred action schedules
- Credit limit changes
- Deletion of charge, discount, monitor, and profile sharing groups
- Disputes
- Event adjustments
## Setting Up Permissions in BRM Applications

Permissions determine which tasks a user can perform with BRM applications.

A set of permissions defines a role. A role represents a set of actions that a person holding a particular job or position can perform. For more information, see "About Managing Roles".

You can restrict activities in Customer Center, Pricing Center, and other applications by assigning CSRs to a role and setting permissions for that role. For example, you can specify which CSRs can change a password, apply credits, and give refunds.

As part of setting permissions, you do the following:

- Add permissions to a role. For information on the types of permissions that you can set, see "Permission Types".

  **Note:** For all applications except Customer Center, permission types are not granted by default. CSRs do not have access to the application or feature associated with the permission until permission is explicitly assigned. For Customer Center, some permission types are granted by default. For more information, see Customer Center Help.

- Assign an access level to each permission. For example:
  - You can specify read-only permissions to access critical data such as credit card and pricing information.
  - You can specify read-write access to customer data such as billing address and contact email ID.

- For some permissions, such as giving credits, set a minimum and maximum amount.

  When setting the minimum and maximum values for permissions that allow crediting and debiting a customer’s account, make sure the value you set is appropriate for all non-currency resources that apply to products any customer might own. For example, if 25 is the maximum credit a CSR can issue to a
customer, the CSR cannot credit more than 25 frequent flyer miles or 25 hours of service usage.

You can set up permissions using both Permissioning Center and Customer Center. However, you must have proper permissions to add, change, or delete permissions. In most cases, only a person with root access, such as a system administrator, is granted permission to change CSR permissions. For more information, see "About Managing Permissions in Permissioning Center".

---

**Note:** If your company uses a proprietary application for administering accounts, you can write your own code to set and enforce permissions.

---

You can also restrict access to accounts in a specific brand. To do this, use Access Privileges in Configuration Center. See "About Granting Access to Brands" in *BRM Managing Customers*.

### About Permissioning Center

Permissioning Center is a BRM application you can use to enhance security by managing the roles and permissions of BRM client tool users, such as CSRs.

You perform these tasks by using Permissioning Center:

- **Manage roles.** You can create, rename, and delete roles; add child roles; and assign CSRs to roles. For more information, see "About Managing Roles".

- **Manage permissions.** You can add and delete permissions. For more information, see "About Managing Permissions in Permissioning Center".

- **Manage CSRs.** You can create CSR accounts, assign CSRs to roles, and unassign CSRs from roles.

---

**Note:** You can create CSRs by using Customer Center or Permissioning Center. You assign individual permissions by using Customer Center, and you assign role-based permissions to CSRs by using Permissioning Center.

---

You can view the permissions included in a role and the CSRs assigned to a role by using the Role view and People view in Permissioning Center.

Using Permissioning Center, you can grant or deny access to entire applications, such as Pricing Center or Customer Center, or to individual tabs and fields within Customer Center. This allows you to assign CSRs to roles that correspond to their job functions.

For information on how to create roles and permissions and how to assign CSR accounts to roles, see Permissioning Center Help.

The installation procedure for Permissioning Center is similar to the installation of the other client applications in BRM. See "Installing BRM Client Applications on Windows" in *BRM Installation Guide*.

### About Access to Permissioning Center

CSRs who require access to Permissioning Center must be assigned to a role that includes permissions to use Permissioning Center. You include access permissions to a role in the same way you include any other permissions in Permissioning Center.
A person who has BRM root privileges has full access permissions to create roles and to add or delete permissions to a role in Permissioning Center, Customer Center, or Pricing Center.

A brand manager has default access to Permissioning Center. For other client applications, like Customer Center and Pricing Center, brand managers have to assign roles that contain permissions for accessing the other client applications to themselves.

---

**Note:** After installing Permissioning Center, only the BRM root login name has permissions to use Permissioning Center.

---

For more information on how to manage and include permissions in a role, see Permissioning Center Help.

**About Managing Roles**

You use roles to configure permissions for a group of CSRs based on the tasks they need to perform. For example, you can create different types of CSRs and assign them to different kinds of roles:

- **Manager CSRs** can create new roles, assign CSRs to roles, change permission settings, change credit limits, give refunds, and change account status. A manager can also validate the work that junior CSRs perform, for example, by making sure that new accounts are created correctly and have all the necessary information.

- **Junior CSRs** can check customer account balances, check and change billing information, and answer common customer questions.

For example, CSRs A and B can be assigned to the role Manager, and CSRs C and D can be assigned to the role Lead-CSR, where:

- CSRs A and B have read-write permissions for customer credit card information.
- CSRs C and D have read-only permissions for customer credit card information.

You can also create roles with higher levels of permissions. For example, you can create roles that include permissions to create and manage roles using Permissioning Center.

**About Multiple Role Support**

You can include specific permissions in a role to access one or more client applications. In addition, a CSR can be assigned to multiple roles. For example, a CSR can be assigned to a Manager role in Permissioning Center and to a Junior-CSR role in Pricing Center.

**About Hierarchical Roles**

You can create a role hierarchy in Permissioning Center. To do this, you create child roles and associate them with a parent role.

You organize hierarchical roles according to their permission levels. At each level above the bottom of the hierarchy, the child roles can also be parent roles. A child role inherits all permission settings that are associated with its parent role.

For example, the parent role, CSR, can also have the following child roles:

- Lead-CSR
- Junior-CSR
The child roles include all the permissions that belong to the parent role, CSR. In addition, the child roles have all the specific permissions that belong to their particular role, Lead-CSR or Junior-CSR.

About Managing Permissions in Permissioning Center

In Permissioning Center, permissions are based on roles. The role’s permissions determine the specific levels of access for the role. Using Permissioning Center, you can create new CSR accounts, assign CSRs to roles, and unassign CSRs from roles. This role-based approach makes it easy to quickly grant or deny permissions to an individual CSR or a group of CSRs with a specific role. For more information, see "About Permissioning Center".

In Customer Center, you can provide only one permission at a time to a CSR. Roles cannot be created in Customer Center.

---

**Important:** The permissions set in Customer Center overwrite those set in Permissioning Center.

---

Managing CSR Passwords

To improve security features and provide access to BRM client applications, the following password policies are included in Permissioning Center:

- **Ability to set password expiry limits.** The duration of time that a password is valid until the system prevents a user from logging in or forces the password to be changed.

- **Ability to define temporary passwords.** The ability to force CSRs to change their passwords after accessing the application the first time or after a new CSR account has been set up by an administrator.

- **Password content validation.** The ability to validate the contents of the password to make sure that certain characters are or are not included, such as numbers.

Setting CSR Account Password Status

The following are the valid password statuses for CSR accounts:

- **Temporary**
- **Normal**
- **Expires**
- **Invalid**

You can change a CSR account’s password status in Permissioning Center.

---

**Note:** By default, the password status for a brand manager is set to Normal, and the status for a CSR account is set to Temporary.

---

When a CSR logs in to a BRM application, the system checks for the password status and takes the following actions:

- If the CSR logs in to the application for the first time or if the password is set to Temporary, the Change Password screen is displayed. The CSR is required to change the password before using the application.
Setting Up Permissions in BRM Applications

- If the password status is set to **Normal**, the CSR password never expires.

- By default, the password status is changed from **Temporary** to **Expires** after the first log in attempt. The password expires after a specified number of days. By default, the password expiry duration is 90 days. (See "Setting the Default Password Expiry Duration"). When a CSR password expires, by default, Customer Center displays a screen directing the CSR to change the expired password before launching the application. It is mandatory to change the password after it has expired.

- If the password status is set to **Invalid**, access to the application is denied.

**Automatic Logout**

After you log in to a BRM client application and leave it idle for a specified interval, the session expires automatically. To reconnect to the session again, you are prompted to enter your password again. However, if you have specified the password in the **Infranet.properties** file, the application automatically reconnects by reading the password from the **Infranet.properties** file.

**Note:** The default interval for automatic logout is set to 0 minutes, which implies that this feature is disabled by default. However, this might be different in your BRM system depending on the number specified in the **cm_timeout** entry in the CM’s **pin.conf** file. For more information on **cm_timeout**, see "Setting the CM Time Interval between Opcode Requests".

To access the application after an automatic logout:

1. In the Confirm Password dialog box, enter the password.
2. Click **OK**.

**Note:** BRM client applications use multiple connections. As the interval for automatic logout is different for each connection, you have to perform the above steps once for each connection. Also, if you do not access a window or dialog box for a specified interval, the session expires automatically.

**Changing the Password**

After you log in to a BRM client application, you can change your existing password. After you change the password, you can log in to any BRM application with your new password.
To change the password:

   The Change Password dialog box opens.
2. In the Enter current password field, enter your current password.
3. In the Enter new password field, enter your new password.
4. In the Re-enter new password field, enter the same password to confirm.
5. Click OK.

Unlocking a Locked CSR Account

You can unlock a locked CSR account by using the pin_unlock_service utility.

To unlock a locked CSR account:

1. On the command prompt, run the pin_unlock_service utility. The pin_unlock_service utility is in BRM_home/bin.

   Important:
   - The pin_unlock_service utility needs a configuration (pin.conf) file in the same directory from which you run the utility. See "Creating Configuration Files for BRM Utilities".
   - Make sure that the account in the pin.conf file is not locked.

A menu appears.

2. Press 1.
   You are prompted to select the type of service: admin_client or pcm_client.
3. Select the service type that is associated with the account you want to unlock:
   - For admin_client, press 1.
   - For pcm_client, press 2.
   You are prompted to enter the login ID for the account that you want to unlock.
4. Enter the Login ID and press the Enter key.
5. Enter a new password for the account.
The password must satisfy the following requirements:

- It is at least 6 characters in length.
- It does not exceed 255 characters.
- It contains a combination of letters and numbers.
- It is not the same as the login ID.

6. Confirm the password to unlock the account.
   A message stating that the account has been successfully unlocked appears, followed by a menu:

7. To unlock another account, press 1.
8. To exit the utility, press 2.

Setting the Default Password Expiry Duration
To change the default password expiry duration, edit the passwd_age entry in the CM pin.conf file.

1. Open the CM pin.conf file in BRM_Home/sys/cm.
2. Change the value of the passwd_age entry:
   
   ```
   cm passwd_age 90
   ```
   The default is 90.
3. Stop and restart the CM. See "Starting and Stopping the BRM System".

How Permission Settings Are Stored
You use Permissioning Center to define permission settings for each role at the root or brand level in the /service/admin_client object in the BRM database.

When you assign a CSR account to a role, the account includes all the permissions that are assigned to the role.

---

**Important:** A CSR account can be assigned to only one role per application.

---

The PIN_FLD_ROLE field in the account contains the roles assigned to the CSR account.

Permission Types
You can set permissions for the following applications:

- Collections Center. See "Collections Center Permission Types".
- Customer Center. See Customer Center Help.
- Pricing Center. See "Pricing Center Permission Types".
- Suspense Management Center. See "Suspense Management Center Permission Types".
- Other client applications. See "Other Client Application Permission Types".
Note:

- You can set some permissions in both Permissioning Center and Customer Center. If you set the same permission in both, the permission you set in Customer Center takes precedence.

- For all applications except Customer Center, permission types are not granted by default. CSRs do not have access to the application or feature associated with the permission until permission is explicitly assigned.

- Permission types are case sensitive.

- The Min value does not apply to any of the Suspense Management Center permission types.

Collections Center Permission Types

Table 4–1 shows the permissions you can set for Collections Center:

<table>
<thead>
<tr>
<th>Permission type</th>
<th>Provides permission to . . .</th>
</tr>
</thead>
<tbody>
<tr>
<td>/appcenter/collectioncenter</td>
<td>Use Collections Center.</td>
</tr>
<tr>
<td>/accounttool/collections/agent</td>
<td>Perform the role of collections agent, who carries out collections tasks without any supervisory responsibilities.</td>
</tr>
<tr>
<td>/accounttool/collections/manager</td>
<td>Perform the role of collections manager, who supervises collections agents.</td>
</tr>
<tr>
<td>/collectionapps/collections/actionhistory</td>
<td>View the history of individual tasks.</td>
</tr>
<tr>
<td>/collectionapps/collections/assign</td>
<td>Assign bill units to collections agents.</td>
</tr>
<tr>
<td>/collectionapps/collections/changestatus</td>
<td>Change the status of a task.</td>
</tr>
<tr>
<td>/collectionapps/collections/chargeCC</td>
<td>Receive credit card payments.</td>
</tr>
<tr>
<td>/collectionapps/collections/exempt</td>
<td>Exempt bill units from collections.</td>
</tr>
<tr>
<td>/collectionapps/collections/insert</td>
<td>Insert an additional task into a bill unit scenario.</td>
</tr>
<tr>
<td>/collectionapps/collections/removeexempt</td>
<td>Remove bill units from exemption.</td>
</tr>
<tr>
<td>/collectionapps/collections/reschedule</td>
<td>Reschedule tasks.</td>
</tr>
</tbody>
</table>

Pricing Center Permission Types

Table 4–2 shows the permissions you can set for Pricing Center:

<table>
<thead>
<tr>
<th>Permission type</th>
<th>Provides permission to . . .</th>
</tr>
</thead>
<tbody>
<tr>
<td>/appcenter/pricingcenter</td>
<td>Use Pricing Center.</td>
</tr>
<tr>
<td>/appcenter/provisioningtags</td>
<td>Use Provisioning Tags.</td>
</tr>
<tr>
<td>/appcenter/ResourceEditor</td>
<td>Use Resource Editor.</td>
</tr>
<tr>
<td>/appcenter/ZoneMapper</td>
<td>Use Zone Mapper.</td>
</tr>
</tbody>
</table>
Suspense Management Center Permission Types

Table 4–3 shows the permissions you can set for Suspense Management Center:

<table>
<thead>
<tr>
<th>Permission type</th>
<th>Provides permission to . . .</th>
<th>Max value applies¹</th>
</tr>
</thead>
<tbody>
<tr>
<td>/appcenter/suspensemgt</td>
<td>Use Suspense Management Center.</td>
<td>No</td>
</tr>
<tr>
<td>/appcenter/suspensemgt/archive_and_purge</td>
<td>Archive and purge call records save the records to an archive file and remove them from the BRM database.</td>
<td>Yes</td>
</tr>
<tr>
<td>/appcenter/suspensemgt/batch</td>
<td>Search batches.</td>
<td>Yes</td>
</tr>
<tr>
<td>/appcenter/suspensemgt/batch_writeoff</td>
<td>Write off batches.</td>
<td>Yes</td>
</tr>
<tr>
<td>/appcenter/suspensemgt/batch_purge</td>
<td>Remove batches from the database.</td>
<td>Yes</td>
</tr>
<tr>
<td>/appcenter/suspensemgt/batch_resubmit</td>
<td>Resubmit batches send them back through a pipeline for rating.</td>
<td>Yes</td>
</tr>
<tr>
<td>/appcenter/suspensemgt/bulkedit</td>
<td>Edit a large number suspended call records in one database operation.</td>
<td>Yes</td>
</tr>
<tr>
<td>/appcenter/suspensemgt/bulkpurge</td>
<td>Delete a large number of suspended call records in one database operation.</td>
<td>Yes</td>
</tr>
<tr>
<td>/appcenter/suspensemgt/bulkrecycle</td>
<td>Recycle a large number of suspended call records in one database operation.</td>
<td>Yes</td>
</tr>
<tr>
<td>/appcenter/suspensemgt/bulkwriteoff</td>
<td>Write off a large number of suspended call records in one database operation.</td>
<td>Yes</td>
</tr>
<tr>
<td>/appcenter/suspensemgt/edit</td>
<td>Edit suspended call records.</td>
<td>Yes</td>
</tr>
<tr>
<td>/appcenter/suspensemgt/purge</td>
<td>Purge call records from the BRM database.</td>
<td>Yes</td>
</tr>
<tr>
<td>/appcenter/suspensemgt/records</td>
<td>Search records.</td>
<td>Yes</td>
</tr>
<tr>
<td>/appcenter/suspensemgt/recycle</td>
<td>Recycle suspended call records send them back through a pipeline for rating.</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Setting Up Access to Brands

### Other Client Application Permission Types

Table 4–4 shows the permissions for accessing client applications. To set up permission for accessing a client application, choose that application’s name from the Application list and enter the permission type.

<table>
<thead>
<tr>
<th>Permission type</th>
<th>Provides permission to . . .</th>
<th>Max value applies¹</th>
</tr>
</thead>
<tbody>
<tr>
<td>/appcenter/suspmgt/restore</td>
<td>Restore call records from an archive file.</td>
<td>No</td>
</tr>
<tr>
<td>/appcenter/suspmgt/undo_edit</td>
<td>Undo edits to suspended call records.</td>
<td>No</td>
</tr>
<tr>
<td>/appcenter/suspmgt/writeoff</td>
<td>Write off suspended call records.</td>
<td>Yes</td>
</tr>
</tbody>
</table>

### Access Levels

The permission levels included in a role determine the access level of the CSR. The permission levels are listed in Table 4–5:

<table>
<thead>
<tr>
<th>Permission level</th>
<th>Display data?</th>
<th>Change data?</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Read Only</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Write Only</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Note:</strong> This level is typically used only for credit card numbers or passwords. For most permission types, this level is the same as None.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Read/Write</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

### Setting Up Access to Brands

BRM systems that use branded service management to host multiple brands of service use privileges to keep each brand’s customer accounts and price lists separate. The root account has access to all brands. The Brand Manager accounts that you define when creating a brand have access to all data in that brand.

Other users, such as CSRs, get access to brands through access groups. Each access group gives member users access to a single brand or to a single sub-brand or account group within a brand.

You create access groups in the Access Privileges application in Configuration Center.

For more information, see “About Granting Access to Brands” in *BRM Managing Customers* and the Help in Access Privileges.
Protecting BRM Passwords

BRM stores account passwords (for /service/admin_client and /service/pcm_client) in encrypted form in the database.

---

**Important:** If a BRM user forgets a password, you must provide a new one; you cannot retrieve the old one.

---

You can change the encryption algorithm, which is AES by default, to another algorithm. To do this, modify the fm_cust_pol_encrypt_passwd.c policy module in BRM_Home/source/sys/fm_cust_pol. See "Adding and Modifying Policy Facilities Modules" in BRM Developer’s Guide.

BRM developers can also encrypt other string fields. If your BRM system uses a custom encryption method, the encryption values in the DM configuration file (BRM_Home/sys/dm_oracle/pin.conf) for encryption should be updated. For instructions, see "Encrypting Fields" in BRM Developer’s Guide.

IP passwords (for /service/ip) are clear text; this is a requirement of Challenge Handshake Authentication Protocol (CHAP). However, the password is never sent to the NAS (terminal server), either in encrypted or clear-text form because CHAP encrypts a random-number challenge token based on the customer’s password.
This document provides information and guidelines to help you manage the day-to-day operation of your Oracle Communications Billing and Revenue Management (BRM) system by using HP OpenView and the BRM 7.4 Certified Smart Plug-in for HP OpenView.

It also includes instructions for configuring the BRM Smart Plug-in (SPI).

**About Using HP OpenView to Monitor BRM**

You can use the BRM certified SPI for HP OpenView to manage and control your distributed BRM system from a central graphical user interface. You can do the following tasks:

- Monitor and control key BRM processes; for example, CMs and DMs. You can start and stop processes, and get the status.
- Display log files.
- Collect and store data about EDR and AAA opcode processing and display historical comparisons in graphs.
- Display BRM system topology and process status.
- Configure alarms for the following cases:
  - When a BRM process goes down or comes up.
  - When a Critical or Major error occurs.
  - When the CPU or Memory usage by a BRM process exceeds a configured threshold.
  - When EDR processing (EDRs per second) by a pipeline falls below a configured threshold.
  - When AAA opcode performance falls below a configured level.

**Hardware and Software Requirements**

The BRM certified SPI for HP OpenView is supported on Solaris, HP-UX IA64, and AIX. For a list of supported operating system versions, see "BRM Software Compatibility" in *BRM Installation Guide*. 
The installed software requires 4.1 MB of disk space.

For information about HP OpenView hardware and software requirements, see the HP OpenView documentation.

### Configuring HP OpenView When You Use Multiple Install Directory Names

If you use multiple directory names for \textit{BRM\_Home} and multiple process names, you might need to configure the BRM templates and applications in HP OpenView before you deploy them to HP OpenView agents. This requirement depends on your configuration.

If you use either of the following configurations, you do not need to configure the templates and applications before deployment.

- Different process names and different \textit{BRM\_Home} names as shown in Table 5–1:

<table>
<thead>
<tr>
<th>Host</th>
<th>Process</th>
<th>\textit{BRM_Home}</th>
</tr>
</thead>
<tbody>
<tr>
<td>host_1</td>
<td>cm</td>
<td>/opt/portal/7.4</td>
</tr>
<tr>
<td>host_2</td>
<td>cm_2</td>
<td>/opt/portal/7.4_alt</td>
</tr>
</tbody>
</table>

- Different process names and the same \textit{BRM\_Home} name as shown in Table 5–2:

<table>
<thead>
<tr>
<th>Host</th>
<th>Process</th>
<th>\textit{BRM_Home}</th>
</tr>
</thead>
<tbody>
<tr>
<td>host_1</td>
<td>cm</td>
<td>/opt/portal/7.4</td>
</tr>
<tr>
<td>host_2</td>
<td>cm_2</td>
<td>/opt/portal/7.4</td>
</tr>
</tbody>
</table>

If you use the \textit{same} process name with different \textit{BRM\_Home} names as shown in Table 5–3, you need to configure templates and applications before deployment:

<table>
<thead>
<tr>
<th>Host</th>
<th>Process</th>
<th>\textit{BRM_Home}</th>
</tr>
</thead>
<tbody>
<tr>
<td>host_1</td>
<td>cm</td>
<td>/opt/portal/7.4</td>
</tr>
<tr>
<td>host_2</td>
<td>cm</td>
<td>/opt/portal/7.4_alt</td>
</tr>
</tbody>
</table>

In this case, before you deploy templates and applications to the HP OpenView agents, you need to configure BRM HP OpenView templates and applications to use the correct process names and \textit{BRM\_Home} directories. Configure the templates and applications on each managed node. To do so, use the HP OpenView template and tools editors. See the HP OpenView documentation for information.

### About the HP OpenView Architecture

You use HP OpenView by running an HP OpenView console application. The HP OpenView console application connects to the HP OpenView Management Server.
The HP OpenView Management Server can be configured to connect to one or more host systems. An HP OpenView agent is installed on each BRM host system.

The agent carries out the actions defined in the BRM SPI. An SPI is a bundle of templates and tools. The systems on which the agent and the SPI are installed are called managed nodes.

- **Templates** are rules for monitoring and carrying out actions. For example, when monitoring disk space, the agent relies on a template to know when to report low disk space.

- **Tools** are wrappers for scripts and utilities. For example, an agent uses the `pin_ctl` utility to start and stop processes.

Agents rely on system events to trigger messages or actions. For example, if a process stops, that event tells the agent to send a message to the HP OpenView Management Server, for display by the HP OpenView clients.

The HP OpenView architecture is organized like Figure 5–1:

You can use the HP OpenView tools and template editors to customize the BRM templates and tools. For example, you can change the severity of an error message, require a password for a command, change alarm thresholds, and so forth.

You can set up application groups to control who has access to the BRM tools in HP OpenView. See the HP OpenView documentation.

To customize and configure HP OpenView, see the HP OpenView documentation.
Managing BRM Components in HP OpenView

You can use the BRM tools to stop and start components, display status, and display log files.

To run the BRM tools in HP OpenView:

1. In the HP OpenView Management Server, choose the node from the Node Bank.
2. Open the Application Bank.
4. Double-click the icon for the tool you want to run.

Each tool has an icon; for example:

- CM Status
- Start CMMP
- View DM_Fusa Log

---

**Note:**

- HP OpenView carries out commands by running the `pin_ctl` utility. Because the `pin_ctl` utility needs to be run by a BRM user, you need to update the user name and password in the Execute as User box for each application. See the HP OpenView documentation for information about modifying applications.

- The icons that appear are based on the components included in the PORTALBASE_COMPONENT_LIST entry in the `pin_comp.values` file. For example, if you include a CM in the list, the tools for the CM are displayed in the Application Bank. See "Configuring the BRM SPI by Editing the pin_comp.values File".

- There is no start-all or stop-all tool in the BRM tools. You can use the `pin_ctl` utility to stop or start all processes.

When you start or stop a component, a status message is displayed. You can also see the command results in the HP OpenView Message Browser window.

---

Monitoring Error Messages in Log Files by Using HP OpenView

To monitor errors, BRM reads log files to find specific error codes that indicate problems that need attention. If an error message is found, HP OpenView displays it in the Message Browser window. The Message Browser shows the severity, date, time, process, and message text.

Not all error messages are monitored. BRM monitors a subset of error messages. See "Error Messages Monitored by HP OpenView".

You can use the HP OpenView tools and template editors to change how often errors are monitored and to customize which errors are monitored.
Error Messages Monitored by HP OpenView

The following tables show the default error messages monitored by HP OpenView. For more information about each error code, see "Reference Guide to BRM Error Codes".

Table 5–4 lists the BRM System Error Codes monitored by HP OpenView:

<table>
<thead>
<tr>
<th>Error Component</th>
<th>Error Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>NULL MAPINFO pointer IN</td>
<td>PIN_ERR_CM_ADDRESS_LOOKUP_FAILED</td>
</tr>
<tr>
<td>PIN_ERR_CM_ADDRESS_LOOKUP_FAILED</td>
<td>CM</td>
</tr>
<tr>
<td>PIN_ERR_CONNECTION_LOST</td>
<td>CM</td>
</tr>
<tr>
<td>PIN_ERR_DEADLOCK</td>
<td>Oracle DM</td>
</tr>
<tr>
<td>PIN_ERR_DM_ADDRESS_LOOKUP_FAILED</td>
<td>CM</td>
</tr>
<tr>
<td>PIN_ERR_DM_CONNECT_FAILED</td>
<td>CM</td>
</tr>
<tr>
<td>PIN_ERR_MASTER_DOWN</td>
<td>CM</td>
</tr>
<tr>
<td>PIN_ERR_NO_MEM</td>
<td>CM</td>
</tr>
<tr>
<td>PIN_ERR_NO_SOCKET</td>
<td>CM</td>
</tr>
<tr>
<td>PIN_ERR_STORAGE</td>
<td>CM</td>
</tr>
<tr>
<td>PIN_ERR_STORAGE_DISCONNECT</td>
<td>CM</td>
</tr>
<tr>
<td>PIN_ERR_STORAGE_FAILOVER</td>
<td>CM</td>
</tr>
<tr>
<td>PIN_ERR_STREAM_EOF</td>
<td>CM</td>
</tr>
<tr>
<td>PIN_ERR_STREAM_IO</td>
<td>CM</td>
</tr>
<tr>
<td>PIN_ERR_TRANS_LOST</td>
<td>CM</td>
</tr>
<tr>
<td>PIN_ERRCLASS_APPLICATION</td>
<td>CM</td>
</tr>
<tr>
<td>PIN_ERRCLASS_SYSTEM_DETERMINATE</td>
<td>CM</td>
</tr>
<tr>
<td>PIN_ERRCLASS_SYSTEM_INDETERMINATE</td>
<td>CM</td>
</tr>
<tr>
<td>PIN_ERRCLASS_SYSTEM_RETRYABLE</td>
<td>CM</td>
</tr>
<tr>
<td>Unable to establish database connection.</td>
<td>Batch Controller</td>
</tr>
</tbody>
</table>

Table 5–5 lists the Pipeline Manager Error Codes monitored by HP OpenView:

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_ALL_CM_CONNECTIONS_DOWN</td>
<td>17006</td>
</tr>
<tr>
<td>ERR_ALL_CM_CONNECTIONS_LOST</td>
<td>15902</td>
</tr>
<tr>
<td>ERR_BALANCE_DATABASE</td>
<td>31233</td>
</tr>
</tbody>
</table>
Using HP OpenView to Display System Status and Topology

You can use the HP OpenView Service Navigator to display the entire distributed BRM system topology from a single location.

In addition to the topology display, the service navigator can display the commands available for each component (for example, stop and start) and alarms assigned to the components.

Table 5–5 (Cont.) Pipeline Manager Error Codes Monitored by HP OpenView

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_BALANCE_THREAD_INIT</td>
<td>31235</td>
</tr>
<tr>
<td>ERR_BALRT_EXECUTING_OPCODE</td>
<td>31303</td>
</tr>
<tr>
<td>ERR_BEGIN_EDR</td>
<td>18006</td>
</tr>
<tr>
<td>ERR_CANNOT_CREATE_CONTROLLER</td>
<td>11205</td>
</tr>
<tr>
<td>ERR_CANNOT_CREATE_CTL_QUEUE</td>
<td>11203</td>
</tr>
<tr>
<td>ERR_CANNOT_CREATE_RECORD</td>
<td>11204</td>
</tr>
<tr>
<td>ERR_CANNOT_FORK</td>
<td>00354</td>
</tr>
<tr>
<td>ERR_CANNOT_OPEN_DATABASE</td>
<td>00213</td>
</tr>
<tr>
<td>ERR_CM_CONNECTION_LOST</td>
<td>17008</td>
</tr>
<tr>
<td>ERR_COMMIT_EDR</td>
<td>18007</td>
</tr>
<tr>
<td>ERR_CONNECT_FAILED</td>
<td>17001</td>
</tr>
<tr>
<td>ERR_DATABASE_CONNECTION</td>
<td>11225</td>
</tr>
<tr>
<td>ERR_DOC_OPEN_NEXT_QUEUE</td>
<td>11160</td>
</tr>
<tr>
<td>ERR_DOC_TABLE_FIND</td>
<td>11114</td>
</tr>
<tr>
<td>ERR_DB_CONNECTION_MODULE</td>
<td>00218</td>
</tr>
<tr>
<td>ERR_DB_CONNECTION_NOT_VALID</td>
<td>00411</td>
</tr>
<tr>
<td>ERR_INP_RECYCLE_NO_DB_CONNECTION</td>
<td>10510</td>
</tr>
<tr>
<td>ERR_INSUFFICIENT_MEMORY</td>
<td>00231</td>
</tr>
<tr>
<td>ERR_LINE_NOT_IDENTIFIED</td>
<td>00410</td>
</tr>
<tr>
<td>ERR_LOGIN_FAILED</td>
<td>17002</td>
</tr>
<tr>
<td>ERR_NOT_ENOUGH_CONNECTIONS</td>
<td>30014</td>
</tr>
<tr>
<td>ERR_QUEUE_NOT_INSTALLED</td>
<td>31007</td>
</tr>
<tr>
<td>ERR_RECYCLING_DATA_MODULE_NOT_FOUND</td>
<td>10502</td>
</tr>
<tr>
<td>ERR_SEND_DATA</td>
<td>15914</td>
</tr>
<tr>
<td>ERR_STREAM_NOT_FOUND</td>
<td>00110</td>
</tr>
<tr>
<td>ERR_THREAD_CANCELLED</td>
<td>30081</td>
</tr>
<tr>
<td>ERR_THREAD_DIED_UNEXPECTED</td>
<td>30021</td>
</tr>
</tbody>
</table>

**Note:** Only error message text (for example, ERR_ALL_CM_CONNECTIONS_DOWN) is monitored. Error numbers are not monitored.
To display the BRM system topology in HP OpenView, you need to customize and load the service configuration file. See "Editing and Loading the Service Configuration File".

Using HP OpenView to Display Performance History Data

You can use the HP OpenView Performance Manager to display performance data.

You can plot graphs of performance data. You can also compare performance data between two or more components, on the same system or on different systems.

You can collect the following data:
- CPU and memory usage for all BRM system processes.
- EDR count and EDRs-per-second throughput statistics for pipelines. Alarms are configured for EDR throughput.
- GSM opcode performance.
- Opcode usage and average processing time for GSM AAA Gateway Manager opcodes. Alarms are configured for opcode average processing time.

To configure which performance data to collect by using HP OpenView, see "Configuring the BRM SPI by Editing the pin_comp.values File".

You can use the HP OpenView template editor to customize how long to store historical data and how often to poll and save data.

Using Alarms

You can use HP OpenView to configure system alarms; for example:
- CPU usage.
- Memory usage.
- Pipeline performance.
- Opcode performance.

Alarms are displayed in the Message Browser.

You can use HP OpenView to configure actions to carry out in response to alarms. By default, the BRM alarms are not configured to trigger any actions.

BRM includes a set of default alarms. You can customize the thresholds used with these alarms. For example, you can change the CPU usage value at which an alarm is triggered. See "Configuring the BRM SPI by Editing the pin_comp.values File".

---

**Note:**
- Different components and processes require different threshold values. The default threshold values might not be appropriate for your system. To determine reasonable threshold values, collect historical data for your system and adjust the values accordingly.
- All alarms are reset back to normal when performance returns to normal.

---

Table 5–6 lists the BRM default alarms:
Before installing the BRM SPI, install the BRM Third-Party software and HP OpenView. See "Installing the Third-Party Software" in BRM Installation Guide. You do

### Table 5–6  BRM Alarms

<table>
<thead>
<tr>
<th>Alarm</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU usage</td>
<td>This type of alarm monitors CPU usage for BRM components such as CMs, DMs, and AAA Gateway. Alarms are raised as follows:</td>
</tr>
<tr>
<td></td>
<td>■ If the CPU usage for a process crosses 75% and remains at that level for 30 minutes, a Warning alarm is raised.</td>
</tr>
<tr>
<td></td>
<td>■ If the CPU usage crosses 75% for 30 minutes, but is not more than 80% for more than 15 minutes, a Minor alarm is raised.</td>
</tr>
<tr>
<td></td>
<td>■ If the CPU usage crosses 75% for 30 minutes, and is between 80% and 85% for more than 15 minutes, a Major alarm is raised.</td>
</tr>
<tr>
<td></td>
<td>■ If the CPU usage crosses 75% for 30 minutes, and is more than 85% for more than 15 minutes, a Critical alarm is raised.</td>
</tr>
<tr>
<td>Memory usage</td>
<td>This type of alarm monitors memory usage for BRM components such as CMs, DMs, and AAA Gateway. Alarms are raised as follows:</td>
</tr>
<tr>
<td></td>
<td>■ If memory usage is greater than 40% for 30 minutes, BRM raises an alarm every 30 minutes. A critical alarm is raised when usage reaches 45%.</td>
</tr>
<tr>
<td>Opcode performance</td>
<td>BRM measures the average opcode latency over time. An alarm is raised when the latency reaches a threshold; for example, 55 milliseconds. The alarms support the GSM AAA Gateway Manager opcodes. Alarms are raised as follows:</td>
</tr>
<tr>
<td></td>
<td>■ If the opcode average processing time stays above the specified minimum threshold for more than 30 minutes, a Minor alarm is raised.</td>
</tr>
<tr>
<td></td>
<td>■ If the opcode processing time stays above the minimum threshold for an additional 15 minutes, a Major alarm is raised.</td>
</tr>
<tr>
<td></td>
<td>■ If the opcode processing time goes above the maximum threshold, a Critical alarm is raised.</td>
</tr>
<tr>
<td>Error messages</td>
<td>These alarms occur whenever a specified error message occurs. The alarms are read from log files for the following components:</td>
</tr>
<tr>
<td></td>
<td>■ Pipeline Manager</td>
</tr>
<tr>
<td></td>
<td>■ Connection Manager</td>
</tr>
<tr>
<td></td>
<td>■ Oracle DM</td>
</tr>
<tr>
<td></td>
<td>■ Batch Controller</td>
</tr>
<tr>
<td></td>
<td>See &quot;Monitoring Error Messages in Log Files by Using HP OpenView&quot;.</td>
</tr>
<tr>
<td>Pipeline EDR</td>
<td>BRM measures the average EDR throughput over time. Alarms are raised as follows:</td>
</tr>
<tr>
<td>throughput</td>
<td>■ If the EDR throughput (EDRs per second) stays less than the specified minimum threshold for more than 30 minutes, a Minor alarm is raised.</td>
</tr>
<tr>
<td></td>
<td>■ If the EDR throughput stays below the minimum threshold for an additional 15 minutes, a Major alarm is raised.</td>
</tr>
<tr>
<td></td>
<td>■ If the EDR throughput goes below the maximum threshold, a Critical alarm is raised.</td>
</tr>
</tbody>
</table>
not need to install the BRM server software, such as CMs and DMs, on the HP OpenView system.

1. Install the BRM SPI software package.
2. Configure the pin_comp.values file. See "Configuring the BRM SPI by Editing the pin_comp.values File".
3. Configure the BRM SPI. See "Configuring the BRM SPI".

Installing the BRM SPI

1. Download the software to a temporary directory (temp_dir).

   **Important:** You must increase the heap size used by the Java Virtual Machine (JVM) before running the installation program to avoid “Out of Memory” error messages in the log file. See 'Increasing Heap Size to Avoid 'Out of Memory' Error Messages” in BRM Installation Guide.

2. Go to the directory where you installed the Third-Party package and source the source.me file.

   **Caution:** You must source the source.me file to proceed with installation, otherwise “suitable JVM not found” and other error messages appear.

   Bash shell:
   ```
   source source.me.sh
   ```

   C shell:
   ```
   source source.me.csh
   ```

3. Go to the temp_dir directory and enter this command:
   ```
   7.4_Portal_SPI_For_HPOpenView_platform_32_opt.bin
   ```

   **Note:** You can use the -console parameter to run the installation in command-line mode. To enable a graphical user interface (GUI) installation, install a GUI application such as X Windows and set the DISPLAY environment variable before you install the patch.

4. The installation asks you to specify the destination directory. The default BRM SPI installation directory is /var/opt/OV/share/tmp/portal_spi_hpopenview.

   Your BRM SPI installation is now complete.

**portal_spi_hpopenview Directory Contents**
The install directory contains the following subdirectories:
Installing and Configuring the HP OpenView BRM SPI

- **bin**: Contains the following:
  - Executable scripts for monitoring and controlling BRM processes.
  - Scripts for performance setup and data collection. You deploy these scripts to the managed nodes.
- **portal_graph_conf**: Contains graph templates.
- **portal_perf_conf**: Contains performance templates.
- **portal_spi_conf**: Contains application, log file monitoring, and process monitoring templates.
- **schemas**: Contains the service navigator XML file, which needs to be uploaded into the HP OpenView Service Navigator.

Uninstalling the BRM SPI

Go to `Install_dir/uninstaller/Portal_SPI_For_HPOpenView` and run the `uninstaller.bin` program.

For example:
```
uninstaller.bin -console
```

Configuring the BRM SPI by Editing the `pin_comp.values` File

Edit the `pin_comp.values` file to configure the components that you want to monitor as well as the components for which you want to collect performance data. For example, you can configure:

- The processes you want to monitor; for example, pipeline and opcodes.
- Alarm thresholds; for example, the CPU usage threshold that sends an alarm. See "Using Alarms".

How the `pin_comp.values` File Is Organized

The `pin_comp.values` file includes several sets of component lists and related performance settings. For example, this entry lists the pipelines that are configured for data collection:

```plaintext
@PIPELINE_EDR_LIST = ('ALL_RATE','PRE_CYCLE');
```

These entries configure each pipeline.

```plaintext
$MAX_EDR_PROCESSED{$PIPELINE_EDR_LIST[0]} = 88;
$MIN_EDR_PROCESSED{$PIPELINE_EDR_LIST[0]} = 99;

$MAX_EDR_PROCESSED{$PIPELINE_EDR_LIST[1]} = 11;
$MIN_EDR_PROCESSED{$PIPELINE_EDR_LIST[1]} = 22;
```

The `$PIPELINE_EDR_LIST` entries identify the pipelines by the order they appear in the `@PIPELINE_EDR_LIST` entry, identified numerically; for example, `$PIPELINE_EDR_LIST[0]` refers to the first entry (0) in the `@PIPELINE_EDR_LIST` entry, which is `ALL_RATE`.

Editing the `pin_comp.values` File

1. Open the `pin_comp.values` file in a text editor. The default location is `/var/opt/OV/share/tmp/portal_spi_hpopenview`. 
2. Edit the @PORTALBASE_PROCESS_LIST entry to include the BRM system components you want to control and monitor in HP OpenView. For example:

```
@PORTALBASE_PROCESS_LIST = ("cm", "cm_1");
```

**Important:** The names must match the names of component executables; for example, “cm” and “cm_1.”

The @PORTALBASE_PROCESS_LIST entry determines which components you can control from the HP OpenView Application Bank. For example, if you include a CM, the Application Bank includes tools for stopping and starting the CM, viewing the log file, and getting the CM status.

3. Edit the $MAP_PROCESS entries for the BRM system components. There is one set of data for each component listed in the PORTALBASE_PROCESS_LIST entry. For example:

```
$MAP_PROCESS{$PORTALBASE_PROCESS_LIST[0]} = "cm";
$PIN_LOG{$PORTALBASE_PROCESS_LIST[0]} = "/var/portal/7.4";
$PIN_HOME{$PORTALBASE_PROCESS_LIST[0]} = "/opt/portal/7.4";
$PIN_LOG_PATH{$PORTALBASE_PROCESS_LIST[0]} = "/var/portal/7.4/cm/cm.pinlog";
$CPU_WARN_PERCENT{$PORTALBASE_PROCESS_LIST[0]} = 75;
$CPU_MIN_PERCENT{$PORTALBASE_PROCESS_LIST[0]} = 80;
$CPU_MAX_PERCENT{$PORTALBASE_PROCESS_LIST[0]} = 85;
$MEM_WARN_PERCENT{$PORTALBASE_PROCESS_LIST[0]} = 40;
$MEM_MIN_PERCENT{$PORTALBASE_PROCESS_LIST[0]} = 40;
$MEM_MAX_PERCENT{$PORTALBASE_PROCESS_LIST[0]} = 45;
```

The $MAP_PROCESS entries are in the same order as they are listed in the @PORTALBASE_PROCESS_LIST entry.

Table 5–7 lists the $MAP_Process entries:
### Table 5-7  $MAP_PROCESS entries

<table>
<thead>
<tr>
<th>Entry</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>$MAP_PROCESS</td>
<td>The process name. The names must be:</td>
<td>$MAP_PROCESS($PORTALBASE_PROCESS_LIST[0]) = &quot;cm&quot;;</td>
</tr>
<tr>
<td></td>
<td>■ cm: All Connection Managers, such as cm_1, cm_2, and so forth.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ dm_oracle: Data Managers such as dm_oracle, dm_fusa, dm_ifw_sync, dm_email, dm_eai, and dm_invoice.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ batch_controller: Batch Controller.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ .eai_js: EAI Manager.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ formatter: Invoice Formatter.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$PIN_LOG The BRM log message home directory. This should be the same directory assigned</td>
<td>$PIN_LOG($PORTALBASE_PROCESS_LIST[0])=&quot;/var/portal/7.4&quot;;</td>
</tr>
<tr>
<td></td>
<td>in the PIN_LOG variable during the BRM installation.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>By default, the directory is /var/portal/7.4. See &quot;BRM Environment Variables&quot; in BRM Installation</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Guide.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$PIN_HOME The BRM Home directory. This should be the same directory assigned</td>
<td>$PIN_HOME($PORTALBASE_PROCESS_LIST[0])=&quot;/opt/portal/7.4&quot;;</td>
</tr>
<tr>
<td></td>
<td>in the PIN_HOME variable during the BRM installation.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>By default, the directory is /opt/portal/7.4. See &quot;BRM Environment Variables&quot; in BRM Installation</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Guide.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$PIN_LOG_PATH The location of the component’s log files, as configured in the component’s pin.conf</td>
<td>$PIN_LOG_PATH($PORTALBASE_PROCESS_LIST[0])=&quot;/var/portal/7.4/cm/cm.pinlog&quot;;</td>
</tr>
<tr>
<td></td>
<td>or Infranet.properties file.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$CPU_WARN_PERCENT The percentage of CPU usage at which a CPU usage Minor (yellow) alarm is</td>
<td>$CPU_WARN_PERCENT($PORTALBASE_PROCESS_LIST[0]) = 75;</td>
</tr>
<tr>
<td></td>
<td>raised. See &quot;Using Alarms&quot;.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$CPU_MIN_PERCENT The threshold at which a CPU usage Major (orange) alarm is</td>
<td>$CPU_MIN_PERCENT($PORTALBASE_PROCESS_LIST[0]) = 80;</td>
</tr>
<tr>
<td></td>
<td>raised.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$CPU_MAX_PERCENT The threshold at which a CPU usage Critical (red) alarm is</td>
<td>$CPU_MAX_PERCENT($PORTALBASE_PROCESS_LIST[0]) = 85;</td>
</tr>
<tr>
<td></td>
<td>raised.</td>
<td></td>
</tr>
</tbody>
</table>
4. Edit the $PERF_LOG entry to specify the location of the portal_log file. The portal_log file is the binary file where BRM-specific performance data is stored.

5. Edit the $SEQ entry to change the starting point from which unique sequence numbers for class specifications are created.

   The $SEQ value is the starting number used to generate unique numbers associated with each type of data you collect; for example, CPU usage and memory allocation. You need to change this number only if you already collect performance data that uses the $SEQ number defined in this file.

   The default is 5000.

6. Edit the @PIPELINE_PROCESS_LIST entry to include the Pipeline Manager configurations that you use. For example:

   @PIPELINE_PROCESS_LIST = ('bre', 'rtp', 'aaa');

   where:

   ■ bre: Batch pipeline rating.
   ■ rtp: Real-time pipeline zoning and discounting.
   ■ aaa: AAA Gateway Manager.

7. Edit the $MAP_PROCESS entries for Pipeline Manager. There is one set of data for each component listed in the @PIPELINE_PROCESS_LIST entry. The Pipeline Manager $MAP_PROCESS entries must be in the same order as they are listed in the @PIPELINE_PROCESS_LIST entry.

   The Pipeline Manager $MAP_PROCESS entries are the same as the BRM system $MAP_PROCESS entries. The differences are:

   ■ The $MAP_PROCESS entry is always ifw.process; for example:

     $MAP_PROCESS({$PIPELINE_PROCESS_LIST[0]}) = "ifw.process";

   ■ $IFW_HOME is Pipeline_Home.

     $IFW_HOME({$PIPELINE_PROCESS_LIST[0]})="/opt/ifw";

   See $MAP_PROCESS entries.

8. Edit the @PIPELINE_LIST entry to include the pipelines that you want to monitor:

   @PIPELINE_LIST = ("discounting", "PRE_CYCLE", "ALL_RATE", "");
9. For each pipeline, edit the $IFW_LOG_PATH entry and $MAP_PROCESS entries; for example:

```bash
$IFW_LOG_PATH{$PIPELINE_LIST[0]}="/opt/ifw/log/pipeline/pipeline_discounting.log";
$MAP_PROCESS{$PIPELINE_LIST[0]} = "ifw.pipeline";

$IFW_LOG_PATH{$PIPELINE_LIST[1]}="/opt/ifw/log/pipeline/pipeline_PRE_CYCLE.log";
$MAP_PROCESS{$PIPELINE_LIST[1]} = "ifw.pipeline";
```

The $IFW_LOG_PATH entry specifies the path to the pipeline log file. This path should be the same as the path defined in the **PipelineLog** registry entry. See "About Pipeline Manager Log Files".

- The entries must be in the same order as in the @PIPELINE_LIST entry.
- $MAP_PROCESS is always **ifw.process**.

10. Edit the @OPCODE_NUMBER_LIST entry to include the opcodes that you want to get performance data from; for example:

```bash
@OPCODE_NUMBER_LIST = ("4034", "4036");
```

**Important:** You must use only opcode numbers, not opcode names.

Opcode numbers are in the opcode header file; for example, **gsm_aaa.h**.

By default, the following opcodes are monitored:

- PCM_OP_GSM_AAA_AUTHORIZE (4036)
- PCM_OP_GSM_AAA_CANCEL_AUTHORIZATION (4038)
- PCM_OP_GSM_AAA_REAUTHORIZE (4037)
- PCM_OP_GSM_AAA_STOP_ACCOUNTING (4041)

You can monitor any of the GSM AAA opcodes.

11. Edit the $MAX_AVG_PROCESS_TIME and $MIN_AVG_PROCESS_TIME entries for each opcode.

- The entries must be in the same order that the opcodes are listed in the @OPCODE_NUMBER_LIST entry.
- The processing time is in milliseconds:

  ```bash
  $MAX_AVG_PROCESS_TIME{$OPCODE_NUMBER_LIST[0]} = 55;
  $MIN_AVG_PROCESS_TIME{$OPCODE_NUMBER_LIST[0]} = 66;
  ```

  The $MIN_AVG_PROCESS_TIME value is the threshold at which a Critical (Red) alarm is raised.
  The $MAX_AVG_PROCESS_TIME value is the threshold at which a Major (orange) alarm is raised.

12. Edit the @PIPELINE_EDR_LIST entry to include the pipelines for which you want to collect EDR throughput statistics; for example:

```bash
@PIPELINE_EDR_LIST = ("ALL_RATE","PRE_CYCLE");
```
13. Edit the $MAX_EDR_PROCESSED and $MIN_EDR_PROCESSED entries for each pipeline. The values are in EDRs per second.

- The $MIN_EDR_PROCESSED value is the threshold at which a Critical (red) alarm is raised.
- The $MAX_EDR_PROCESSED value is the threshold at which a Major (orange) alarm is raised.

See "Using Alarms".

For example:

$MAX_EDR_PROCESSED{$PIPELINE_EDR_LIST[0]} = 88;
$MIN_EDR_PROCESSED{$PIPELINE_EDR_LIST[0]} = 99;

$MAX_EDR_PROCESSED{$PIPELINE_EDR_LIST[1]} = 11;
$MIN_EDR_PROCESSED{$PIPELINE_EDR_LIST[1]} = 22;

The entries must be in the same order that the pipelines are listed in the @PIPELINE_EDR_LIST entry.

Configuring the BRM SPI

Follow these steps after you have installed the BRM SPI and edited the pin_comp.values file. See "Configuring the BRM SPI by Editing the pin_comp.values File".

Important: The pipeline names must match the names configured in the registry.

Important: You need root permission to perform some of these steps.

1. Run the post_install_config.pl script from the installation directory, by default, 
   /var/opt/OV/share/tmp/portal_spi_hpopenview.

   The post_install_config.pl script generates the following data and directories:
   
   - Install_dir/portal_spi: Contains application, logfile monitoring, and process monitoring templates that you will upload to the HP OpenView Management Server.
   - Install_dir/portal_graph: Contains the graph template that you will copy to HP OpenView Performance Manager.
   - Install_dir/portal_perf: Contains performance-related class specifications and the alarmdef file that you will copy to each managed node from which performance data is to be collected.

2. Configure BRM process monitoring and controlling by running the opccfgupld command to upload the SPI into the HP OpenView Management Server:

   /opt/OV/bin/OpC/opccfgupld -add path_to_portal_spi

Important: You need root permission to run this command.

See the HP OpenView documentation for information about the opccfgupld command.
3. Run the **opcdeploy** command to deploy BRM scripts from *Install_dir/bin* to the managed node.

**Important:** You need root permission to run this command.

Deploy the **call_pin_ctl** script from *Install_dir/bin* to */var/opt/OV/bin/instrumentation/bin* on the managed node. Deploy this script to all the managed nodes that have BRM processes installed:

```
opcdeploy -deploy -file ./call_pin_ctl -node node_1
    -targetdir /var/opt/OV/bin/instrumentation
    -trd absolute
```

**Important:** If you have not set the **PERL5LIB** environment variable to point to the Third-Party software install directory, you need to either set the variable or edit the **call_pin_ctl** script before you deploy it. See "Setting Up the **pin_ctl** Utility".

See the HP OpenView documentation for information about the **opcdeploy** command.

4. Deploy the BRM templates from the HP OpenView console Message Source Template window to each managed node. The list and type of templates to be deployed to each node depend on the BRM processes running on each node.

See the HP OpenView documentation for information on how to deploy templates.

5. Configure BRM performance data collection by running the **opcdeploy** command multiple times to deploy the following scripts:

**Note:** Set the **PIN_TMP_DIR** environment variable for all users who run the **perf** scripts. **PIN_TMP_DIR** is the location where all temporary files for performance data collection are created. Because this is used by the roll, run, and feed scripts, it should be the same for all the users.

**a.** Deploy the performance-specific scripts to */var/opt/OV/bin/instrumentation*:

```
portal_perf_run
portal_edr_run
portal_qos_run
```

**b.** Deploy the following performance-specific scripts from *Install_dir/bin* to *BRM_Home/bin* on the managed node from which performance data needs to be collected:

```
portal_perf_feed
portal_perf_snmp_collect
```

6. Deploy the class spec to the directory you specified as **$PERF_LOG** in the **pin_comp.values** file. The spec files are in the **portal_perf** directory.

The spec files use the suffix **.spec**. The number of spec files depends on the number of processes you are monitoring.
7. Deploy the `alarmdef` files to `/var/opt/perf`. The `alarmdef` files are in the `portal_perf` directory.

8. The number of `alarmdef` files depends on the number of processes you are monitoring.

9. Compile the class spec files using the `sdlcomp` command:

   ```bash
dsdlcomp spec_file logset_file
   ```

   where:
   - `spec_file` is the BRM class specification file that contains the metrics that need to be measured.
   - `logset_file` is the binary file where BRM-specific performance data is stored.

   For example:

   ```bash
   sdlcomp portal.spec portal_log
   ```

10. Add the following line to the `perfldb.rc` file in `/var/opt/perf`. The line can go anywhere in the file.

    This entry provides the path to the logset file. This is the same path assigned to `$PERF_LOG` in the `pin_comp.values` file:

    ```
    DATASOURCE=PORTAL LOGFILE=/path/portal_log
    ```

11. Add the following line to the end of the default `alarmdef` file (`/var/opt/perf/alarmdef`) on each managed node:

    ```
    include "path_to_alarmdef_file"
    ```

    For example:

    ```
    include "/var/opt/perf/portal_alarmdef"
    ```

12. Stop and restart the HP OpenView Performance Agent. See the HP OpenView documentation for instructions.

    ```
    Important: You need root permission to run this command.
    ```

13. To enable SNMP data collection, start the SNMP Master Agent. See "Using the SNMP Instrumentation Protocol to Monitor and Control BRM Components".

    ```
    master_agent -l Master_agent_port -x Sub_agent_port &
    ```

    The `Sub_agent_port` must be the same as defined in the registry file.

14. After the SNMP Master Agent starts, start the BRM processes.
Deploying Performance Graphs

After you have completed the post-installation steps, follow these steps to deploy graphs:

2. For each managed node, in the `VPI_GraphsUserPortal_Performance_Graphs.txt` file, change `__NODE_NAME__` to the name of the node.
3. Save and close the file.

BRM-specific graphs are displayed in the HP OpenView Performance Manager.

Collecting Performance Data

After the BRM SPI is configured, you start `dsilog` processes for the following scripts that perform data collection:

- **portal_perf_run**: CPU and memory usage.
- **portal_edr_run**: EDR statistics.
- **portal_qos_run**: Opcode QoS statistics.

The following scripts should be run as BRM user:

- **portal_perf_feed**
- **portal_perf_snmp_collect**

Once started, the scripts run continuously. If any of the scripts are stopped, you need to stop and restart all of them. See "Restarting Data Collection".

1. Run the `portal_perf_run` script for every component listed in the `@PORTALBASE_PROCESS_LIST` and `@PIPELINE_PROCESS_LIST` entries in the `pin_comp.values` file. This includes system components, such as CMs and DMs. This script creates the input file through which data is transferred to the BRM logset. It also starts the `dsilog` processes.

   Use the process names as defined in the `pin_comp.values` file; for example:
   
   ```
   portal_perf_run cm_1
   ```

2. Run the `portal_edr_run` script for every component listed in the `@PIPELINE_EDR_LIST` section in the `pin_comp.values` file. This includes pipelines such as ALL_RATE and PRE_RECYCLE. This script creates the input files through which data is transferred to the BRM logset. It also starts the `dsilog` processes.

   Use the pipeline names as defined in the `pin_comp.values` file; for example:
   
   ```
   portal_edr_run ALL_RATE
   ```

3. Run the `portal_qos_run` script for every component listed in the `OPCODE_NUMBER_LIST` section in the `pin_comp.values` file. This includes opcodes such as PCM_OP_GSM_START_ACCOUNTING and PCM_OP_GSM_STOPACCOUNTING. This script creates the input files through which data is transferred to the BRM logset. It also starts the `dsilog` processes.

   Use the opcode names defined in the `pin_comp.values` file; for example:
Run the `portal_perf_feed` script for every component listed in the @PORTALBASE_PROCESS_LIST and @PIPELINE_PROCESS_LIST sections in the pin_comp.values file. This includes system components, such as CMs and DMs. This script collects and feeds the performance data into the BRM logset.

Use the process names as defined in the pin_comp.values file; for example:

```
portal_perf_feed cm_1
```

5. Run the `portal_perf_snmp_collect` script to collect performance data for EDR and QoS statistics from pipelines:

```
portal_perf_snmp_collect -m Master_agent_host -p Master_agent_port -n Process -i | -s
```

where:
- `-m` is the Master Agent host name.
- `-p` is the Master Agent port number.
- `-n` is the process for which data is collected.

The process name is defined in the registry file; for example:

```
SnmpServer
{
  Port = 21405
  ProcessDescription = ifw
}
```

- `-i` indicates a pipeline process.
- `-s` is the polling interval in seconds in which data is to be collected. The default is 300.

For example, to collect data about Pipeline Manager:

```
portal_perf_snmp_collect -m blrsnse4 -p 12306 -n ifw -i
```

**Restarting Data Collection**

You can restart data collection at any time. Restarting data collection does not reset any historical data. However, restarting data collection can affect alarm generation because the time considered for alarm thresholds is interrupted.

To restart data collection:

1. Stop all `dsilog` and `cat` processes that were started by the `portal_perf_run` script. See the HP OpenView Performance Agent documentation.

2. Run the `portal_perf_run` script for every component listed in the @PORTALBASE_PROCESS_LIST and @PIPELINE_PROCESS_LIST sections in the pin_comp.values file.

3. Run the `portal_edr_run` script for every component listed in the @OPCODE_NUMBER_LIST section in the pin_comp.values file.
4. Run the `portal_perf_feed` script for every component listed in the `@PORTALBASE_PROCESS_LIST` and `@PELINE_PROCESS_LIST` sections in `pin_comp.values` file.

5. Run the `portal_perf_snmp_collect` script for EDR and QoS statistics.

**Editing and Loading the Service Configuration File**

The HP OpenView Service Navigator uses a service configuration file to display the system topology. The service configuration file is an XML file that describes the nodes and processes in your BRM system.

For information about the service configuration file, see the HP OpenView Operations Service Navigator Concepts and Configuration Guide. A sample file is installed with BRM. See "About the Sample Service Configuration File".

To display the BRM system topology in HP OpenView, you need to customize and load the service configuration file. To do so:

1. Install the BRM certified SPI. See "Installing and Configuring the HP OpenView BRM SPI".

2. Edit the `portal_services_HA.xml` file in `/var/opt/OV/share/tmp/portal_spi_hopenview/schema`.

3. Save and close the file.

4. Load the file by using the HP OpenView `opcservice - add` command:

   ```
   opcservice -add service_map_file
   ```

   To assign a service configuration file to a user, use this HP OpenView command:

   ```
   opcservice -assign user service_map_file
   ```

   where, `user` is the HP OpenView user.

   For more information, see the HP OpenView documentation.

**About the Sample Service Configuration File**

The sample service configuration file, `portal_servicesHA.xml`, provides the system topology for a high-availability system.

The sample file shows how to configure the following:

- The actions you want to carry out on the BRM components; for example, stop and start.
- The BRM components you want to display; for example, CMs and DMs.
- Alarm thresholds.

To configure a command in the service configuration file, use the `Action` tag. This example configures the `pin_ctl start` command:

```xml
<Action>
  <Name>start-aaa</Name>
  <Label>Start BRM AAA</Label>
  <Program>
    <Command>pin_ctl start aaa</Command>
    <User>pin</User>
    <ServiceNodes/>
  </Program>
</Action>
```
Editing and Loading the Service Configuration File

For information on pin_ctl parameters, see "pin_ctl" and "Using the pin_ctl Utility to Monitor BRM".

To configure a BRM component in the service configuration file, use the Service tag:

```xml
<Service>
  <Name>portal-dmo-1-p</Name>
  <Label>Portal DMO1 Primary</Label>
  <ActionRef>start-dmo</ActionRef>
  <ActionRef>stop-dmo</ActionRef>
  <ActionRef>status-dmo</ActionRef>
  <NodeRef>nodename</NodeRef>
  <!-- Customer should replace nodename with the node name where DMO exists -->
  <Source>
    <Composition/>
    <ServiceRef>RAC_1</ServiceRef>
  </Source>
  <Source>
    <Composition/>
    <ServiceRef>RAC_2</ServiceRef>
  </Source>
</Service>
```

where:

- **Name** contains the name of the service used internally in this file.
  
  The name of the service should match the name configured in the template that displays the status in the service navigator.
  
  By default, the service name in the template is created by adding Portal to the value given in the pin_comp.values file. For example, if the value is:
  
  PORTALBASE_PROCESS_LIST = ( "cm" );
  
  the service name is **Portal cm**.

  This name is used by other components to create the system architecture hierarchy.

- **Label** is the name of the component that is displayed in HP OpenView.

- **ActionRef** is the list of commands that you can use for this component. The command names are defined in the **Action** tag.

- **NodeRef** is the node name where the component runs.

- **Source** is the list of components that the component connects to. For example, a CM will include connections for DMs.

**Figure 5–2** shows the topology defined in the file:
To configure an alarm in the service configuration file, use the `CalcRule` tag; for example:

```xml
<CalcRule>
    <Name>portal-memory-usage</Name>
    <CalcMultiThreshold>
        <Relative/>
        <Calc>
            <Critical/>
            <Threshold>0.40</Threshold>
            <SetTo>
                <Critical/>
            </SetTo>
        </Calc>
        <Calc>
            <Major/>
            <Threshold>0.35</Threshold>
            <SetTo>
                <Major/>
            </SetTo>
        </Calc>
        <Calc>
            <Minor/>
            <Threshold>0.30</Threshold>
            <SetTo>
                <Minor/>
            </SetTo>
        </Calc>
        <Calc>
            <Warning/>
            <Threshold>0.25</Threshold>
            <SetTo>
                <Warning/>
            </SetTo>
        </Calc>
    </CalcMultiThreshold>
</CalcRule>
```
Note: The service configuration file shows the alarms assigned to components, but the alarms are configured elsewhere. See "Using Alarms".
This document describes the Oracle Communications Billing and Revenue Management (BRM) Operations Management Framework (OMF) instrumented objects. It also includes a simplified, more readable, version of the BRM Management Information Base (MIB).

For more information, see "Using the SNMP Instrumentation Protocol to Monitor and Control BRM Components".

About the BRM MIB

The BRM MIB defines the structure of the managed data in the BRM system. All instrumented objects are included in the MIB except for probes that use the Big Table format.

The BRM MIB is described in the PORTAL-MIB.txt file in BRM_Home/instrumentation.

About the Top-Level Components in the MIB

- log: Provides recent messages from log files.

---

Note: These probes are BigTable probes and are therefore not included in the BRM MIB.

---

- omf: Provides data about the SNMP and HTTP OMF protocols and the Diagnostic Data Handler.
- dat: Provides data about AAA Gateway connection pool and number probability.
- mml: Provides data about the Memory Manager.
- plg: Provides data about EDR statistics.
- fct: Provides data about opcode statistics.
- ppl: Provides data about the Pipeline Manager input controller.

Transaction Manager Probes

The Transaction Manager probes are listed in Table 6–1:
Log File Probes

For information about the log file OIDs, see "Log File OIDs".

This probe collects status and error messages in a circular buffer. It collects all messages, including debugging information, independent of log file settings.

By default, the circular buffer size is set to 1000 messages. You can change that setting in the registry. See "Getting Recent Pipeline Log File Entries".

You can display this data in the HTTP protocol. It is also included in the file created by the Diagnostic Data Handler.

Note: Log file probes as shown in Table 6–2 are Big Table probes and are therefore not included in the BRM MIB.

Operations Management Framework Probes

Operations Management Framework (OMF) includes the following probes:

- **http**: See "HTTP Probes".
- **snmp**: See "SNMP Probes".
- **diagnosticDataHandler**: See "Diagnostic Data Handler Probe".

For information about the OMF OIDs, see "OMF OIDs".

HTTP Probes

The HTTP probes are listed in Table 6–3:

Table 6–3  **HTTP Probes**

<table>
<thead>
<tr>
<th>Probe</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>port</td>
<td>The port number for the HTTP listener as configured in the Port registry entry.</td>
</tr>
<tr>
<td>styleSheetName</td>
<td>The XML style sheet as configured in the StyleSheetFile registry entry.</td>
</tr>
<tr>
<td>portalLogoName</td>
<td>The file name of the logo used on the HTTP display, as defined in the PortalLogoFile registry entry.</td>
</tr>
<tr>
<td>numHttpRequests</td>
<td>The number of requests to get data from the HTTP server.</td>
</tr>
</tbody>
</table>
SNMP Probes

The probes listed in Table 6–4 show data about:

- The SNMP instrumentation configuration.
- SNMP activity; for example, the number of GET requests.

The SNMP table includes SNMP activity about specific instrumented objects. Figure 6–1 shows an example:

Figure 6–1 Sample SNMP Table

<table>
<thead>
<tr>
<th>Snmp Mib Table Index</th>
<th>Mib Table Name</th>
<th>Snmp Mib Table Oid</th>
<th>Number of 'GET' Requests</th>
<th>Number of 'GETNEXT' Requests</th>
<th>Number of 'SET' Requests</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ProcessTable</td>
<td>1.3.6.1.4.1.3512.1.101.1</td>
<td>5</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>RegistryTable</td>
<td>1.3.6.1.4.1.3512.1.102.1</td>
<td>5</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>diagnosticDataHandlerGroup</td>
<td>1.3.6.1.4.1.3512.1.4.3.1.1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>http.Info</td>
<td>1.3.6.1.4.1.3512.1.4.1.1.1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>snmp.Info</td>
<td>1.3.6.1.4.1.3512.1.4.2.1.1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>snmpTable</td>
<td>1.3.6.1.4.1.3512.1.4.2.2.1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>inputControllerGroup</td>
<td>1.3.6.1.4.1.3512.1.11.1.1.1</td>
<td>6</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>processingTimeTable</td>
<td>1.3.6.1.4.1.3512.1.11.2.1.1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>statisticGroup</td>
<td>1.3.6.1.4.1.3512.1.9.1.1.1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>statisticTable</td>
<td>1.3.6.1.4.1.3512.1.9.1.2.1</td>
<td>4</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 6–4 SNMP Probes

<table>
<thead>
<tr>
<th>Probe</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>port</td>
<td>The port number for the SNMP sub-agent as configured in the Port registry entry.</td>
</tr>
<tr>
<td>numSnmpRequests</td>
<td>The number of SNMP requests received.</td>
</tr>
<tr>
<td>snmpTableEntry</td>
<td>A table showing OMF instrumented objects and SNMP Get, GetNext, and Set requests used for those objects.</td>
</tr>
<tr>
<td>snmpTableOid</td>
<td>The OID of the object for which SNMP data is shown.</td>
</tr>
<tr>
<td>snmpTableName</td>
<td>The object name in the MIB; for example, ProcessTable, statisticGroup, or diagnosticDataHandlerGroup.</td>
</tr>
<tr>
<td>numSnmpGetRequests</td>
<td>The number of SNMP Get requests received.</td>
</tr>
<tr>
<td>numSnmpGetNextRequests</td>
<td>The number of SNMP GetNext requests received.</td>
</tr>
<tr>
<td>numSnmpSetRequests</td>
<td>The number of SNMP Set requests received.</td>
</tr>
</tbody>
</table>

Diagnostic Data Handler Probe

Use the probe listed in Table 6–5 to get a snapshot from the Diagnostic Data Handler.

For information about the Diagnostic Data Handler, see "Using the Diagnostic Data Handler to Get OMF Diagnostic Data".
Multi-Threaded Framework

Multi-threaded framework (MTF) includes the following probes:

- **connectionConfigurations**: See "Connection Configurations Probes".
- **threadCheckManager**: See "Thread Check Manager Probes".
- **stateManager**: See "MTF State Manager Probes".

For information about the MTF OIDs, see "MTF OIDs".

Connection Configurations Probes

The connection configurations probes include:

- **connectionConfigurationsGroup**: These probes provide data about the Oracle DM configuration settings. See "Connection Configurations Probes".
- **dmoTable**: These probes provide data about the configured Oracle DMs. See "Oracle DM Server Probes".

Connection Configurations Probes

These probes shown in Figure 6–2 and listed in Table 6–6 provide data about the available Oracle DM servers.

Figure 6–2  Connection Configuration Probes

<table>
<thead>
<tr>
<th>Connection Configurations:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Total number of configured DMO Servers</strong></td>
</tr>
<tr>
<td><strong>Number of available DMO Servers</strong></td>
</tr>
<tr>
<td><strong>Peer Configured</strong></td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>yes</td>
</tr>
</tbody>
</table>

Table 6–6  Connection Configurations Probes

<table>
<thead>
<tr>
<th>Probe</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dmoServerConfigured</td>
<td>The number of Oracle DM servers configured.</td>
</tr>
<tr>
<td>dmoServerAvailable</td>
<td>The number of available Oracle DM servers.</td>
</tr>
<tr>
<td>peerConfigured</td>
<td>Specifies if a peer is configured.</td>
</tr>
<tr>
<td>addServerInstance</td>
<td>Rebalances DM connections when you restart a failed DM or when you add a new DM to the DM pool. When you use this to add a new DM, provide the host name, port number, and maximum connections it can process. Note: This probe is a write-only probe. It is not displayed in the HTTP interface.</td>
</tr>
</tbody>
</table>
Oracle DM Server Probes
The probes shown in Figure 6–3 and listed in Table 6–7 provide data about each Oracle DM.

Figure 6–3  Oracle DM Server Probes

DMO Server Table:

<table>
<thead>
<tr>
<th>DMO Server</th>
<th>DMO Server Host Name</th>
<th>DMO Server Port Number</th>
<th>Maximum Connections</th>
<th>Active Connections</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>sunra</td>
<td>13008</td>
<td>64</td>
<td>11</td>
</tr>
</tbody>
</table>

Table 6–7  Oracle DM Server Probes

<table>
<thead>
<tr>
<th>Probe</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dmoHost</td>
<td>The Oracle DM host name.</td>
</tr>
<tr>
<td>dmoPort</td>
<td>The Oracle DM port number.</td>
</tr>
<tr>
<td>dmoMaxConn</td>
<td>The number of maximum connections that the Oracle DM can handle at any time.</td>
</tr>
<tr>
<td>dmoActiveConn</td>
<td>Active connections for this Oracle DM.</td>
</tr>
</tbody>
</table>

Thread Check Manager Probes
Thread Check Manager monitors threads and reports failures as part of a high-availability configuration.

Thread Check Manager Probes
The probe listed in Table 6–8 shows the current ThreadCheckManager registry settings.

Table 6–8  Thread Check Manager Probes

<table>
<thead>
<tr>
<th>Probe</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>requestInterval</td>
<td>The time in seconds to poll for thread responsiveness.</td>
</tr>
</tbody>
</table>

Thread Chart Probes
The probes shown in Figure 6–4 and listed in Table 6–9 provide data about active threads.
Figure 6–4  Thread Chart Probes

Thread Chart Table:

<table>
<thead>
<tr>
<th>Thread Chart Table Index</th>
<th>Name</th>
<th>Id</th>
<th>Status</th>
<th>Count</th>
<th>Time last updated</th>
<th>Message of last update</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>tmosMgr.TmosDataManager.AcceptorPool</td>
<td>14</td>
<td>HEALTHY</td>
<td>22</td>
<td>06/20/06 12:02:59</td>
<td>ClientConnection</td>
</tr>
<tr>
<td>2</td>
<td>tmosMgr.TmosDataManager.ConnectorPool</td>
<td>23</td>
<td>HEALTHY</td>
<td>22</td>
<td>06/20/06 12:02:59</td>
<td>Default Gateway</td>
</tr>
<tr>
<td>3</td>
<td>tmosMgr.TmosDataManager.AcceptorPool</td>
<td>22</td>
<td>HEALTHY</td>
<td>22</td>
<td>06/20/06 12:02:59</td>
<td>ClientGateway</td>
</tr>
<tr>
<td>4</td>
<td>tmosMgr.TmosDataManager.AcceptorPool</td>
<td>21</td>
<td>HEALTHY</td>
<td>22</td>
<td>06/20/06 12:03:00</td>
<td>ClientConnection</td>
</tr>
</tbody>
</table>

Table 6–9  Thread Chart Probes

<table>
<thead>
<tr>
<th>Probe</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>threadChartThreadName</td>
<td>The name of the thread.</td>
</tr>
<tr>
<td>threadChartThreadId</td>
<td>The thread ID.</td>
</tr>
<tr>
<td>threadChartStatus</td>
<td>The status, HEALTHY or UNHEALTHY.</td>
</tr>
<tr>
<td>threadChartTime</td>
<td>The timestamp when the thread data was collected.</td>
</tr>
<tr>
<td>threadChartCount</td>
<td>The number of times the thread has been checked for responsiveness.</td>
</tr>
<tr>
<td>threadChartMessage</td>
<td>The type of thread being monitored:</td>
</tr>
<tr>
<td>■ ClientConnection</td>
<td></td>
</tr>
<tr>
<td>■ ClientGateway</td>
<td></td>
</tr>
<tr>
<td>■ DataMigrator</td>
<td></td>
</tr>
<tr>
<td>■ WorkerDataMigratorThread</td>
<td></td>
</tr>
<tr>
<td>■ PoidIdManager</td>
<td></td>
</tr>
</tbody>
</table>

MTF State Manager Probes

The probes shown in Figure 6–5 and listed in Table 6–10 provide data about the MTF framework.

Figure 6–5  MTF State Manager Probes

MTF State Manager:

<table>
<thead>
<tr>
<th>State</th>
<th>Running</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of open connections</td>
<td>0</td>
</tr>
</tbody>
</table>
AAA Manager Connection and Number Portability Probes

Use these probes to manage AAA Manager connections and number portability.

- **connectionPool**: See "Connection Pool Probes".
- **connectionMonitor**: See "Connection Monitor Probes".
- **numberPortability**: These probes provide data for number portability. See "Number Portability Probes".

### Connection Pool Probes

Use the probe in Table 6–11 to rebalance connections in the AAA Gateway Manager connection pool. See "Rebalancing CM Connections" in *BRM AAA Gateway Manager*.

You can also rebalance connections by using the `pin_ctl` utility. See "pin_ctl".

For information about the connection pool OIDs, see "AAA Manager Connection and Number Portability OIDs".

### Table 6–10  MTF State Manager Probes

<table>
<thead>
<tr>
<th>Probe</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>state</td>
<td>The current execution state of the MTF framework:</td>
</tr>
<tr>
<td></td>
<td>- <strong>Initial</strong>: The system is starting up.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Running</strong>: Normal operation. The MTF framework is accepting client requests.</td>
</tr>
<tr>
<td></td>
<td>- <strong>SuspendInProgress</strong>: The MTF framework is in the process of shutting down client connections. No new connections are allowed in this state.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Suspended</strong>: The MTF framework is not accepting connections so that a shutdown or switchover can occur.</td>
</tr>
<tr>
<td>numOfConnection</td>
<td>The number of current client connections.</td>
</tr>
</tbody>
</table>

### Connection Monitor Probes

Use the probe listed in Table 6–12 to send a Diameter Disconnect Peer Request (DPR) message. See "Supported Diameter Messages" in *BRM AAA Gateway Manager*.

For information about the connection monitor OIDs, see "AAA Manager Connection and Number Portability OIDs".

### Table 6–11  Connection Pool Probe

<table>
<thead>
<tr>
<th>Probe</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rebalanceConnection</td>
<td>Rebalances all cached AAA Gateway CM connections.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong>: This probe is a write-only probe. It is not displayed in the HTTP interface.</td>
</tr>
</tbody>
</table>

### Number Portability Probes

Use the probes listed in Table 6–13 to print, load, and append the number portability records. See "Setting Up Number Portability" in *BRM Configuring Pipeline Rating and Discounting*. 

### Table 6–12  Connection Monitor Probe

<table>
<thead>
<tr>
<th>Probe</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sendDPR</td>
<td>Sends a Disconnect Peer Request (DPR) message.</td>
</tr>
</tbody>
</table>
For more information about the number portability OIDs, see "AAA Manager Connection and Number Portability OIDs".

### Table 6–13 Number Portability Probes

<table>
<thead>
<tr>
<th>Probe</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>reload</td>
<td>Reloads the data from the number portability file.</td>
</tr>
<tr>
<td></td>
<td>Value: True or False.</td>
</tr>
<tr>
<td></td>
<td>Set to True to continue to use the old number portability data if the reload operation fails.</td>
</tr>
<tr>
<td>deltaLoad</td>
<td>Appends the additional number portability data in the memory.</td>
</tr>
<tr>
<td></td>
<td>Location: Pipeline_Home</td>
</tr>
<tr>
<td></td>
<td>Value: File_name</td>
</tr>
<tr>
<td></td>
<td>Where File_name is the name of the delta file.</td>
</tr>
<tr>
<td>printData</td>
<td>Displays the in-memory number portability data or saves it in a file.</td>
</tr>
<tr>
<td></td>
<td>Location: Pipeline_Home</td>
</tr>
<tr>
<td></td>
<td>Value: NULL or File_name</td>
</tr>
<tr>
<td></td>
<td>Set to NULL to display the number portability data on the screen.</td>
</tr>
<tr>
<td></td>
<td>Set to the File_name to save the number portability data in the file.</td>
</tr>
</tbody>
</table>

### Memory Monitor Probes

The probe in Table 6–14 provide data on memory allocation.

For information about the memory monitor OIDs, see "Memory Monitor OIDs".

### Table 6–14 Memory Monitor Probes

<table>
<thead>
<tr>
<th>Probe</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>totalMemoryAllocated</td>
<td>The total memory allocated to the process.</td>
</tr>
</tbody>
</table>

### Pipeline Statistics Probes

You can get EDR throughput statistics for individual pipelines.

For information about the pipeline statistics OIDs, see "Statistics OIDs".

### Throughput Statistics Probes

The probes shown in Figure 6–6 and listed in Table 6–15 provide data about the total EDRs processed by a pipeline.

### Figure 6–6 Throughput Statistics Probes

<table>
<thead>
<tr>
<th>Throughput Statistics:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Total EDR Count (real-time)</td>
<td>210</td>
</tr>
<tr>
<td>Total EDR Count (after transaction ended)</td>
<td>120</td>
</tr>
<tr>
<td>Accumulated Txn Processing Time (sec)</td>
<td>3</td>
</tr>
<tr>
<td>Total Txn Count</td>
<td>40</td>
</tr>
</tbody>
</table>
Opcode Latency Probes

Table 6–15  Throughput Statistics Probes

<table>
<thead>
<tr>
<th>Probe</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>totalEdrCount</td>
<td>The total number of EDRs processed since startup, independent of any transaction, including EDRs processed in canceled and rolled-back transactions. This value is incremented for each EDR.</td>
</tr>
<tr>
<td>totalTxnEdrCount</td>
<td>The total number of EDRs processed in a transaction. This value is incremented after a transaction finishes.</td>
</tr>
<tr>
<td>totalProcTime</td>
<td>The total processing time in seconds since startup.</td>
</tr>
<tr>
<td>totalTxnCount</td>
<td>The number of transactions processed since startup.</td>
</tr>
</tbody>
</table>

Last Throughputs Probes

The probes shown in Figure 6–7 and listed in Table 6–16 provide data about EDR throughput (in a transaction) for a pipeline.

Figure 6–7  Last Throughputs Probes

<table>
<thead>
<tr>
<th>Index</th>
<th>Throughput (edrs/sec)</th>
<th>Timestamp</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1053</td>
<td>16.06.2006 11:49:21</td>
</tr>
<tr>
<td>2</td>
<td>1025</td>
<td>16.06.2006 11:49:21</td>
</tr>
<tr>
<td>3</td>
<td>1076</td>
<td>16.06.2006 11:49:22</td>
</tr>
<tr>
<td>4</td>
<td>1055</td>
<td>16.06.2006 11:49:22</td>
</tr>
<tr>
<td>5</td>
<td>1476</td>
<td>16.06.2006 11:49:22</td>
</tr>
<tr>
<td>6</td>
<td>1057</td>
<td>16.06.2006 11:49:22</td>
</tr>
<tr>
<td>7</td>
<td>948</td>
<td>16.06.2006 11:49:22</td>
</tr>
<tr>
<td>8</td>
<td>1149</td>
<td>16.06.2006 11:49:23</td>
</tr>
<tr>
<td>9</td>
<td>1439</td>
<td>16.06.2006 11:49:23</td>
</tr>
<tr>
<td>10</td>
<td>1147</td>
<td>16.06.2006 11:49:23</td>
</tr>
</tbody>
</table>

Table 6–16  Last Throughputs Probes

<table>
<thead>
<tr>
<th>Probe</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>throughput</td>
<td>The number of EDRs per second.</td>
</tr>
<tr>
<td>throughputTimestamp</td>
<td>The timestamp when the throughput measurement occurred.</td>
</tr>
</tbody>
</table>

Opcode Latency Probes

These probes provide data about the performance of the FCT_Opcode module.

For information about the opcode latency OIDs, see "fct Probe OIDs".

Opcode Plug-In Probes

The probes listed in Table 6–17 provide statistics on the opcodePlugInGroup probes.
Last Latency Probes

The probes listed in Table 6–18 provide data on opcode latency.

<table>
<thead>
<tr>
<th>Probe</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>latency</td>
<td>The processing time for opcodes.</td>
</tr>
<tr>
<td>latencyTimestamp</td>
<td>The timestamp when the latency measurement occurred.</td>
</tr>
</tbody>
</table>

Input Probes

These probes provide data about pipeline input processing.

For information about the input OIDs, see "Input Controller probes".

Input Controller Probes

The probes shown in Figure 6–8 and listed in Table 6–19 show processing time for real-time pipeline events.

Figure 6–8 Input Controller Probes

Input Controller:

Maximum Processing Time (ns) 3428982 (16.06.2006 11:49:18)

Minimum Processing Time (ns) 1091240 (16.06.2006 11:49:18)

<table>
<thead>
<tr>
<th>Probe</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>maxProcessingTime</td>
<td>The longest processing time for a real-time event.</td>
</tr>
<tr>
<td>minProcessingTime</td>
<td>The shortest processing time for a real-time event.</td>
</tr>
</tbody>
</table>

Last Processing Times Probes

The probes shown in Figure 6–9 and listed in Table 6–20 show pipeline processing times for the last 10 real-time events.
The following sections describe the Portal object IDs (POIDs) in order of the ID number. The same information is available in the BRM MIB, but not in order. However, the BRM MIB includes details about every instrumented object; for example:

```
-- 1.3.6.1.4.1.3512.1.5.5.1.1.2
numOfConnection OBJECT-TYPE
  SYNTAX Integer32
  MAX-ACCESS read-only
  STATUS current
  DESCRIPTION "Number of open connections"
  ::= { stateManagerGroupEntry 2 }
```

To derive the complete OIDs, in the following sections, replace the:

- `process_id` with the process ID of the SNMP sub-agent process that is registered with the SNMP master-agent.
- `registry_id` with the registry ID of the SNMP sub-agent process that is registered with the SNMP master-agent.

For more information, see "About Dynamic Object IDs".

### MIB Prefix

The BRM MIB uses the industry-standard `internet.private` root, with 3512 as the start of the BRM sub-root. The only branch following `portal` (3512) is `components` (1), so the prefix for all entries is:
SNMP Object IDs

iso.org.dod.internet.private.enterprise.portal.components

or:
1.3.6.1.4.1.3512.1

Transaction Manager OIDs

txm
1.3.6.1.4.1.3512.1.2

txmanage
1.3.6.1.4.1.3512.1.2.1

transactionManagerGroup
1.3.6.1.4.1.3512.1.2.1.1

transactionManagerGroupEntry
1.3.6.1.4.1.3512.1.2.1.1.1

batchSizeLimit
1.3.6.1.4.1.3512.1.2.1.1.1.1

loggingOff
1.3.6.1.4.1.3512.1.2.1.1.1.2

transactionManagerGroupIndex
1.3.6.1.4.1.3512.1.2.1.1.1.3

Log File OIDs

Because log file probes use the Big Table format, they are not included in the MIB file.

log
1.3.6.1.4.1.3512.1.3

OMF OIDs

omf
1.3.6.1.4.1.3512.1.4

http
1.3.6.1.4.1.3512.1.4.1

httpInfo
1.3.6.1.4.1.3512.1.4.1.1

httpInfoEntry
1.3.6.1.4.1.3512.1.4.1.1.1

port
1.3.6.1.4.1.3512.1.4.1.1.1.1

styleSheetName
1.3.6.1.4.1.3512.1.4.1.1.1.2

portalLogoName
1.3.6.1.4.1.3512.1.4.1.1.1.3

numHttpRequests
SNMP Object IDs

httpInfoIndex
1.3.6.1.4.1.3512.1.4.1.1.1.4

httpInfoIndex
1.3.6.1.4.1.3512.1.4.1.1.1.5

snmp
1.3.6.1.4.1.3512.1.4.2

snmpInfo
1.3.6.1.4.1.3512.1.4.2.1

snmpInfoEntry
1.3.6.1.4.1.3512.1.4.2.1.1

port
1.3.6.1.4.1.3512.1.4.2.1.1.1

numSnmpRequests
1.3.6.1.4.1.3512.1.4.2.1.1.2

snmpInfoIndex
1.3.6.1.4.1.3512.1.4.2.1.1.3

snmpTable
1.3.6.1.4.1.3512.1.4.2.2

snmpTableEntry
1.3.6.1.4.1.3512.1.4.2.2.1

snmpTableIndex
1.3.6.1.4.1.3512.1.4.2.2.1.1

snmpTableOid
1.3.6.1.4.1.3512.1.4.2.2.1.2

snmpTableName
1.3.6.1.4.1.3512.1.4.2.2.1.3

numSnmpGetRequests
1.3.6.1.4.1.3512.1.4.2.2.1.4

numSnmpGetNextRequests
1.3.6.1.4.1.3512.1.4.2.2.1.5

numSnmpSetRequests
1.3.6.1.4.1.3512.1.4.2.2.1.6

diagnosticDataHandler
1.3.6.1.4.1.3512.1.4.3

diagnosticDataHandlerGroup
1.3.6.1.4.1.3512.1.4.3.1

diagnosticDataHandlerGroupEntry
1.3.6.1.4.1.3512.1.4.3.1.1

createSnapshot
1.3.6.1.4.1.3512.1.4.3.1.1.1
SNMP Object IDs

diagnosticDataHandlerGroupIndex
1.3.6.1.4.1.3512.1.4.3.1.1.2

MTF OIDs

mtf
1.3.6.1.4.1.3512.1.5

highAvailabilityManager
1.3.6.1.4.1.3512.1.5.1

connectionConfigurations
1.3.6.1.4.1.3512.1.5.2

connectionConfigurationsGroup
1.3.6.1.4.1.3512.1.5.2.1

connectionConfigurationsGroupEntry
1.3.6.1.4.1.3512.1.5.2.1.1

dmoServerConfigured
1.3.6.1.4.1.3512.1.5.2.1.1.1

dmoServerAvailable
1.3.6.1.4.1.3512.1.5.2.1.1.2

peerConfigured
1.3.6.1.4.1.3512.1.5.2.1.1.3

refreshConnections
1.3.6.1.4.1.3512.1.5.2.1.1.4

addServerInstance
1.3.6.1.4.1.3512.1.5.2.1.1.5

connectionConfigurationsGroupIndex
1.3.6.1.4.1.3512.1.5.2.1.1.6

dmoTable
1.3.6.1.4.1.3512.1.5.2.2

dmoEntry
1.3.6.1.4.1.3512.1.5.2.2.1

dmoIndex
1.3.6.1.4.1.3512.1.5.2.2.1.1

dmoHost
1.3.6.1.4.1.3512.1.5.2.2.1.2

dmoPort
1.3.6.1.4.1.3512.1.5.2.2.1.3

dmoMaxConn
1.3.6.1.4.1.3512.1.5.2.2.1.4

dmoActiveConn
1.3.6.1.4.1.3512.1.5.2.2.1.5
dmoState
1.3.6.1.4.1.3512.1.5.2.2.1.6

threadCheckManager
1.3.6.1.4.1.3512.1.5.3

threadCheckManagerGroup
1.3.6.1.4.1.3512.1.5.3.1

threadCheckManagerGroupEntry
1.3.6.1.4.1.3512.1.5.3.1.1

requestInterval
1.3.6.1.4.1.3512.1.5.3.1.1.1

notifyFailure
1.3.6.1.4.1.3512.1.5.3.1.1.2

threadCheckManagerGroupIndex
1.3.6.1.4.1.3512.1.5.3.1.1.3

threadCheckManagerTable
1.3.6.1.4.1.3512.1.5.3.2

threadChartTableEntry
1.3.6.1.4.1.3512.1.5.3.2.1

threadChartTableIndex
1.3.6.1.4.1.3512.1.5.3.2.1.1

threadChartThreadName
1.3.6.1.4.1.3512.1.5.3.2.1.2

threadChartThreadId
1.3.6.1.4.1.3512.1.5.3.2.1.3

threadChartStatus
1.3.6.1.4.1.3512.1.5.3.2.1.4

threadChartTime
1.3.6.1.4.1.3512.1.5.3.2.1.5

threadChartCount
1.3.6.1.4.1.3512.1.5.3.2.1.6

threadChartMessage
1.3.6.1.4.1.3512.1.5.3.2.1.7

threadPool
1.3.6.1.4.1.3512.1.5.4

threadPoolAcceptorGroup
1.3.6.1.4.1.3512.1.5.4.1

threadPoolAcceptorGroupEntry
1.3.6.1.4.1.3512.1.5.4.1.1

acceptorThreadPoolType
1.3.6.1.4.1.3512.1.5.4.1.1.1
acceptorThreadCount
1.3.6.1.4.1.3512.1.5.4.1.1.2

acceptorReactorType
1.3.6.1.4.1.3512.1.5.4.1.1.3

threadPoolAcceptorGroupIndex
1.3.6.1.4.1.3512.1.5.4.1.1.4

threadPoolAcceptorTable
1.3.6.1.4.1.3512.1.5.4.2

threadPoolAcceptorTableEntry
1.3.6.1.4.1.3512.1.5.4.2.1

acceptorConnectionThreadNumber
1.3.6.1.4.1.3512.1.5.4.2.1.1

acceptorAccumulatedConnectionCount
1.3.6.1.4.1.3512.1.5.4.2.1.2

acceptorCurrentConnectionCount
1.3.6.1.4.1.3512.1.5.4.2.1.3

threadPoolConnectorGroup
1.3.6.1.4.1.3512.1.5.4.3

threadPoolConnectorGroupEntry
1.3.6.1.4.1.3512.1.5.4.3.1

connectorThreadPoolType
1.3.6.1.4.1.3512.1.5.4.3.1.1

connectorThreadCount
1.3.6.1.4.1.3512.1.5.4.3.1.2

connectorReactorType
1.3.6.1.4.1.3512.1.5.4.3.1.3

threadPoolConnectorGroupIndex
1.3.6.1.4.1.3512.1.5.4.3.1.4

threadPoolConnectorTable
1.3.6.1.4.1.3512.1.5.4.4

threadPoolConnectorTableEntry
1.3.6.1.4.1.3512.1.5.4.4.1

connectorConnectionThreadNumber
1.3.6.1.4.1.3512.1.5.4.4.1.1

connectorAccumulatedConnectionCount
1.3.6.1.4.1.3512.1.5.4.4.1.2

connectorCurrentConnectionCount
1.3.6.1.4.1.3512.1.5.4.4.1.3

threadPoolPrivatePeerGroup
1.3.6.1.4.1.3512.1.5.4.5
threadPoolPrivatePeerGroupEntry
1.3.6.1.4.1.3512.1.5.4.5.1

  privatePeerThreadPoolType
  1.3.6.1.4.1.3512.1.5.4.5.1.1

  privatePeerThreadCount
  1.3.6.1.4.1.3512.1.5.4.5.1.2

  privatePeerReactorType
  1.3.6.1.4.1.3512.1.5.4.5.1.3

threadPoolPrivatePeerGroupIndex
1.3.6.1.4.1.3512.1.5.4.5.1.4

threadPoolPrivatePeerTable
1.3.6.1.4.1.3512.1.5.4.6

threadPoolPrivatePeerTableEntry
1.3.6.1.4.1.3512.1.5.4.6.1

  privatePeerConnectionThreadNumber
  1.3.6.1.4.1.3512.1.5.4.6.1.1

  privatePeerAccumulatedConnectionCount
  1.3.6.1.4.1.3512.1.5.4.6.1.2

  privatePeerCurrentConnectionCount
  1.3.6.1.4.1.3512.1.5.4.6.1.3

qosOpcodeConfigTable
1.3.6.1.4.1.3512.1.5.4.7

qosOpcodeConfigTableEntry
1.3.6.1.4.1.3512.1.5.4.7.1

  qosOpcodeConfigIndex
  1.3.6.1.4.1.3512.1.5.4.7.1.1

  qosOpcodeConfigName
  1.3.6.1.4.1.3512.1.5.4.7.1.2

  qosOpcodeConfigBucket1
  1.3.6.1.4.1.3512.1.5.4.7.1.3

  qosOpcodeConfigBucket2
  1.3.6.1.4.1.3512.1.5.4.7.1.4

  qosOpcodeConfigBucket3
  1.3.6.1.4.1.3512.1.5.4.7.1.5

  qosOpcodeConfigBucket4
  1.3.6.1.4.1.3512.1.5.4.7.1.6

  qosOpcodeConfigBucket5
  1.3.6.1.4.1.3512.1.5.4.7.1.7

  qosOpcodeConfigBucket6
  1.3.6.1.4.1.3512.1.5.4.7.1.8
SNMP Object IDs

qosOpcodeConfigBucket7
1.3.6.1.4.1.3512.1.5.4.7.1.9

qosOpcodeConfigBucket8
1.3.6.1.4.1.3512.1.5.4.7.1.10

qosOpcodeTable
1.3.6.1.4.1.3512.1.5.4.8

qosOpcodeTableEntry
1.3.6.1.4.1.3512.1.5.4.8.1

qosOpcodeIndex
1.3.6.1.4.1.3512.1.5.4.8.1.1

qosOpcodeName
1.3.6.1.4.1.3512.1.5.4.8.1.2

qosTotalSamples
1.3.6.1.4.1.3512.1.5.4.8.1.3

qosAverageProcessingTime
1.3.6.1.4.1.3512.1.5.4.8.1.4

qosMinProcessingTime
1.3.6.1.4.1.3512.1.5.4.8.1.5

qosMaxProcessingTime
1.3.6.1.4.1.3512.1.5.4.8.1.6

qosThroughput
1.3.6.1.4.1.3512.1.5.4.8.1.7

qosOpcodeBucketCount1
1.3.6.1.4.1.3512.1.5.4.8.1.8

qosOpcodeBucketPercent1
1.3.6.1.4.1.3512.1.5.4.8.1.9

qosOpcodeBucketCount2
1.3.6.1.4.1.3512.1.5.4.8.1.10

qosOpcodeBucketPercent2
1.3.6.1.4.1.3512.1.5.4.8.1.11

qosOpcodeBucketCount3
1.3.6.1.4.1.3512.1.5.4.8.1.12

qosOpcodeBucketPercent3
1.3.6.1.4.1.3512.1.5.4.8.1.13

qosOpcodeBucketCount4
1.3.6.1.4.1.3512.1.5.4.8.1.14

qosOpcodeBucketPercent4
1.3.6.1.4.1.3512.1.5.4.8.1.15

qosOpcodeBucketCount5
1.3.6.1.4.1.3512.1.5.4.8.1.16
### SNMP Object IDs

**BRM OMF Instrumented Objects**

<table>
<thead>
<tr>
<th>Object ID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>qosOpcodeBucketPercent5</td>
<td>1.3.6.1.4.1.3512.1.5.4.8.1.17</td>
</tr>
<tr>
<td>qosOpcodeBucketCount6</td>
<td>1.3.6.1.4.1.3512.1.5.4.8.1.18</td>
</tr>
<tr>
<td>qosOpcodeBucketPercent6</td>
<td>1.3.6.1.4.1.3512.1.5.4.8.1.19</td>
</tr>
<tr>
<td>qosOpcodeBucketCount7</td>
<td>1.3.6.1.4.1.3512.1.5.4.8.1.20</td>
</tr>
<tr>
<td>qosOpcodeBucketPercent7</td>
<td>1.3.6.1.4.1.3512.1.5.4.8.1.21</td>
</tr>
<tr>
<td>qosOpcodeBucketCount8</td>
<td>1.3.6.1.4.1.3512.1.5.4.8.1.22</td>
</tr>
<tr>
<td>qosOpcodeBucketPercent8</td>
<td>1.3.6.1.4.1.3512.1.5.4.8.1.23</td>
</tr>
<tr>
<td>qosResetTable</td>
<td>1.3.6.1.4.1.3512.1.5.4.9</td>
</tr>
<tr>
<td>qosResetEntry</td>
<td>1.3.6.1.4.1.3512.1.5.4.9.1</td>
</tr>
<tr>
<td>qosReset</td>
<td>1.3.6.1.4.1.3512.1.5.4.9.1.1</td>
</tr>
<tr>
<td>qosResetIndex</td>
<td>1.3.6.1.4.1.3512.1.5.4.9.1.2</td>
</tr>
<tr>
<td>threadPoolStatisticsGroup</td>
<td>1.3.6.1.4.1.3512.1.5.4.10</td>
</tr>
<tr>
<td>threadPoolStatisticsGroupEntry</td>
<td>1.3.6.1.4.1.3512.1.5.4.10.1</td>
</tr>
<tr>
<td>poolTotalRequestCount</td>
<td>1.3.6.1.4.1.3512.1.5.4.10.1.1</td>
</tr>
<tr>
<td>poolThroughput</td>
<td>1.3.6.1.4.1.3512.1.5.4.10.1.2</td>
</tr>
<tr>
<td>threadPoolStatisticsGroupIndex</td>
<td>1.3.6.1.4.1.3512.1.5.4.10.1.3</td>
</tr>
<tr>
<td>perThreadStatisticsTable</td>
<td>1.3.6.1.4.1.3512.1.5.4.11</td>
</tr>
<tr>
<td>perThreadStatisticsTableEntry</td>
<td>1.3.6.1.4.1.3512.1.5.4.11.1</td>
</tr>
<tr>
<td>threadIndex</td>
<td>1.3.6.1.4.1.3512.1.5.4.11.1.1</td>
</tr>
<tr>
<td>threadTotalRequestCount</td>
<td>1.3.6.1.4.1.3512.1.5.4.11.1.2</td>
</tr>
</tbody>
</table>
threadThroughput
1.3.6.1.4.1.3512.1.5.4.11.1.3

threadOverloadIntervalRequestCount
1.3.6.1.4.1.3512.1.5.4.11.1.4

overloadDetectionConfigGroup
1.3.6.1.4.1.3512.1.5.4.12

overloadDetectionConfigGroupEntry
1.3.6.1.4.1.3512.1.5.4.12.1

overloadDetectionEnabled
1.3.6.1.4.1.3512.1.5.4.12.1.1

overloadDetectionRate
1.3.6.1.4.1.3512.1.5.4.12.1.2

overloadDetectionInterval
1.3.6.1.4.1.3512.1.5.4.12.1.3

overloadDetectionThreshold
1.3.6.1.4.1.3512.1.5.4.12.1.4

overloadDetectionConfigGroupIndex
1.3.6.1.4.1.3512.1.5.4.12.1.5

stateManager
1.3.6.1.4.1.3512.1.5.5

stateManagerGroup
1.3.6.1.4.1.3512.1.5.5.1

stateManagerGroupEntry
1.3.6.1.4.1.3512.1.5.5.1.1

state
1.3.6.1.4.1.3512.1.5.5.1.1.1

numOfConnection
1.3.6.1.4.1.3512.1.5.5.1.1.2

stateManagerGroupIndex
1.3.6.1.4.1.3512.1.5.5.1.1.3

Data Object OIDs

oal
1.3.6.1.4.1.3512.1.6

storageManager
1.3.6.1.4.1.3512.1.6.1

objectTable
1.3.6.1.4.1.3512.1.6.1.1

objectTableEntry
1.3.6.1.4.1.3512.1.6.1.1.1
objectIndex
1.3.6.1.4.1.3512.1.6.1.1.1.1

objectName
1.3.6.1.4.1.3512.1.6.1.1.1.2

objectCount
1.3.6.1.4.1.3512.1.6.1.1.1.3

objectAvgSize
1.3.6.1.4.1.3512.1.6.1.1.1.4

indexController
1.3.6.1.4.1.3512.1.6.2

indexControllerTable
1.3.6.1.4.1.3512.1.6.2.1

indexTableEntry
1.3.6.1.4.1.3512.1.6.2.1.1

tableIndex
1.3.6.1.4.1.3512.1.6.2.1.1.1

indexType
1.3.6.1.4.1.3512.1.6.2.1.1.2

unique
1.3.6.1.4.1.3512.1.6.2.1.1.3

indexName
1.3.6.1.4.1.3512.1.6.2.1.1.4

reverseKey
1.3.6.1.4.1.3512.1.6.2.1.1.5

totalInsert
1.3.6.1.4.1.3512.1.6.2.1.1.6

insertCollision
1.3.6.1.4.1.3512.1.6.2.1.1.7

collisionRatio
1.3.6.1.4.1.3512.1.6.2.1.1.8

bucketInstrumentation
1.3.6.1.4.1.3512.1.6.2.1.1.9

zeroBucket
1.3.6.1.4.1.3512.1.6.2.1.1.10

oneBucket
1.3.6.1.4.1.3512.1.6.2.1.1.11

twoBucket
1.3.6.1.4.1.3512.1.6.2.1.1.12

threeBucket
1.3.6.1.4.1.3512.1.6.2.1.1.13
SNMP Object IDs

fourBucket  
1.3.6.1.4.1.3512.1.6.2.1.1.14

fiveBucket  
1.3.6.1.4.1.3512.1.6.2.1.1.15

sixEightBucket  
1.3.6.1.4.1.3512.1.6.2.1.1.16

nineSixteenBucket  
1.3.6.1.4.1.3512.1.6.2.1.1.17

overSixteenBucket  
1.3.6.1.4.1.3512.1.6.2.1.1.18

longestSize  
1.3.6.1.4.1.3512.1.6.2.1.1.19

longestBucket  
1.3.6.1.4.1.3512.1.6.2.1.1.20

poidIndexController  
1.3.6.1.4.1.3512.1.6.2.3

  poidIndexControllerTable  
  1.3.6.1.4.1.3512.1.6.3.1

    poidIndexTableEntry  
    1.3.6.1.4.1.3512.1.6.3.1.1

      tableIndex  
      1.3.6.1.4.1.3512.1.6.3.1.1.1

      poidTypeName  
      1.3.6.1.4.1.3512.1.6.3.1.1.2

      poidNumber  
      1.3.6.1.4.1.3512.1.6.3.1.1.3

      residency  
      1.3.6.1.4.1.3512.1.6.3.1.1.4

      lockWaits  
      1.3.6.1.4.1.3512.1.6.3.1.1.5

      totalInsert  
      1.3.6.1.4.1.3512.1.6.3.1.1.6

      insertCollision  
      1.3.6.1.4.1.3512.1.6.3.1.1.7

      collisionRatio  
      1.3.6.1.4.1.3512.1.6.3.1.1.8

      zeroBucket  
      1.3.6.1.4.1.3512.1.6.3.1.1.9

      oneBucket  
      1.3.6.1.4.1.3512.1.6.3.1.1.10
twoBucket
1.3.6.1.4.1.3512.1.6.3.1.1.11

threeBucket
1.3.6.1.4.1.3512.1.6.3.1.1.12

fourBucket
1.3.6.1.4.1.3512.1.6.3.1.1.13

fiveBucket
1.3.6.1.4.1.3512.1.6.3.1.1.14

sixEightBucket
1.3.6.1.4.1.3512.1.6.3.1.1.15

nineSixteenBucket
1.3.6.1.4.1.3512.1.6.3.1.1.16

overSixteenBucket
1.3.6.1.4.1.3512.1.6.3.1.1.17

longestSize
1.3.6.1.4.1.3512.1.6.3.1.1.18

longestBucket
1.3.6.1.4.1.3512.1.6.3.1.1.19

AAA Manager Connection and Number Portability OIDs

dat
1.3.6.1.4.1.3512.1.7

  connectionPool
  1.3.6.1.4.1.3512.1.7.1

    connectionPoolGroup
    1.3.6.1.4.1.3512.1.7.1.1

      connectionPoolGroupEntry
      1.3.6.1.4.1.3512.1.7.1.1.1

       rebalanceConnection
       1.3.6.1.4.1.3512.1.7.1.1.1.1

    connectionPoolGroupIndex
    1.3.6.1.4.1.3512.1.7.1.1.2

  connectionMonitor
  1.3.6.1.4.1.3512.1.7.2

    connectionMonitorGroup
    1.3.6.1.4.1.3512.1.7.2.1

      connectionMonitorGroupEntry
      1.3.6.1.4.1.3512.1.7.2.1.1

       sendDPR
       1.3.6.1.4.1.3512.1.7.2.1.1.1
connectionMonitorGroupIndex
1.3.6.1.4.1.3512.1.7.2.1.1.2

numberPortability
1.3.6.1.4.1.3512.1.7.3

numberPortabilityGroup
1.3.6.1.4.1.3512.1.7.3.1

NumberPortabilityGroupEntry
1.3.6.1.4.1.3512.1.7.3.1.1

reload
1.3.6.1.4.1.3512.1.7.3.1.1.1

deltaLoad
1.3.6.1.4.1.3512.1.7.3.1.1.2

printData
1.3.6.1.4.1.3512.1.7.3.1.1.3

Memory Monitor OIDs

mml
1.3.6.1.4.1.3512.1.8

memoryManager
1.3.6.1.4.1.3512.1.8.1

memoryManagerGroup
1.3.6.1.4.1.3512.1.8.1.1

memoryManagerGroupEntry
1.3.6.1.4.1.3512.1.8.1.1.1

totalMemoryAllocated
1.3.6.1.4.1.3512.1.8.1.1.1.1

memoryManagerGroupIndex
1.3.6.1.4.1.3512.1.8.1.1.2

Statistics OIDs

plg
1.3.6.1.4.1.3512.1.9

statistic
1.3.6.1.4.1.3512.1.9.1

statisticGroup
1.3.6.1.4.1.3512.1.9.1.1

statisticGroupEntry
1.3.6.1.4.1.3512.1.9.1.1.1

totalEdrCount
1.3.6.1.4.1.3512.1.9.1.1.1.1

totalTxnEdrCount
1.3.6.1.4.1.3512.1.9.1.1.1.2
SNMP Object IDs

**BRM OMF Instrumented Objects**

- `totalProcTime`
  1.3.6.1.4.1.3512.1.9.1.1.1.3

- `totalTxnCount`
  1.3.6.1.4.1.3512.1.9.1.1.1.4

- `statisticGroupIndex`
  1.3.6.1.4.1.3512.1.9.1.1.1.5

- `statisticTable`
  1.3.6.1.4.1.3512.1.9.1.2

- `statisticEntry`
  1.3.6.1.4.1.3512.1.9.1.2.1

- `statisticTableIndex`
  1.3.6.1.4.1.3512.1.9.1.2.1.1

- `throughput`
  1.3.6.1.4.1.3512.1.9.1.2.1.2

- `throughputTimestamp`
  1.3.6.1.4.1.3512.1.9.1.2.1.3

**fct Probe OIDs**

- `fct`
  1.3.6.1.4.1.3512.1.10

- `opcodePlugIn`
  1.3.6.1.4.1.3512.1.10.1

- `opcodePlugInGroup`
  1.3.6.1.4.1.3512.1.10.1.1

- `opcodePlugInGroupEntry`
  1.3.6.1.4.1.3512.1.10.1.1.1

- `totalOpcodeCalls`
  1.3.6.1.4.1.3512.1.10.1.1.1.1

- `totalLatency`
  1.3.6.1.4.1.3512.1.10.1.1.1.2

- `opcodePlugInGroupIndex`
  1.3.6.1.4.1.3512.1.10.1.1.1.3

- `latencyTable`
  1.3.6.1.4.1.3512.1.10.1.2

- `latencyEntry`
  1.3.6.1.4.1.3512.1.10.1.2.1

- `latencyTableIndex`
  1.3.6.1.4.1.3512.1.10.1.2.1.1

- `latency`
  1.3.6.1.4.1.3512.1.10.1.2.1.2
SNMP Object IDs

Input Controller probes

```plaintext
latencyTimestamp
1.3.6.1.4.1.3512.1.10.1.2.1.3

ppl
1.3.6.1.4.1.3512.1.11

inputController
1.3.6.1.4.1.3512.1.11.1

inputControllerGroup
1.3.6.1.4.1.3512.1.11.1.1

inputControllerGroupEntry
1.3.6.1.4.1.3512.1.11.1.1.1

maxProcessingTime
1.3.6.1.4.1.3512.1.11.1.1.1.1

minProcessingTime
1.3.6.1.4.1.3512.1.11.1.1.1.2

inputControllerGroupIndex
1.3.6.1.4.1.3512.1.11.1.1.1.3

processingTimeTable
1.3.6.1.4.1.3512.1.11.1.2

processingTimeEntry
1.3.6.1.4.1.3512.1.11.1.2.1

processingTimeTableIndex
1.3.6.1.4.1.3512.1.11.1.2.1.1

processingTime
1.3.6.1.4.1.3512.1.11.1.2.1.2

timestamp
1.3.6.1.4.1.3512.1.11.1.2.1.3
```

Process Table

```plaintext
processTable
1.3.6.1.4.1.3512.1.101

processEntry
1.3.6.1.4.1.3512.1.101.1

processIndex
1.3.6.1.4.1.3512.1.101.1.1

processDescr
1.3.6.1.4.1.3512.1.101.1.2
```

Registry Table

```plaintext
registryTable
1.3.6.1.4.1.3512.1.102
```
registryEntry (1)
1.3.6.1.4.1.3512.1.102.1

registryIndex (1)
1.3.6.1.4.1.3512.1.102.1.1

registryName (2)
1.3.6.1.4.1.3512.1.102.1.2
This document describes how to manage Oracle Communications Billing and Revenue Management (BRM) Pipeline Manager framework components and pipelines.

For background information, see “About the Pipeline Manager System Architecture” in BRM Concepts.

About Configuring Pipeline Manager

To configure Pipeline Manager, you use the following files:

- **Registry files**, which you use to configure a Pipeline Manager instance at system startup. See "Using Registry Files to Configure Pipeline Manager".

- **Semaphore files**, which you use to configure and control pipelines during runtime. See "Using Semaphore Files to Control Pipeline Manager".

You can also use HP OpenView and the `pin_ctl` utility to start and stop Pipeline Manager. See "Starting and Stopping the BRM System".

Using Registry Files to Configure Pipeline Manager

A registry file is an ASCII text file that configures a Pipeline Manager instance at system startup. (There is one registry file for each Pipeline Manager instance.) You use a registry file to configure all of your Pipeline Manager system settings, such as the location of log files, your input stream format, data modules, pipelines, and the number of system threads.

---

**Note:** All directories and folders referenced in the registry file must exist prior to starting the Pipeline Manager.

---

After you have configured the registry file, you use the registry file name in the command for starting Pipeline Manager:

```
ifw -r RegistryFile
```

About the Registry File Structure

Registry files use a hierarchical structure, with each subsection nested within another. Each subsection provides the configuration for a module. These can be system modules, such as the Memory Monitor.
Each nested subsection is indented by several spaces and surrounded by curly braces { }. For example, the following shows how you specify the semaphore entries, `FilePath` and `FileName`:

```
ifw
{
  ...
  Semaphore
  {
    FilePath = /opt/ifw/semaphore
    FileName = semaphore.reg
  }
}
```

The registry hierarchy is shown in this documentation by the `dot (.)` convention. For example, this hierarchy:

```
ifw
{
  ...
  ProcessLog
  {
    Module
    {
      ...
    }
  }
}
```

is shown like this:

```
ifw.ProcessLog.Module
```

Where each period represents a level in the hierarchy.

The following shows the top-level subsections in the registry file. Each of these subsections controls a system-wide function as described in Table 7–1. The `ifw.Pipelines` section contains system-wide entries that apply to all pipelines, and subsections for each pipeline.

```
ifw
{
  Instrumentation
  DiagnosticDataHandler
  LogMessageTable
  Semaphore
  Registry
  ProcessLog
  MemoryMonitor
  EventHandler
  DataPool
  TransactionIdController
  SequencerPool
  Pipelines
  ...
}
```

where:
About the Registry File Syntax

Most registry file entries are key-value pairs separated by an equal sign (=):

```
Entry = Value
```

where:

- **Entry** specifies the entry name. Make sure you use the correct entry name spelling and capitalization; entry names are case-sensitive.
- **Value** is a value specific to the configuration entry.

For example, **Source = File** or **Split = True**.

A few registry file entries, such as **Reload**, do not take a value. In these cases, follow the identifier with curly braces. For example, **Reload {}**.

---

**Table 7–1   Top-Level Subsections in Registry File**

<table>
<thead>
<tr>
<th>Registry entry</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>ifw</td>
<td>Specifies the registry name for the Pipeline Manager instance. This is always the first entry in the registry. It is read by the Pipeline Manager Controller. See “About the Controller” in BRM Concepts.</td>
<td>Yes</td>
</tr>
<tr>
<td>ifw.Instrumentation</td>
<td>Section that configures Operations Management Framework (OMF) instrumentation data collection. See ”Enabling SNMP Instrumentation Data Collection”.</td>
<td>Yes</td>
</tr>
<tr>
<td>ifw.ParallelLoadManager</td>
<td>Section that configures multithreaded loading of your pipelines, data modules, and function modules. See ”Reducing Startup Times with Parallel Loading”.</td>
<td>No</td>
</tr>
<tr>
<td>ifw.DiagnosticDataHandler</td>
<td>Section that configures diagnostic data collection. See ”Using the Diagnostic Data Handler to Get OMF Diagnostic Data”.</td>
<td>Yes</td>
</tr>
<tr>
<td>ifw.LogMessageTable</td>
<td>Section that configures global log file setting. See ”About Pipeline Manager Log Files”.</td>
<td>Yes</td>
</tr>
<tr>
<td>ifw.Semaphore</td>
<td>Section that defines the name and location of your semaphore files. See ”Using Semaphore Files to Control Pipeline Manager”.</td>
<td>Yes</td>
</tr>
<tr>
<td>ifw.Registry</td>
<td>Section that defines the names and locations of the files that contain updated registry information. See ”Controller” in BRM Configuring Pipeline Rating and Discounting.</td>
<td>Yes</td>
</tr>
<tr>
<td>ifw.ProcessLog</td>
<td>Section that configures your process log. For information about the log entries, see ”About Pipeline Manager Log Files”.</td>
<td>Yes</td>
</tr>
<tr>
<td>ifw.MemoryMonitor</td>
<td>Section that configures memory monitoring. See ”Monitoring Pipeline Manager Memory Usage”.</td>
<td>No</td>
</tr>
<tr>
<td>ifw.DataPool</td>
<td>Section that configures your data modules. See ”Configuring the Data Pool”.</td>
<td>Yes</td>
</tr>
<tr>
<td>ifw.EventHandler</td>
<td>Section that configures the Event Handler. See ”About the Event Handler” in BRM Concepts.</td>
<td>No</td>
</tr>
<tr>
<td>ifw.TransactionIdController</td>
<td>Section that configures your Transaction ID Controller. See ”About the Transaction ID Controller” in BRM Concepts and ”About Pipeline Manager Transactions”.</td>
<td>Yes</td>
</tr>
<tr>
<td>ifw.SequencerPool</td>
<td>Section that configures all Sequencers used by a single Pipeline Manager instance. See ”Configuring Sequence Checking”.</td>
<td>No</td>
</tr>
<tr>
<td>ifw.Pipelines</td>
<td>Section that configures your individual pipelines. See ”About Configuring Pipelines”.</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Registry entries are either mandatory or optional. You must specify all mandatory entries. You can delete optional entries or comment them out by using a cross-hatch (#); Pipeline Manager uses default values for all unspecified optional entries.

Values can be either hard-coded and must be exact, or you can define your own.

- Hard-coded values must be entered exactly as documented. For example, when you enter a module name, it must have the correct spelling and capitalization. For FCT_Account, you cannot use FCT_account or FCT_ACCOUNT.

- Values that you define are often used elsewhere in the file, at which point they must be entered exactly as you defined them. For example, you might define the section for the DAT_AccountBatch module by using the entry `CustomerData`:

  ```
  #-----------------------------------------------------------
  # Infranet Customer Data
  #-----------------------------------------------------------
  CustomerData
  {
    ModuleName = DAT_AccountBatch
  }
  ```

  When you refer to that module elsewhere in the registry file, you point to `CustomerData`:

  ```
  DataModule = ifw.DataPool.CustomerData
  ```

**About the Sample Registry Files**

Pipeline Manager includes these sample registry files listed in Table 7–2 to help you get started:

<table>
<thead>
<tr>
<th>File name</th>
<th>Directory location</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>simple.reg</td>
<td>Pipeline_Home/samples/simple</td>
<td>Simple registry file that you can use to verify that Pipeline Manager installed properly. It does not require a database. It tests the pipeline input, output, and runs one functional module (FCT_PrefixDesc).</td>
</tr>
<tr>
<td>wireless.reg</td>
<td>Pipeline_Home/conf</td>
<td>Registry file that configures most function modules. This sample is a good place to start for creating your customized registry file.</td>
</tr>
<tr>
<td>wirelessRealtime.reg</td>
<td>Pipeline_Home/conf</td>
<td>Registry file that configures most function modules for real-time features. This sample is a good place to start for creating your customized registry file.</td>
</tr>
</tbody>
</table>

**About Configuring Pipelines**

Pipelines perform the Pipeline Manager functions, such as rating and zoning. See "About Pipelines" in BRM Concepts.

You configure pipelines in the `ifw.Pipelines` registry section. For example, a Pipeline Manager configuration with multiple pipelines looks like this:

```
ifw
{
  ...
  Pipelines
  {
    PipelineName
```
You can use any name you want to identify pipelines. You use that name in many places to point to the pipeline, so it should identify the function of the pipeline.

For each pipeline, you configure a pipeline controller. This section configures pipeline-specific configurations, such as threads, log files, the EDR Factory, and the `ifw.Pipelines.DataDescription` section. See the following topics:

- "About the Pipeline Controller" (see BRM Concepts)
- Using Events to Start External Programs
- About Pipeline Manager Transactions
- About Pipeline Manager Log Files

In addition, for each pipeline controller, you configure:

- An input section. See "Configuring EDR Input Processing" in BRM Configuring Pipeline Rating and Discounting.
- An `ifw.Pipelines.Functions` section. This section configures the function modules in the pipeline. The modules are run in the order that they are configured in this section. See "About Configuring Function Modules".
- An output section. See "Configuring EDR Output Processing" in BRM Configuring Pipeline Rating and Discounting.

The registry subsections in a pipeline are listed in Table 7–3:

<table>
<thead>
<tr>
<th>Registry Entry</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>ifw.Pipelines</td>
<td>Section that configures your individual pipelines.</td>
<td>Yes</td>
</tr>
<tr>
<td>ifw.Pipelines.PipelineName</td>
<td>Section that configures a single pipeline.</td>
<td>Yes</td>
</tr>
<tr>
<td>ifw.Pipelines.PipelineName.Input</td>
<td>Section that configures a pipeline’s input module. For information about the input module entries, see &quot;Pipeline Manager Input and Output Modules&quot; in BRM Configuring Pipeline Rating and Discounting.</td>
<td>Yes</td>
</tr>
<tr>
<td>ifw.Pipelines.PipelineName.Functions</td>
<td>Section that configures a pipeline’s function modules. For information about the function module entries, see &quot;About Configuring Function Modules&quot;.</td>
<td>Yes</td>
</tr>
<tr>
<td>ifw.Pipelines.PipelineName.Output</td>
<td>Section that configures a pipeline’s output module. For information about the output module entries, see &quot;Pipeline Manager Input and Output Modules&quot; in BRM Configuring Pipeline Rating and Discounting.</td>
<td>Yes</td>
</tr>
</tbody>
</table>

About Configuring Function Modules

You configure function modules in the `ifw.Pipelines.Functions` section.
The `ifw.Pipelines.Functions` section uses this hierarchy:

```plaintext
ifw
{
    ...
    Pipelines
    {
        PipelineName
        {
            Input
            ...
            Functions
            {
                Function_pool_name
                {
                    FunctionPool
                    {
                        Module_identifier
                        {
                            ModuleName = Module_executable
                            Module
                            {
                                Entry = value
                            }
                        }
                    }
                }
            }
        }
    }
    ...
}
```

The entries listed in Table 7–4 are a combination of required text and text that you define.

**Table 7–4  Pipeline Registry Functions Section Entries**

<table>
<thead>
<tr>
<th>Entry</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functions</td>
<td>Section name. You must use <code>Functions</code>.</td>
</tr>
<tr>
<td>Function_pool_name</td>
<td>The name of the function pool. You define this name. See &quot;Optimizing a Pipeline by Using Function Pools&quot;.</td>
</tr>
<tr>
<td>FunctionPool</td>
<td>Section name. You must use <code>FunctionPool</code>.</td>
</tr>
<tr>
<td>Module_identifier</td>
<td>The descriptive module identifier. For example, the module identifier for FCT_Account in the sample registry is CustomerSearch.</td>
</tr>
<tr>
<td></td>
<td>You define these names. They are often referenced by other modules; for example, to connect to the DAT_AccountBatch module, the FCT_Account module points to CustomerData.</td>
</tr>
<tr>
<td>ModuleName = Module_executable</td>
<td><code>ModuleName</code> is the entry. You must use <code>ModuleName</code>. <code>Module_executable</code> is the name of the module; for example, FCT_Account. This name is case-sensitive and must be spelled correctly; for example, you must use FCT_Account, not FCT_account or FCT_ACCOUNT. You can find the exact spelling and capitalization by looking at the executable name in the Pipeline_Home/lib directory.</td>
</tr>
<tr>
<td>Module</td>
<td>Section name. You must use <code>Module</code>.</td>
</tr>
<tr>
<td>Entry = value</td>
<td>These are the registry entries, for example: <code>Active = True</code>.</td>
</tr>
</tbody>
</table>

This example shows a sample hierarchy. This sample does the following:

- Creates a function pool named `PreProcessing`.
• Runs the FCT_IRules module, using the identifier *PipelineSplit*.

```c
    Functions
    {
      PreProcessing
      {
        FunctionPool
        {
          PipelineSplit
          {
            ModuleName = FCT_IRules
            Module
            {
              Active = True
            }
          }
        }
      }
    }
```

### About iScripts and iRules

iScripts and iRules perform processing tasks similar to function modules. They are run by the FCT_iScript and FCT_iRules modules. In addition to the iScripts and iRules provided by BRM, you can create your own iScripts and iRules.

See "Creating iScripts and iRules" in *BRM Developer’s Guide*.

### About Configuring iScripts

To run iScripts, you use the FCT_iScript module. See "FCT_iScript" in *BRM Configuring Pipeline Rating and Discounting*.

The registry section for the FCT_iScript module includes the script to run, for example:

```c
  ApplyTaxIScript
  {
    ModuleName = FCT_iScript
    Module
    {
      Active = True
      Source = File
      Scripts
      {
        ApplyTaxIScript
        {
          FileName = ./iScriptLib/iScriptLib_Roaming/ISC_ApplyTax.isc
        }
      }
    }
  }
```

You can provide registry parameters to use in the iScript. This example provides the iScript with a G/L code:

```c
  Scripts
  {
    ConsolidatedCPIScript
    {
      FileName = ./iScriptLib/iScriptLib_Roaming/ISC_ConsolidatedCP.isc
      GL_CODE = 1514
    }
  }
```
About Configuring iRules

To run iRules, you use the FCT_IRules modules. See "FCT_IRules" in BRM Configuring Pipeline Rating and Discounting.

To configure the FCT_IRules module, provide a connection to the Pipeline Manager database. The FCT_IRules module runs the rules that apply to the conditions in the pipeline. If a condition in a rule item matches the current EDR container, the evaluation stops and the script associated with the rule item is executed for the current EDR container.

This example shows a typical FCT_IRules registry section:

```plaintext
PipelineSplit
{
  ModuleName = FCT_IRules
  Module
  {
    Active = TRUE
    Source = Database
    DataConnection = integrate.DataPool.DataConnection
    Rules
    {
    }
  }
}
```

You can use the Rules entry to specify a specific script to run:

```plaintext
Rules
{
  TAP3.VAL
}
```

Configuring Multiple Instances of a Pipeline

To simplify the configuration of multiple pipelines, use the ifw.Pipelines.Instances subsection. Pipeline Manager reads the required number of instances for a given pipeline and instantiates each of them accordingly.

---

**Note:** The ifw.Pipelines.Instances subsection creates multiple instances of pipelines. To create multiple instances of sequencers, output streams, or system brands for multiple roaming partners, use the Instances module. See "About Configuring Multiple Instances of Sequencers, Output Streams, or System Brands" for more information.

---

For example, this subsection configures ten instances or the authorization pipeline:

```plaintext
ifw
{
  ...
  Pipelines
  {
    Instances
    {
      AuthPipeline
      {
        NumberOfInstances = 10
        InstanceSpecificRegistries
```
About Configuring Multiple Instances of Sequencers, Output Streams, or System Brands

To specify instance-specific registry entries, you add the entries in the ifw.Pipelines.Instances.Pipeline_Name.InstanceSpecificRegistries section.

The pipeline generates the instance-specific log file names by adding the instance ID to the base pipeline file names.

For example, if the base pipeline file name for the TransactionManager log file is binaryLogFile_RT_GPRS.dat, then the instance-specific files generated are binaryLogFile_RT_GPRS.dat0, binaryLogFile_RT_GPRS.dat1, and binaryLogFile_RT_GPRS.dat2.

Note: If instance-specific entries are not specified, the pipeline uses the base pipeline configurations.

About Configuring Multiple Instances of Sequencers, Output Streams, or System Brands

To manage multiple roaming partners, you can use the Instances module to configure multiple instances of sequencers, output streams, or system brands. You configure the Instances module by adding the ifw.Instances registry section in the roaming registry file (Pipeline_Home/conf/roaming.reg).

Note: To create multiple instances of pipelines, use the ifw.Pipelines.Instances subsection. See "Configuring Multiple Instances of a Pipeline" for more information.

The Instances module configures multiple instances of sequencers, output streams, or system brands using template sections or entries in the roaming registry file. Instead of creating multiple sections of entries, you use the single section or entry templates in the roaming registry file. When the pipeline runs, data for each roaming partner is inserted into the templates, effectively instantiating multiple registry sections or entries. For example, if there are two roaming partners, OPRT1 and OPRT2, the template is instantiated into two sections of entries in the pipeline.

To identify which roaming partners to use with the template, the Instances module reads the roaming configuration data file generated by the RoamingConfigGen64 utility. This file includes data for each of the roaming partners. For example, the data can include the sequencing information, output information, and so on.

You use the SequencerPool or OUT_GenericStream template section or the SystemBrands template entry in the roaming registry file to configure multiple sequencers, output streams, or system brands. These template sections or entries contain the variables that must be changed in each new instance of the SequencerPool or OUT_GenericStream section or the SystemBrands entry instantiated in the pipeline.
The following example shows the `SequencerPool` template section:

```plaintext
SequencerPool
{
  SEQ_GEN_TAPOUT_XXX
  {
    Source = Database
    Controller
    {
      SequencerType = Generation
      ReuseGap = True
      SequenceLength = 5
      DatabaseConnection = ifw.DataPool.Login
    }
  }
}
```

where `XXX` is the visiting network operator code that must be changed in each new instance of the `SequencerPool` section; for example, OPRT1, OPRT2, and so on.

Use the Instances module in conjunction with the `RoamingConfigGen64` utility. The `RoamingConfigGen64` utility collects the roaming partner information from the Pipeline Manager database and creates the roaming configuration data file. The Instances module uses the values in the roaming configuration data file to replace the variables in each instance of the `SequencerPool` or `OUT_GenericStream` section or the `SystemBrands` entry instantiated in the pipeline.

When you run the `RoamingConfigGen64` utility, you specify a home network operator code. The utility searches the Pipeline Manager database to find the VPLMNs associated with that home network operator. For example, if the home network operator has two VPLMNs, a record for each of them is created in the roaming configuration data file.

The following example shows the roaming configuration data file generated by the `RoamingConfigGen64` utility:

```
# Column Headers
# Column Headers
# Column Headers
# Column Headers
# Column Headers
# Column Headers
# Column Headers
VPLMN| TAPOUTSEQUENCER| NRTREDEOUTSEQUENCER| TAPOUTSTREAM| NRTREDEOUTSTREAM| TAPOUTPATH| NRTREDEOUTPATH| TAPOUTPREFIX| NRTREDEOUTPREFIX| TMPPATH| TMPDATAPREFIX
OPRT1| SEQ_GEN_TAPOUT_OPRT1| SEQ_GEN_NRTREDEOUT_OPRT1| TAPOutput_OPRT1| NRTREDEOutput_OPRT1| ./data/outcollect/tapout/oprt1| ./data/outcollect/nrtrdeout/oprt1| CDEUR01OPRT1| NREUR01OPRT1| temptest_oprt1| temp.oprt1.tmp.
OPRT2| SEQ_GEN_TAPOUT_OPRT2| SEQ_GEN_NRTREDEOUT_OPRT2| TAPOutput_OPRT2| NRTREDEOutput_OPRT2| ./data/outcollect/tapout/oprt2| ./data/outcollect/nrtrdeout/oprt2| CDEUR01OPRT2| NREUR01OPRT2| temptest_oprt2| temp.oprt2.tmp.
```

The following example shows the entries in the `ifw.Instances` registry section to configure multiple instances of sequencers:

```plaintext
{
  ifw
  {
    Instances
    {
      SEQ_GEN_TAPOUT
      {
        BlockName = SequencerPool.SEQ_GEN_TAPOUT_XXX
        DataFile = ./RoamingPartnerConf.dat
      }
    }
  }
}
```
InstanceSpecificEntries
{
    ModifyBlockName
    {
        Instance = [BlockName]
        UseColumn = TAPOUT_SEQUENCER
    }
}

The following example shows the two instances of sequencers instantiated in the pipeline, based on the entries in the ifw.Instances registry section, using the TAPOUTSEQUENCER values in the data file:

SequencerPool
{
    SEQ_GEN_TAPOUT_OPRT1
    {
        Source = Database
        Controller
        {
            SequencerType = Generation
            ReuseGap = True
            SequenceLength = 5
            DatabaseConnection = ifw.DataPool.Login
        }
    }
    SEQ_GEN_TAPOUT_OPRT2
    {
        Source = Database
        Controller
        {
            SequencerType = Generation
            ReuseGap = True
            SequenceLength = 5
            DatabaseConnection = ifw.DataPool.Login
        }
    }
}

See "Configuring Multiple Instances of Sequencers, Output Streams, or System Brands" for instructions.

Configuring Multiple Instances of Sequencers, Output Streams, or System Brands

To configure multiple instances of sequencers, output streams, or system brands:

1. Create the roaming configuration data file by running the following command:

   RoamingConfigGen64 -l database_access_library -s server_name [-d database_name] 
   -c operator_code [-o output_path] [-b base_path]

   where:
   - database_access_library is the database access library; for example, 
     liboci10g6312d.a for Oracle on AIX.
   - server_name specifies the name of the host machine running the Pipeline 
     Manager database.
   - database_name specifies the database name of the Pipeline Manager database. 
     The default is an empty string (" ").
About Configuring Multiple Instances of Sequencers, Output Streams, or System Brands

- **operator_code** specifies the home network operator code. The default is PORTL.
- **output_path** specifies the output path for the data file generated by the RoamingConfigGen64 utility. By default, the data file is saved in the Pipeline_Home/conf/ directory.
- **base_path** specifies the base path to the directory for Transferred Account Procedure (TAP) and Near Real Time Roaming Data Exchange (NRTRDE) output files. The default path is Pipeline_Home/data/outcollect/

For example:

```bash
RoamingConfigGen64 -l liboci10g6312d.so -s $ORACLE_SID -d ' ' -c EUR01 -o Pipeline_Home/conf/ -b Pipeline_Home/data/outcollect/
```

For more information about the RoamingConfigGen64 Perl script, see "RoamingConfigGen64" in BRM Configuring Pipeline Rating and Discounting.

2. Open the roaming registry file (Pipeline_Home/conf/roaming.reg) file in a text editor.

3. Ensure that the SequencerPool or OUT_GenericStream template section or the SystemBrands template entry exists in the roaming registry file.

   If the template for the roaming registry section or entry you want to instantiate does not exist, create a template for that registry section or entry in the file.

   The following example shows the SequencerPool template section:

   ```bash
   SequencerPool
   { 
   SEQ_GEN_TAPOUT_XXX
   { 
   Source = Database
   Controller 
   { 
   SequencerType = Generation
   ReuseGap = True
   SequenceLength = 5
   DatabaseConnection = ifw.DataPool.Login
   }
   }
   }
   ```

4. Add the instance-specific entries in the ifw.Instances.InstantiationName.InstanceSpecificEntries subsection. If the ifw.Instances registry section does not exist, you must add the section in the file.

   The ifw.Instances registry section uses the following hierarchy:

   ```bash
   Instances
   { 
   InstantiationName 
   { 
   BlockName  =?TemplatePath
   DataFile  =DataFilePath
   InstanceSpecificEntries 
   { 
   InstanceChangeName 
   { 
   Instance = InstanceValue
   UseColumn = ColumnName
   Mode = ModeValue
   } 
   } 
   } 
   }
   ```
Configuring the Data Pool

To configure data modules, you configure the ifw.DataPool registry subsection. This subsection uses the following hierarchy:

    DataPool
    {
        Module_identifier
        {
            ModuleName = Module_executable
            Module
            {
                Entry = value
            }
        }
    }

where:

- **InstantiationName** is the descriptive name of the instantiation; for example, SEQ_GEN_TAPOUT.

- **TemplatePath** is the template section or entry in the roaming registry file that is used to instantiate multiple registry sections or entries. For example, SequencerPool.SEQ_GEN_TAPOUT_XXX

- **DataFilePath** is the path to the data file generated by the RoamingConfigGen64 utility; for example, Pipeline_Home/conf/RoamingPartnerConf.dat.

- **InstanceChangeName** is the descriptive name of the change required in each instance; for example, ModifyBlockName.

- **InstanceValue** specifies whether to change the section name, entry name, or the value of the entry in each new instance created.

  The valid values are:
  
  - `[BlockName]` specifies that the section name or entry name must be changed in each new instance.
  
  - `[BlockValue]` specifies that the value of the entry must be changed in each new instance.
  
  - `RegistryEntry` specifies the entry in the template section for which the value must be changed in each new instance; for example, Module.Recipient.

- **ColumnName** is the column in the data file generated by the RoamingConfigGen64 utility that is used to change the section name, entry name, or the value of the entry in each instance according to the change mode. For example, TAPOUT_SEQUENCER.

- **ModeValue** is the mode of changing (such as REPLACE) the section name, entry name, or the value of the entry in each instance using the column values in the data file generated by the RoamingConfigGen64 utility.

For more information on the Instances module, see "Instances" in BRM Configuring Pipeline Rating and Discounting.

5. Save and close the file.

6. Stop and restart Pipeline Manager.

### Configuring the Data Pool

To configure data modules, you configure the ifw.DataPool registry subsection. This subsection uses the following hierarchy:

    DataPool
    {
        Module_identifier
        {
            ModuleName = Module_executable
            Module
            {
                Entry = value
            }
        }
    }

where:

- **InstantiationName** is the descriptive name of the instantiation; for example, SEQ_GEN_TAPOUT.

- **TemplatePath** is the template section or entry in the roaming registry file that is used to instantiate multiple registry sections or entries. For example, SequencerPool.SEQ_GEN_TAPOUT_XXX

- **DataFilePath** is the path to the data file generated by the RoamingConfigGen64 utility; for example, Pipeline_Home/conf/RoamingPartnerConf.dat.

- **InstanceChangeName** is the descriptive name of the change required in each instance; for example, ModifyBlockName.

- **InstanceValue** specifies whether to change the section name, entry name, or the value of the entry in each new instance created.

  The valid values are:
  
  - `[BlockName]` specifies that the section name or entry name must be changed in each new instance.
  
  - `[BlockValue]` specifies that the value of the entry must be changed in each new instance.
  
  - `RegistryEntry` specifies the entry in the template section for which the value must be changed in each new instance; for example, Module.Recipient.

- **ColumnName** is the column in the data file generated by the RoamingConfigGen64 utility that is used to change the section name, entry name, or the value of the entry in each instance according to the change mode. For example, TAPOUT_SEQUENCER.

- **ModeValue** is the mode of changing (such as REPLACE) the section name, entry name, or the value of the entry in each instance using the column values in the data file generated by the RoamingConfigGen64 utility.

For more information on the Instances module, see "Instances" in BRM Configuring Pipeline Rating and Discounting.

5. Save and close the file.

6. Stop and restart Pipeline Manager.
The entries listed in Table 7–5 are a combination of required text and text that you define.

**Table 7–5  Pipeline Registry DataPool Section Entries**

<table>
<thead>
<tr>
<th>Entry</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DataPool</td>
<td>Section name. You must use DataPool.</td>
</tr>
<tr>
<td>Module_identifier</td>
<td>The descriptive module identifier. For example, in the sample registry, the module identifier for DAT_AccountBatch is CustomerData.</td>
</tr>
<tr>
<td>ModuleName = Module_executable</td>
<td>ModuleName is the entry. You must use ModuleName. Module_executable is the name of the module; for example, DAT_AccountBatch. This name is case-sensitive and must be spelled correctly; for example, you must use DAT_AccountBatch, not DAT_Accountbatch or DAT_Account_Batch.</td>
</tr>
<tr>
<td>Module</td>
<td>Section name. You must use Module.</td>
</tr>
</tbody>
</table>

This example shows a sample hierarchy:

```
DataPool
{
    CustomerData
    {
        ModuleName = DAT_AccountBatch
        Module
        {
            IntegrateConnection = ifw.DataPool.Login
        }
    }
}
```

**Connecting a Module to a Database**

You connect modules to the Pipeline Manager database and the BRM database through the Database Connect module. To do so:

1. Configure the Database Connect module in the `ifw.DataPool` section of the registry file. For information, see "Database Connect (DBC)" in *BRM Configuring Pipeline Rating and Discounting*.

   You can configure three types of connections:
   - A connection to the Pipeline Manager database.
   - A connection to the BRM database.
   - A connection to the database login queue (used by the DAT_Listener module).

2. When configuring a module that needs a connection to the Pipeline Manager database, use one of the following registry entries:
   - `DataConnection`
   - `IntegrateConnection`
These entries do the same thing; they point to the `ifw.DataPool.Login` section. For example:

```plaintext
DataConnection = ifw.DataPool.Login
IntegrateConnection = ifw.DataPool.Login
```

See the documentation for each module to determine which entry to use.

---

**Note:** Some modules can get data either from the database or from a file. If you configure the module to get data from a file, the module does not connect to the database.

---

3. When configuring a module that needs a connection to the BRM database, configure one of the following registry entries:

- **DataConnection**
- **InfranetConnection**

These entries do the same thing; they point to the `ifw.DataPool.LoginInfranet` section. For example:

```plaintext
DataConnection = ifw.DataPool.LoginInfranet
InfranetConnection = ifw.DataPool.LoginInfranet
```

---

**Forcing a Database Reconnection**

You can force the Database Connect module to reconnect to the Pipeline Manager database by using the following semaphore entry:

```plaintext
ifw.DataPool>Login.Module.Reconnect {}
```

This semaphore closes all open database connections and reconnects the Database Connect module to the Pipeline Manager database.

For information on how to create semaphore files, see "Updating Configuration Settings During Runtime by Using Semaphore Files".

---

**Reloading Data into a Pipeline Manager Module**

When you update data in the Pipeline Manager database, it is not automatically loaded into the modules. For example, if you change pricing data, EDRs continue to be rated by using the old pricing data until the new data is loaded into the data modules.

You use the `Reload` semaphore entry to reload data from the database into a module.

If the reload operation does not succeed, the module stops processing EDRs until data is loaded correctly. In some cases, you can configure how a module behaves if reloading fails:

- To configure a module to immediately resume processing using the previous data, set its `ReuseOnFailure` startup registry entry to `True`. Not all modules have this registry entry. Check the module’s reference documentation to determine whether its registry includes `ReuseOnFailure`.

- To ensure that a module does not resume processing EDRs until the latest data is loaded, do not include `ReuseOnFailure` in the registry. This is the only option for modules that do not include this registry entry.
Using Business Parameter Settings from the BRM Database

You enable or disable optional BRM features and functionality by configuring business parameter settings, which are stored in `/config/business_params` objects in the BRM database. Pipeline Manager can determine whether these features and functionality are enabled by using the DAT_PortalConfig module, which retrieves and stores business parameter settings from the BRM database at pipeline initialization. Any other data modules that need a business parameter setting retrieve it directly from the DAT_PortalConfig module’s internal memory.

Table 7–6 lists the data modules that use business parameter settings, the features that depend on the setting, and the `/config/business_params` parameter class and entry that each feature uses:

<table>
<thead>
<tr>
<th>Pipeline Manager module</th>
<th>Feature</th>
<th>Parameter class</th>
<th><code>/config/business_params</code> entry</th>
</tr>
</thead>
<tbody>
<tr>
<td>DAT_AccountBatch</td>
<td>Balance monitoring. See “About Balance Monitoring” in BRM Managing Accounts Receivable.</td>
<td>multi_bal</td>
<td>BalanceMonitoring</td>
</tr>
<tr>
<td>DAT_BalanceBatch</td>
<td>Validity end time for first-usage resources. See &quot;About Restricting the End Time of Granted Resources that Start on First Usage&quot; in BRM Setting Up Pricing and Rating.</td>
<td>multi_bal</td>
<td>RestrictResourceValidityToOffer SortValidityBy CreditThresholdChecking</td>
</tr>
<tr>
<td>DAT_Discount</td>
<td>Discount validity and exclusion rules. See “About Discount Exclusion Rules” in BRM Configuring Pipeline Rating and Discounting.</td>
<td>billing</td>
<td>ValidateDiscountDependency</td>
</tr>
</tbody>
</table>

To set up Pipeline Manager to use business parameter settings from the BRM database, perform these tasks:

1. Configure the DAT_PortalConfig module in your registry file. This module must be listed in the registry file before any other data modules that are connected to it. See "DAT_PortalConfig" in BRM Configuring Pipeline Rating and Discounting.
2. Configure data modules to retrieve business parameter settings from DAT_PortalConfig. See "Connecting Pipeline Manager Modules to DAT_PortalConfig".

After Pipeline Manager starts, you can:

- Verify that the entries loaded properly by printing the parameters that DAT_PortalConfig has stored in memory. See "Printing Business Parameter Settings Stored in DAT_PortalConfig Memory".
- Refresh business parameter settings stored in the DAT_PortalConfig module’s internal memory. See "Refreshing Business Parameter Settings Stored in DAT_PortalConfig Memory".

Connecting Pipeline Manager Modules to DAT_PortalConfig

You must connect all data modules in your system that need business parameter settings to DAT_PortalConfig. You connect a module to DAT_PortalConfig by using the module’s `PortalConfigDataModule` registry entry. For example:
PortalConfigDataModule=ifw.DataPool.PortalConfigDataModule

Note: You can use any name you want to identify the registry section that configures DAT_PortalConfig, but you must use that name exactly when configuring modules to point to that registry section.

For example, the following entry, shown in bold, connects the DAT_Discount module to DAT_PortalConfig:

```
#-----------------------------------------------------------
# Discount Model Data Module
#-----------------------------------------------------------
DiscountModelDataModule
{
    ModuleName = DAT_Discount
    Module
    {
        InfranetConnection = ifw.DataPool.LoginInfranet
        IntegrateConnection = ifw.DataPool.Login
        PortalConfigDataModule = ifw.DataPool.PortalConfigDataModule
        AccountDataModule = ifw.DataPool.CustomerData
    }
}
```

**Printing Business Parameter Settings Stored in DAT_PortalConfig Memory**

To print to a file the business parameter settings stored in the DAT_PortalConfig module’s memory, use the CBPPrintData semaphore (see "DAT_PortalConfig" in BRM Configuring Pipeline Rating and Discounting). For example:

```
```

where:

- **Path** specifies where to create the output file. By default, the file is created in the current directory.
- **Filename** specifies the name for the output file. The default file name is `DefaultCBPDataFile_timestamp.lst`. The module appends a timestamp to the end of the file name to prevent the module from overwriting existing files.

For example:

```
```

When you submit the print semaphore, DAT_PortalConfig generates an output file that uses the format shown below:

```
<BusParamConfiguration>
<BusParamConfigurationList>
<ParamClass name="group_name">
<Param>
    <Name>parameter_name</Name>
    <Type>data_type</Type>
    <Value>parameter_value</Value>
</Param>
</ParamClass>
</BusParamConfigurationList>
</BusParamConfiguration>
```
For example, the following shows a sample output file for the `billing` parameter class:

```xml
<BusParamConfiguration>
  <BusParamConfigurationList>
    <ParamClass name="billing">
      <Param>
        <Name>rerate_during_billing</Name>
        <Type>INT</Type>
        <Value>0</Value>
      </Param>
      <Param>
        <Name>validate_discount_dependency</Name>
        <Type>INT</Type>
        <Value>0</Value>
      </Param>
      <Param>
        <Name>sub_bal_validity</Name>
        <Type>INT</Type>
        <Value>0</Value>
      </Param>
    </ParamClass>
  </BusParamConfigurationList>
</BusParamConfiguration>
```

For information about semaphores, see "Using Semaphore Files to Control Pipeline Manager".

### Refreshing Business Parameter Settings Stored in DAT_PortalConfig Memory

You must refresh DAT_PortalConfig memory whenever you update the `BalanceMonitoring`, `RestrictResourceValidityToOffer`, or `ValidateDiscountDependency` business parameter settings in the BRM database.

You refresh the memory by using the CBPReload semaphore entry (see "DAT_PortalConfig" in *BRM Configuring Pipeline Rating and Discounting*). For example:

```plaintext
```

For information about semaphores, see "Using Semaphore Files to Control Pipeline Manager".

### Connecting a Pipeline Manager Module to Another Module

Most function modules connect to data modules to get configuration data. For example, the FCT_Account module requires a connection to the DAT_AccountBatch module. Also, some data modules connect to other data modules.

To connect one module to another, you configure a registry entry for the module that requires the connection. For example, to connect the FCT_Account module to the DAT_AccountBatch module, you enter this when you configure the FCT_Account module:

```plaintext
DataModule = ifw.DataPool.CustomerData
```

**CustomerData** identifies the DAT_AccountBatch module, which is configured in the registry like this:

```plaintext
# Infranet Customer Data
```

---

7-18  BRM System Administrator's Guide
CustomerData

```java
ModuleName = DAT_AccountBatch
```

**Note:** You can use any name you want to identify the registry section that configures a module, but you must use that name exactly when configuring modules to point to that registry section.

A function module can connect to more than one data module. For example, the FCT_ApplayBalance module includes two data module connection entries:

- DiscountDataModule = ifw.DataPool.DiscountModelDataModule

In addition, function modules, like data modules, can require a connection to the Pipeline Manager or BRM database, for example:

```java
DataConnection = ifw.DataPool>LoginInfranet
```

## Configuring Pipeline Buffers

Pipeline Manager uses buffers to control the flow of data moving from one thread to another. For example, you insert a buffer block into the LOG module to temporarily store log data received from your thread before it is written by the logging thread to a file.

To insert a buffer, you configure the pipeline’s or module’s **Buffer**, **InputBuffer**, or **OutputBuffer** registry section. In each section, you specify the buffer’s type and size. Pipeline Manager supports the following buffer types:

- Rogue Wave buffers. See "Using Rogue Wave Buffers".
- Block transfer buffers. See "Using Block Transfer Buffers on Solaris Systems".
- Array buffers. See "Using Array Buffers on HP-UX Itanium and Solaris Systems".

**Important:** When configuring buffers in multiple function pools, each buffer must have a unique name.

## Using Rogue Wave Buffers

By default, all buffers in Pipeline Manager are Rogue Wave buffers. These buffers are simple FIFO buffers of a configurable size. When a thread writes to or reads from a Rogue Wave buffer, it locks the entire buffer to ensure the integrity of the data. For example, if a Rogue Wave buffer has 15 containers, all 15 containers are locked when a thread accesses the buffer. Other threads must wait for the buffer to be unlocked before they can read or write data. For this reason, Rogue Wave buffers work best when only one thread will access the buffer.

**Note:** If multiple threads will access the buffer, use a block transfer or array buffer. See "Using Block Transfer Buffers on Solaris Systems" and "Using Array Buffers on HP-UX Itanium and Solaris Systems".

When a thread attempts to write to a full buffer or read from an empty buffer, the thread sleeps before attempting to access the buffer again.
To use a Rogue Wave buffer, you specify only the size of the buffer, by using the `Size` registry entry. This entry, listed in Table 7–7, goes in the `Buffer`, `InputBuffer`, or `OutputBuffer` registry section.

### Table 7–7 Rogue Wave Buffers Registry Entry

<table>
<thead>
<tr>
<th>Registry entry</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>Specifies the size of the internal data buffer.</td>
<td>Yes</td>
</tr>
</tbody>
</table>

The following shows sample registry entries for a Rogue Wave buffer:

```plaintext
Buffer
{
    Size = 100
}
```

This registry example creates a Rogue Wave buffer with 100 containers.

### Using Block Transfer Buffers on Solaris Systems

Block transfer buffers address performance and scalability issues that occur when two or more threads are accessing the same buffer. They are recommended for use on Solaris systems.

A block transfer buffer is a buffer that is separated into logical subsections (or blocks) with a configurable number of buffer containers. When a thread accesses a buffer container, it locks only those containers that are in the same block. This allows other threads to access other buffer containers in the remaining free blocks.

For example, a buffer has 15 containers separated into 3 logical blocks. A thread writing to a container in block B locks the block to prevent other threads from changing the container’s value during the write operation. Threads dedicated to blocks A and C can still read and write data because those blocks are unlocked as shown in Figure 7–1.

### Figure 7–1 Block Transfer Buffer Locking

![Block Transfer Buffer Locking Diagram]

When a thread attempts to write to a full buffer or read from an empty buffer, the thread sleeps before attempting to access the buffer again.

To use a block transfer buffer, you use the following buffer registry entries listed in Table 7–8. These entries go in the `Buffer`, `InputBuffer`, or `OutputBuffer` section.
The following shows sample registry entries for a block transfer buffer:

```plaintext
Buffer
{
  Size = 4000
  BlockTransfer = True
  BlockSize = 500
}
```

This example specifies a buffer size of 4,000 containers and a block size of 500 containers. Therefore, the buffer has eight (4,000/500) blocks.

### Using Array Buffers on HP-UX Itanium and Solaris Systems

Array buffers address performance and scalability issues that occur when two or more threads are accessing the same buffer. They are recommended for use on HP-UX Itanium and Solaris systems.

Array buffers are similar to Rogue Wave buffers, except threads never lock the buffer. To protect shared data, threads use a compare and swap (CAS) method that atomically compares a container’s old and current values before writing new data. This allows a thread to read data from a buffer container, modify it, and write it back only if no other thread modified it in the meantime.

When writing data to an array buffer, a thread performs the following:

1. Reads the buffer container. The thread takes three arguments: the container’s address, the container’s current value, and the new value.
2. Updates the local variable with the new value.
3. Determines whether any other thread modified the container in the interim by comparing the container’s current value with the value it initially read in step 1:
   - If the value has not changed, the thread writes the new value to the container.
   - If the value has changed, another thread modified the container during the write operation. The thread does not change the value and instead starts the process over at step 1.

When a thread attempts to write to a full buffer or read from an empty buffer, the thread spins for a specified number of times and then allows another thread to access the buffer before spinning again. You can specify the maximum number of times the thread yields before sleeping. The thread sleeps for a specified amount of time before starting the spin-and-yield process again.

### Table 7–8 Solaris Block Transfer Buffers Registry Entries

<table>
<thead>
<tr>
<th>Registry Entry</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>Specifies the size of the internal data buffer.</td>
<td>Yes</td>
</tr>
<tr>
<td>BlockTransfer</td>
<td>Specifies whether the buffer operates in block transfer mode. True: The buffer operates in block transfer mode. False: The buffer does not operate in block transfer mode. If set to False, the pipeline ignores the BlockSize registry entry. The default is False.</td>
<td>Yes</td>
</tr>
<tr>
<td>BlockSize</td>
<td>Specifies the size of each buffer block.</td>
<td>Yes, if BlockSize is set to True.</td>
</tr>
</tbody>
</table>

Table 7–8 Solaris Block Transfer Buffers Registry Entries
For example, if the maximum number of spins is 2, the maximum number of yields is 2, and the sleep time is 10 milliseconds, the thread performs the following while waiting for a buffer container to become available:

1. Spins 2 times.
2. Yields to another thread.
3. Spins 2 times.
4. Yields to another thread.
5. Sleeps for 10 milliseconds.

To use an array buffer, you use the following buffer registry entries listed in Table 7–9. These entries go in the Buffer, InputBuffer, or OutputBuffer section.

Table 7–9  Array Buffer Entries

<table>
<thead>
<tr>
<th>Registry entry</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>Specifies the size of the internal data buffer.</td>
<td>Yes</td>
</tr>
<tr>
<td>ArrayType</td>
<td>Specifies whether the buffer is an array buffer.</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>True: The buffer is an array buffer.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>False: The buffer is not an array buffer. If set to False, the pipeline</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ignores the SpinCount, YieldCount, and SleepTimeMilliSec registry entries.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The default is False.</td>
<td></td>
</tr>
<tr>
<td>SpinCount</td>
<td>Specifies the maximum number of times the thread spins while waiting for</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>a buffer container to become available.</td>
<td></td>
</tr>
<tr>
<td>YieldCount</td>
<td>Specifies the maximum number of times a thread yields to another thread</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>before the thread starts a sleep cycle.</td>
<td></td>
</tr>
<tr>
<td>SleepTimeMilliSec</td>
<td>Specifies how long the thread sleeps, in milliseconds, before trying to</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>access the buffer again.</td>
<td></td>
</tr>
</tbody>
</table>

The following shows sample registry entries for an array buffer:

```
Buffer
{
  Size = 100
  ArrayType = True
  SpinCount = 100
  YieldCount = 100
  SleepTimeMilliSec = 10
}
```

Using Semaphore Files to Control Pipeline Manager

You use semaphore files to configure and control Pipeline Manager during runtime. They enable you to perform business tasks regularly without having to stop and restart the pipeline. For example, you can use semaphore files to stop a module or to reload data from the database.

The Controller checks for new semaphore files to process at a regular interval. You configure where and how often the Controller checks for new semaphore files by using the Semaphore and ProcessLoopTimeout registry entries. See “Controller” in BRM Configuring Pipeline Rating and Discounting.
When the Controller finds a semaphore file, it:

1. Prevents new transactions from being created.
2. Finishes processing all open transactions in the framework.
3. Stops the pipeline framework.
4. Loads the semaphore file into memory.
5. Changes the specified configuration settings and/or executes the specified semaphores.
7. Renames or deletes the semaphore file from the directory.
   You configure the Controller to rename or delete semaphore files by using the `RetainFiles` semaphore entry.
8. Stops and restarts the pipeline framework.

For information on creating semaphore files, see "Updating Configuration Settings During Runtime by Using Semaphore Files".

### Updating Configuration Settings During Runtime by Using Semaphore Files

To change the Pipeline Manager configuration during runtime, you must:

1. Specify where and how often the Controller checks for semaphore files. See "Configuring Where and How Often the Controller Checks for Semaphore Files".

   **Note:** You perform this procedure only once, when you first configure your registry file.

2. Create your semaphore files. See "Procedure for Updating Configuration Settings".

Pipeline Manager includes a set of Perl scripts, and associated semaphore files, that you can use to for system administration tasks. See "Using Perl Scripts to Administer Pipeline Manager".

### Configuring Where and How Often the Controller Checks for Semaphore Files

You use the following registry entries in Table 7–10 to specify where and how often the Controller checks for semaphore files:

#### Table 7–10 | Controller Configuration Registry Entries

<table>
<thead>
<tr>
<th>Semaphore</th>
<th>Value</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ifw.ProcessLoopTimeout</code></td>
<td>Integer</td>
<td>Specifies the interval, in seconds, between polling for a new semaphore file.</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> This parameter controls the overall event loop, which includes looking for semaphore files.</td>
<td></td>
</tr>
</tbody>
</table>
Table 7–10  (Cont.) Controller Configuration Registry Entries

<table>
<thead>
<tr>
<th>Semaphore</th>
<th>Value</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Semaphore.FileName</td>
<td>String</td>
<td>Specifies the name of the semaphore file.</td>
<td>Yes</td>
</tr>
<tr>
<td>Semaphore.RetainFiles</td>
<td>True</td>
<td>Specifies whether semaphore files are deleted or saved after they are processed.</td>
<td>No</td>
</tr>
</tbody>
</table>
|               | False   | - **True** specifies to save semaphore files. The Controller renames the file by appending the current timestamp to the file name in the format YYYYMMDD_hhmmss and logs the semaphore file’s new name in the process.log file. For example, the semaphore.reg file is renamed semaphore.reg_20031022_120803.  
- **False** specifies to delete semaphore files immediately after they are processed. The default is **False**. |

Sample Registry Entries

```plaintext
ifw
{
    ...
    ProcessLoopTimeout = 30
    ...
    Semaphore
    {
        FilePath = /opt/ifw/semaphores
        FileName = semaphore.reg
        RetainFiles = True
    }
    ...
}
```

Procedure for Updating Configuration Settings

To update configuration settings during runtime:

1. Create a semaphore file using the file name specified in the registry file. (The examples in this document use `semaphore.reg`.)
2. Add new configuration or semaphore entries to the file. See "Semaphore File Syntax".

**Note:** The maximum number of entries you can add is **10000**.

3. Copy the semaphore file to the semaphore directory.
Semaphore File Syntax

Semaphore commands use one of these formats:

- Key-value pair format, such as `LoadZoneDescription = True`. These semaphore commands require a value.

  **Note:** The semaphore command fails if you do not supply a value.

- Semaphore entry `{ }` format, such as `Reload{}`.

  The commands in the semaphore file can be expressed in a nested hierarchy format or in a flattened syntax that uses periods to delimit nested sections. The syntax of a command reflects the hierarchical structure of the registry.

  **Important:** You must specify the full path for the command when using either the hierarchy or the flattened format.

The following examples show how to set the process log file name by using the hierarchy and flattened formats.

**Hierarchy Format**

```
ifw
{
    ProcessLog
    {
        Module
        {
            ITO
            {
                FileName = process
            }
        }
    }
}
```
Using Events to Start External Programs

Flattened Format

\texttt{ifw.ProcessLog.Module.ITO.FileName = process}

Though registry files can vary in structure, commands for each type of module follow a similar pattern. For function modules, the syntax follows this pattern (shown in flattened format):

\texttt{ifw.Pipelines.Pipeline_Name.Functions.Function_pool_name.}
\texttt{FunctionPool.Module_identifier.Module.Entry = Value}

For example:

\texttt{ifw.Pipelines.ALL_RATE.Functions.Processing.FunctionPool.}
\texttt{Aggregate.Module.Active = False}

For data modules, the syntax is:


For example:

\texttt{ZM_MOBILE = /data9/INTEGRATE/test/config/ZM_MOBILE-new.dat}

You can specify multiple commands in one semaphore file by placing each command on a separate line. For example:

\texttt{ifw.Pipelines.ALL_RATE.Active = True}
\texttt{ifw.ProcessLog.Module.ITO.FileName = process}

\textbf{Important:} Avoid using multi-command semaphore files unless you are sure that each command works without error when submitted in a single-command semaphore file. For more information, see "Semaphore Error Messages".

Semaphore Error Messages

When a semaphore command is executed correctly, the registry entry is removed and a success message is written to the process log.

If no command in a semaphore file can be executed correctly, the warning message \texttt{Semaphore was not processed; check spelling} is written to the process log.

\textbf{Note:} When processing a multi-command semaphore file, if at least one command in the file runs successfully, the pipeline does not log a message indicating that a command has failed.

For more information on the process log, see "About Pipeline Manager Log Files".

Using Events to Start External Programs

To use pipeline events to trigger external programs, use the Event Handler. See "About the Event Handler" in \textit{BRM Concepts}.

\textbf{Note:}
Using Events to Start External Programs

- See the module reference documentation to find the events that a module sends. For example, to find the events that the DAT_ExchangeRate module uses, see "DAT_ExchangeRate" in BRM Configuring Pipeline Rating and Discounting.

Events are named like this:
- EVT_RELOAD_SUCCESSFUL
- EVT_RELOAD_FAILED

- You can configure modules to send custom events to the Event Handler by using iScripts. For information, see "Creating iScripts and iRules" in BRM Developer’s Guide.

About Mapping Events to Programs

You map events to programs by using the registry file’s Event subsection.

The Events subsection specifies the module and event combinations can trigger an external program. Use the following syntax to create the Events subsection:

```
Events
{
    ModuleSendingEvent
    {
        EventName = Action
        EventName = Action
        TimeToWait = WaitValue
    }
}
```

where:

- **ModuleSendingEvent** specifies the registry name of the module that sends the event to the Event Handler. Add an entry for each module that can trigger an external program.

  You can use wild cards (*) to specify multiple modules. For example, use ifw.Pipelines.* to specify all modules nested under the ifw.Pipelines section of the registry file.

- **EventName** specifies the event that triggers an external program. Add an entry for each event that triggers an external program.

- **Action** specifies the external program that is triggered by the event. Specify both the path and file name of the script or program.

- **WaitValue** specifies the time in seconds that the Event Handler waits for the external program to terminate. See "Controlling External Programs".

For example:

```
Events
{
    {
        EVT_ReloadSuccess = ./script/script_1
        EVT_ReloadFailed = ./script/script_2
        TimeToWait = 30
    }
}
```
About Pipeline Manager Transactions

---

**Note:** You cannot change this event-to-program mapping while Pipeline Manager is running. To map an event to a new script or change the existing mapping, you must edit the registry file and stop and restart Pipeline Manager.

---

**Controlling External Programs**

Use the `TimeToWait` registry entry to specify the time in seconds that the Event Handler waits for the external program to terminate. If the program doesn’t terminate before the `TimeToWait` period ends, the external program is killed.

If an event is received while an external program is running, the event is queued and is started after the running program terminates.

When this option is specified, only one external program can be run at a time.

If `TimeToWait` is not enabled, the Event Handler does not wait for the external program to finish its job. Instead it starts new external programs depending on the events in the queue.

By default, no `TimeToWait` value is assumed.

---

**About Running External Programs**

The Event Handler can run only one external program at a time. If the Event Handler receives an event while an external program is running, it queues the event until the program terminates.

---

**Troubleshooting Event Handling**

You can log the events that a data module receives. This allows you to test event logging. To do so, set the data module’s `LogEvents` registry entry to `True`. By default, event logging is off.

---

**Note:** Not all data module support event logging. See the documentation for the data module that you are configuring.

---

**About Pipeline Manager Transactions**

Pipeline Manager uses transactional processing to ensure data integrity. When a system crash or power outage occurs, Pipeline Manager performs an automatic rollback and continues processing. In most cases, the last CDR file that was being processed is rolled back and processed again.

In some cases, Pipeline Manager recognizes an inconsistent state of the file system; for example, an output file is missing. In these cases, Pipeline Manager does not restart and gives an error message.

---

**Note:** A transaction can consist of one CDR file or multiple CDR files. You define the number of CDR files in a transaction by configuring the `UnitsPerTransaction` entry. For information, see “Combining Multiple CDR Files into One Transaction”.

---

Pipeline Manager uses two components for transaction handling:
The Transaction Manager handles transactions for a single pipeline. See "About the Transaction Manager".

The Transaction ID Controller manages transaction IDs for the entire Pipeline Manager instance. See "Configuring the Transaction ID Controller".

**About the Transaction Manager**

The Transaction Manager is a mandatory pipeline component that coordinates the state of all transactions in one pipeline.

The Transaction Manager performs the following functions:

- Monitors a transaction’s state. Transactions move through these three states:
  - Opened (started)
  - Prepared
  - Closed (ended)
- Persists state information to the binary log file. For information, see "About Transaction Log Files".

When a transaction is in progress, the following occurs:

1. The Input Controller notifies the Transaction Manager that a transaction started.
2. The Transaction Manager requests a transaction ID number from the Transaction ID Controller. See "Configuring the Transaction ID Controller".
3. The Transaction ID Controller issues the next ID number to the Transaction Manager.
4. The Input Controller, function modules, and Output Controller process the input stream and notify the Transaction Manager if any of the following are required:
   - Rollback. If a rollback is required, the Transaction Manager rolls back the transaction and undoes all changes.
   - Cancel. If a cancel is required, the Transaction Manager undoes all changes made during the transaction.
5. The Output Controller notifies the Transaction Manager that the transaction ended.
6. The Transaction Manager requests the Input Controller, function modules, and Output Controller to prepare for a commit of the transaction.
7. The Transaction Manager performs one of the following:
   - If all of the modules prepare successfully, the Transaction Manager commits the transaction.
   - If the prepare fails, the Transaction Manager rolls back the transaction.

Two special types of EDRs are used for managing transactions:

- Before EDRs are processed, a `begin transaction EDR` is created. This tells Pipeline Manager which EDRs are part of the transaction.

**Note:** When redo is enabled, the Transaction Manager also cancels any newly opened transactions.
After all EDRs are processed, an end transaction EDR is created. When this EDR arrives at the output, the transaction can be committed.

You configure your Transaction Managers by using the TransactionManager section of the registry file. For information, see "Transaction Manager" in BRM Configuring Pipeline Rating and Discounting.

About Cancelling Transactions When a Rollback Occurs

Use the Transaction Manager RedoEnabled registry entry to cancel all open transactions in the event of a rollback.

When a rollback is demanded, the Transaction Manager performs the following:

1. Disables the creation of new transactions.
2. Rolls back all attached modules.
3. Cancels any open transactions.
4. Re-enables the creation of new transactions.

When RedoEnabled is disabled, the Transaction Manager only rolls back the attached modules.

About Transaction Log Files

All dynamic data, for example, aggregation results, call assembling records, and duplicate check data, is always kept in main memory. In addition, to ensure transactional integrity, data in memory has to be made persistent. To do so, transactional modules write data to work files. Data in the work files is used to record the status of the transaction.

Each Transaction Manager generates its own binary log file, which stores information about a pipeline’s currently open transactions. The Transaction Manager writes information to the file when a transaction starts or changes state and deletes the transaction from the file when it ends. Thus, the file’s size changes constantly.

The binary log file stores the following for each open transaction:

- The transaction’s starting timestamp.
- Transaction ID number.
- The list of CDR files that make up the transaction.
- Whether any of the following occurred:
  - Rollback
  - Cancel
  - Redo
  - Prepare

You should regularly back up binary log files. These files are needed when you stop and restart Pipeline Manager to resolve any open transactions at the time of failure.

**Note:** When you stop and restart Pipeline Manager after an ungraceful shutdown, the Transaction Manager commits all prepared transactions and rolls back all other uncommitted transactions.
Configuring the Transaction ID Controller

You configure the Transaction ID Controller by using the `ifw.TransactionIDController` section of the registry file. For information, see "About the Transaction ID Controller" in BRM Concepts and "Transaction ID Controller" in BRM Configuring Pipeline Rating and Discounting.

About Storing IDs in Cache

When the Transaction ID Controller needs to cache a block of IDs, it:

1. Accesses the state file or table for the increment value and last issued ID number.
2. Caches the next block of transaction IDs.
   - For example, if the last ID is 200 and the increment value is 100, the Transaction ID Controller caches IDs 201 through 300.
3. Resets the last ID number in the state table or file.
   - In the example above, the Transaction ID Controller sets the last ID to 300.

You configure the number of IDs stored in cache by using the Increment registry entry.

About the Transaction ID State File and Table

The state file or table stores the last issued transaction ID number and the configured increment value. You configure where the data is stored by using the Source registry entry.

When you configure the Transaction ID Controller to use a file, the data is stored in the file and directory you specify in the registry.

When you configure the Transaction ID Controller to use the database, the data is stored in the IFW_TAM table, which is automatically created in the Pipeline Manager database by the Pipeline Manager installer.

---

**Caution:** If you configure the Transaction ID Controller to store IDs in the database, only one Pipeline Manager instance at a time can access the Pipeline Manager database. This can reduce transaction processing performance.

---

You should back up the transaction ID state file or table regularly. This state information is needed to ensure that your system continues to create unique, system-wide IDs when you stop and restart Pipeline Manager.

Configuring Sequence Checking

Sequence checking ensures that a CDR file is not processed more than once. You configure your Sequencers by using the `ifw.SequencerPool` registry entries, and you assign Sequencers to pipelines by using the pipeline Output registry entries. See "About the Sequencer" in BRM Concepts.

Sequence Numbers in the Header Record

The Header record in the EDR container includes two fields for sequence numbers:

- `SEQUENCE.NUMBER`. This is a unique reference that identifies each file. It indicates the file number of the specific file type, starting at 1 and incrementing by one for each new file of that type sent. Separate sequence numbering must be used
for test and chargeable data. Having reached the maximum value (999999), the number restarts at 1.

---

**Note:** In the case of retransmission, this number is not incremented.

---

- **ORIGIN_SEQUENCE_NUMBER.** This is the original file sequence number as generated the first time. It is the same as SEQUENCE_NUMBER, but is never changed. It is used as a reference to the original file, if any processor has changed the file sequence number.

---

**Deciding Whether to Use Sequencers**

You should add Sequencers to your system when:

- You want to check for duplicate CDR files.
- Your CDR software doesn’t automatically generate sequence numbers.
- Your pipelines split CDR files into multiple output files.

---

**About Sequence Checking**

When performing sequence checking, the Sequencer:

1. Receives the CDR file from the input module.
2. Checks for duplicates by comparing the sequence number in the stream’s header with the sequence numbers in the state file or state table. See "Sequencer Log Files and Log Tables".
   - When the number is a duplicate, the Sequencer rejects the CDR file and rolls back the transaction.
   - When the number is not a duplicate, it passes the transaction directly to the Output Collection module. See "About Configuring the Output Section in the Registry" in BRM Configuring Pipeline Rating and Discounting.
3. Checks for gaps in sequence numbers by comparing the sequence number in the stream’s header with the last sequence number in the state file or state table. If the sequence number is more than one digit greater than the previous number, a gap is identified. The Sequencer logs a message and stores the unused number in the state file or state table. See "Sequencer State Files and State Tables".

---

**Note:** By default, the Sequencer:

- Allows gaps in sequence numbers (caused by canceled or rolled back transactions). You can direct the Sequencer to reuse these number gaps by using the **Controller.ReuseGap** registry entry.
- Does not start the gap in sequence numbers from 0. For example, if the first sequence number is 3, the Sequencer does not start the gap for the skipped sequence numbers from 0 (that is, gap of 1, 2). You can direct the Sequencer to add a gap for the skipped sequence numbers starting from 0 by using the **Controller.UseGapAtStartup** registry entry.

See "Sequencer" in BRM Configuring Pipeline Rating and Discounting.
To configure the Sequencer to perform sequence checking, set the SequencerType registry entry to **Check**.

**About Sequence Generation**

When performing sequence generation, the Sequencer:

1. Receives the CDR file from the input module.
2. Assigns the next sequence number to the output file. To obtain this number, the Sequencer reads the last generated sequence number in the state file or state table and increments it by one.

This process continues for each CDR file until the maximum value is reached. For information, see "About Maximum and Minimum Sequence Numbers".

---

**Note:** If you configure the Sequencer to reuse gap numbers, it assigns unused gap numbers to the output file before assigning new sequence numbers. See "Sequencer" in **BRM Configuring Pipeline Rating and Discounting**.

---

To configure the Sequencer to perform sequence generation, set the SequencerType registry entry to **Generation**.

**About Maximum and Minimum Sequence Numbers**

The Sequencer generates numbers by starting at the configured minimum value and then incrementing by one until it reaches the configured maximum value. After the Sequencer uses the maximum value, you must manually reset the sequence number to the minimum value.

For example, if the minimum value is 1 and the maximum value is 10,000, the Sequencer assigns 1 to the first output file, 2 to the second output file, 3 to the third output file, and so on. When the sequencer assigns 10,000 to the ten-thousandth output file, you must manually reset the sequence number to 1 by changing the following fields in the IFW_SEQCHECK table:

- Set the **seq_orignumber** field to **0**.
- Set the **seq_gapnumbers** field to **-1**.

---

**Important:** To prevent the Sequencer from incorrectly rejecting files as duplicates after you manually reset the sequence number to the minimum value, remove all the rows from the IFW_SEQLOG_IN table.

---

To configure the maximum and minimum values, do one of the following:

- **State files.** Edit the **MaxSequenceNumber** and **MinSequenceNumber** entries in the state file. The default minimum value is 0; the default maximum value is 99999.

- **State tables.** Use Pricing Center to set these values as described for defining a sequence generation in **BRM Pricing Center Online Help**.

---
About Recycled EDRs

CDR input files sometimes contain non-valid EDRs, which are rejected by the pipeline. When you recycle the input file through a pipeline to process any rejected EDRs, the file's original sequence number is no longer correct. The Sequencer automatically assigns new sequence numbers to recycled files to prevent them from being rejected as duplicates.

For more information about recycling, see "About Standard Recycling" in PI and "Recycling EDRs in Pipeline-Only Systems" in *BRM Configuring Pipeline Rating and Discounting*.

About Sequencer Files and Tables

Each Sequencer generates its own state and logging information, which can be stored in files or tables. You configure where state and logging information is stored by using the registry file. For information, see "Sequencer" in *BRM Configuring Pipeline Rating and Discounting*.

---

**Important:** When you store state and logging information in files, the Sequencer checks for duplicates by comparing the current sequence number against the last checked sequence number only. When you use tables, the Sequencer compares the number against all previously checked sequence numbers. For this reason, Oracle recommends using tables for production systems and using files only when testing your system in a development environment.

---

When you configure Sequencers to store logging information in files, all logging and state data is stored in the file and directory you specify in the registry file.

When you configure Sequencers to use tables, all logging and state data is stored in the database tables listed in Table 7–11, which are automatically created by the Pipeline Manager installer:

<table>
<thead>
<tr>
<th>Table name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IFW_PIPELINE</td>
<td>Stores information about pipelines.</td>
</tr>
<tr>
<td>IFW_SEQCHECK</td>
<td>Stores the state of the Sequencer.</td>
</tr>
<tr>
<td>IFW_SEQLOG_OUT</td>
<td>Stores sequence generation log information.</td>
</tr>
<tr>
<td>IFW_SEQLOG_IN</td>
<td>Stores sequence checking log information.</td>
</tr>
</tbody>
</table>

You use Pricing Center to provide input to IFW_SEQCHECK and to view log information stored in IFW_SEQLOG_OUT and IFW_SEQLOG_IN. See Pricing Center Help.

Sequencer State Files and State Tables

Sequencer state files and state tables store the following information:

- The last generated sequence number
- The last checked sequence number
- Maximum and minimum sequence numbers
You should back up state files and state tables periodically. This information is needed to ensure that your system doesn’t process duplicate CDR files when you stop and restart Pipeline Manager.

**Sequencer Log Files and Log Tables**

Sequencer log files and log tables store an entry for each sequence number that is checked or generated.

**Important:** When the Sequencer reaches the maximum generated sequence number, delete all log entries. Otherwise, your log will contain duplicates. For more information, see "About Maximum and Minimum Sequence Numbers".

**Tip:** Log files and log tables grow indefinitely, so you should trim them periodically to reduce disk usage.

**Checking and Generating Sequence Numbers**

You can use Sequencers to configure pipelines to check for duplicate CDR input files and to check for gaps in sequence numbers. You can also configure pipelines to use Sequencers to generate sequence numbers. For information, see "Configuring Sequence Checking".

To enable sequence checking or sequence generation in a pipeline, perform the following tasks:

1. Configure your Sequencers by editing the **SequencerPool** section of the registry file. Make sure you specify the following:
   - The Sequencer name.
   - Whether Sequencer data is stored in a database table or files.
   - How to connect to the database or the path and file name of the Sequencer files.
   - Whether the Sequencer performs sequence checking or sequence generation. Each Sequencer performs only one of these functions.

   For information, see "Sequencer" in BRM Configuring Pipeline Rating and Discounting.

2. For sequence generation, set minimum and maximum sequence numbers by doing one of the following:
   - If you configured the Sequencer to store data in a *database*, use Pricing Center to set these values. See Pricing Center Help.
   - If you configured the Sequencer to store data in *files*, set the **MaxSequenceNumber** and **MinSequenceNumber** entries in the Sequencer state file. For information, see "About Maximum and Minimum Sequence Numbers".

**Note:** The default minimum value is 0, and the default maximum value is 99999.
3. Assign Sequencers to pipeline output streams:

- To assign a sequence checker to an output stream, edit the Sequencer registry entry in the Pipeline Output Controller. Specify the name of the Sequencer assigned to the output stream:

```
Output
{
    ...
    Sequencer = SequenceCheckerName
    ...
}
```

For information, see "Output Controller" in BRM Configuring Pipeline Rating and Discounting.

- To assign a sequence generator to an output stream, edit the Sequencer registry entry in the output module. Specify the name of the Sequencer assigned to the output stream:

```
OutputStreamName
{
    ModuleName = OUT_GenericStream
    Module
    {
        Sequencer = SequenceGeneratorName
    }
}
```

For information, see "OUT_GenericStream" in BRM Configuring Pipeline Rating and Discounting.

### Configuring the NET_EM Module for Real-Time Processing

You can use Pipeline Manager for real-time discounting, real-time zoning, and real-time rerating. See "About the Pipeline Manager System Architecture" in BRM Concepts.

The NET_EM module provides a link between the Connection Manager (CM) and the pipelines. You configure the NET_EM module in the data pool.

To configure the NET_EM module, you configure connection information such as the port number and threads, and you configure the OpcodeName section for each type of real-time processing: discounting, rerating, and zoning.

In this example, you configure the real-time discounting by specifying the PCM_OP_RATE_DISCOUNT_EVENT opcode:

```
ifw
{
    ...
    DataPool
    {
        RealtimePipeline
        {
            ModuleName = NET_EM
            Module
            {
                ThreadPool
                {
                    Port = 14579
                }
            }
        }
    }
}
```
Each NET_EM module can perform one type of processing; for example, discounting, rerating, or zoning. You must configure a separate instance of Pipeline Manager for each NET_EM module.

You can configure multiple instances of the same type of NET_EM processing, for example, multiple rerating Pipeline Manager instances. You can then configure the CM to point to all the NET_EM modules. When multiple rerating pipeline instances are configured, the NET_EM module routes rerate requests to whichever of these pipeline instances is available.

To configure the NET_EM module:

1. Configure the NET_EM module in the registry. See "Configuring the NET_EM Module".
2. Configure the CM to send data to the NET_EM module. See "Configuring the CM to Send Real-Time Requests to the NET_EM Module".

Configuring the NET_EM Module

The NET_EM module receives various types of requests from the CM and routes the requests to the appropriate pipeline. See "NET_EM" in BRM Configuring Pipeline Rating and Discounting.

Specifying the Type of NET_EM Opcode Processing

To specify the type of processing the NET_EM module is used for, use the OpcodeName entry.

For real-time discounting, use:

```
OpcodeName = PCM_OP_RATE_DISCOUNT_EVENT
```

For real-time zoning, use:

```
OpcodeName = PCM_OP_RATE_GET_ZONEMAP_INFO
```

For real-time rerating, use:

```
OpcodeName = PCM_OP_RATE_PIPELINE_EVENT
```

Configuring the CM to Send Real-Time Requests to the NET_EM Module

To configure the CM to send rerate requests to the NET_EM module:

1. Open the CM configuration file (BRM_Home/sys/cm/pin.conf).
2. For real-time rerating, make sure the following entry is uncommented:
3. Edit the discounting **em_group** entry:

   ```
   - cm em_group em_type Opcode_name
   ```

   where:
   - **em_type** is the type of real-time processing; for example, discounting, zoning, or rerating. You can enter any string up to 15 characters. This entry must match the entry in the **em_pointer** entry.
   - **Opcode_name** is the opcode used.

   For discounting, use:
   ```
   - cm em_group discounting PCM_OP_RATE_DISCOUNT_EVENT
   ```

   For zoning, use:
   ```
   - cm em_group zoning PCM_OP_RATE_GET_ZONEMAP_INFO
   ```

   For rerating, use:
   ```
   - cm em_group rating PCM_OP_RATE_PIPELINE_EVENT
   ```

4. Edit the discounting **em_pointer** entry to match your environment, for example:

   ```
   - cm em_pointer discounting ip cm_host 11945
   - cm em_pointer zoning ip cm_host 11945
   - cm em_pointer rating ip cm_host 11945
   ```

   Instructions for this entry are included in the file.

   You can enter multiple **em_pointer** entries. If the first NET_EM module is unavailable, the CM connects to a different NET_EM module.

   **Note:** To run multiple NET_EM instances, you need to run multiple instances of Pipeline Manager. You use only one NET_EM module for each instance of Pipeline Manager.

5. Save the file.

6. Stop and restart the CM. See “Starting and Stopping the BRM System”.

### About Pipeline Manager Log Files

The log module is an optional pipeline component that generates and manages your system log files, which consist of the logs listed in **Table 7–12**:

<table>
<thead>
<tr>
<th>Log file</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process log</td>
<td>Contains general system messages for the pipeline framework, such as startup, shutdown, version numbers of modules, and semaphore file messages. The module generates one process log for the entire pipeline framework.</td>
</tr>
</tbody>
</table>
You configure your system log files by editing the registry file. You create a set of log module registry entries for each type of log file you want your system to generate. For example, to configure your system to generate all three system log files, you create one set of entries for the process log, one set for the pipeline log, and one set for the stream log.

- You configure the process log in the `ProcessLog` registry section.
- You configure the pipeline log in the `PipelineLog` registry section for each pipeline.
- You configure the stream log in the `OutputLog` registry section for each pipeline.

For information, see "LOG" in BRM Configuring Pipeline Rating and Discounting.

In addition to the log files handled by the log module:

- All processed sequence numbers of the EDR streams are logged in the sequence log file. See "Sequencer Log Files and Log Tables".
- All processed transactions are logged in the transaction log file. See "About Transaction Log Files".

### Pipeline Manager Log File Registry Entries

The registry entries listed in Table 7–13 control Pipeline Manager log files.

<table>
<thead>
<tr>
<th>Entry</th>
<th>Module</th>
<th>Log file</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BalanceLockStatusLog</td>
<td>DAT_BalanceBatch</td>
<td>Process log</td>
<td>Specifies that when an event transaction is locked by an EDR transaction, it is logged to the Process log.</td>
</tr>
<tr>
<td>BinaryLogFileName</td>
<td>Transaction Manager</td>
<td>User specified</td>
<td>Specifies the path and file name of the binary log file, which is used to persist and restore open transactions.</td>
</tr>
<tr>
<td>InfranetPool</td>
<td>DAT_ConnectionPool</td>
<td>-</td>
<td>Specifies whether to log debug messages.</td>
</tr>
</tbody>
</table>
### About Error Message Files

You use error message files to define the errors generated by your pipeline modules. All modules have their own error message file (\*.msg), which is installed by default in the `Pipeline_Home/etc` directory.

The default error message files already define all of the module error codes, but you can add custom error codes or change the existing definitions by editing the files.

Error message file entries use the following format:

```
[messageName] | [messageText] | [messageNumber]
```

where:

- `messageName` specifies the module error code. For example, `ERR_WRITE_FILE`.
- `messageText` specifies the message text to write to the log file.
- `messageNumber` specifies the error number to write to the log file. The default is 0.

For example, the DAT_AccountBatch module uses the `Pipeline_Home/etc/DAT_AccountBatch.msg` message file. This file includes the following entries:

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Message Text</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_LISTENER_NOT_FOUND</td>
<td>Listener '%s' not found.</td>
<td></td>
</tr>
<tr>
<td>INF_STARTED_LOADING</td>
<td>Started loading account data.</td>
<td></td>
</tr>
<tr>
<td>INF_ENTRIES_LOADED</td>
<td>%s %s loaded.</td>
<td></td>
</tr>
<tr>
<td>INF_FINISHED_LOADING</td>
<td>Finished loading account data.</td>
<td></td>
</tr>
</tbody>
</table>

See also "Collecting Diagnostic Information by Using RDA".

### About Error Message Files (Cont.)

**Table 7-13** (Cont.) Pipeline Manager Log File Registry Entries

<table>
<thead>
<tr>
<th>Entry</th>
<th>Module</th>
<th>Log file</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LogEvents</td>
<td>DAT_AccountBatch</td>
<td>Pipeline log</td>
<td>Specifies whether received events should be written to a log file. Use this entry to troubleshoot Pipeline Manager event handling.</td>
</tr>
<tr>
<td></td>
<td>DAT_BalanceBatch</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>DAT_Listener</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>DAT_PriceModel</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>DAT_Rateplan</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>DAT_Recycle</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>DAT_ResubmitBatch</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Logging</td>
<td>FCT_Opcode</td>
<td>Pipeline log</td>
<td>Logs each opcode called from the processing pipeline.</td>
</tr>
<tr>
<td>LogTestResults</td>
<td>FCT_Suspense</td>
<td>-</td>
<td>Determines whether the results of test recycling are logged.</td>
</tr>
<tr>
<td>LogTransactions</td>
<td>DAT_BalanceBatch</td>
<td>Process log</td>
<td>Specifies if the balances affected during the CDR processing are logged.</td>
</tr>
<tr>
<td>LogZoneModelNotFoundEntries</td>
<td>FCT_USC_Map</td>
<td>Stream log</td>
<td>Specifies that all log entries in INF_NO_USC_MAPPING_ENTRY are logged into the Stream log.</td>
</tr>
<tr>
<td>RecycleLog</td>
<td>FCT_Recycle</td>
<td>-</td>
<td>Specifies the log file parameters.</td>
</tr>
<tr>
<td>WriteToLogEnabled</td>
<td>Transaction Manager</td>
<td>Pipeline log</td>
<td>Specifies whether the Transaction Manager writes status information to the pipeline log file.</td>
</tr>
</tbody>
</table>
About Log File Contents

The LOG module logs the following information to the system log file in ITO format:

- Date
- Time
- Node
- Application name
- Message group
- Severity
- Error number
- Text

For example:

03.10.2002 08:18:42 system ifw INTEGRATE NORMAL
00000 - No registry entry 'MultiThreaded (default is true)' found.

Troubleshooting Pipeline Modules

You can troubleshoot problems in the pipeline modules by writing the contents of the EDRs generated by various pipeline modules into a log file. The file shows how each module accessed the EDR and the changes each module made to the EDR. You can read the log file to check if the pipeline modules processed the EDRs as expected and correct any problems you find.

Use the EdrTrace entry in the pipeline registry file to write the contents of the EDR to a file. You can configure EdrTrace to write the EDR contents to a file for specific modules that you want to debug. The EdrTrace entry includes the parameters listed in Table 7–14:

Table 7–14 EdrTrace Log File Registry Entries

<table>
<thead>
<tr>
<th>Entry</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EDRTraceEnabled</td>
<td>Enables or disables EDR trace:</td>
</tr>
<tr>
<td></td>
<td>True enables EDR trace.</td>
</tr>
<tr>
<td></td>
<td>False disables EDR trace.</td>
</tr>
<tr>
<td></td>
<td>The default is False.</td>
</tr>
<tr>
<td>EdrTrace</td>
<td>Specifies the EDR trace.</td>
</tr>
</tbody>
</table>
Table 7–14  (Cont.) EdrTrace Log File Registry Entries

<table>
<thead>
<tr>
<th>Entry</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TraceLog</td>
<td>Specifies the following information about the EDR log file:</td>
</tr>
<tr>
<td></td>
<td>- <strong>FilePath</strong>. The path to the log file. The default is /ifw/log/edrLog.</td>
</tr>
<tr>
<td></td>
<td>- <strong>FileName</strong>. The name of the log file. The default is edrdump.</td>
</tr>
<tr>
<td></td>
<td>- <strong>FilePrefix</strong>. The prefix to the log file name. The default is log_.</td>
</tr>
<tr>
<td></td>
<td>- <strong>FileSuffix</strong>. The log file name extension. The default is .log.</td>
</tr>
<tr>
<td>TraceEndPoint</td>
<td>Specifies the pipeline module from which you want to start logging the EDR contents. This registry entry is mandatory.</td>
</tr>
<tr>
<td>TraceEndPoint</td>
<td>The default is Input.module.</td>
</tr>
<tr>
<td>TraceStartPoint</td>
<td>Specifies the pipeline module up to which you want to log the EDR contents.</td>
</tr>
<tr>
<td>TraceStartPoint</td>
<td>The default is Output.module.</td>
</tr>
<tr>
<td>Important:</td>
<td>If both the TraceStartPoint and TraceEndPoint registry entries are specified, the EDR log file contains changes from all the modules from TraceStartPoint to TraceEndPoint. If only TraceStartPoint is specified, the EDR log file contains changes from the module specified in that entry up to the Output module. To log EDR changes for only one module, TraceStartPoint and TraceEndPoint must specify the same module.</td>
</tr>
</tbody>
</table>

**Writing EDR Contents to a Log File**

To write the contents of the EDR to a log file and use it to debug pipeline modules, include the **EdrTrace** entry by using the following syntax:

```plaintext
...  
Output
{
...
  EdrTraceEnabled = value  
  EdrTrace  
  {
    TraceLog  
    {
      FilePath = file_path  
      FileName = file_name  
      FilePrefix = prefix  
      FileSuffix = suffix  
    }  
    TraceStartPoint = Functions.Processing.FunctionPool.module_name.module  
    TraceEndPoint = Functions.Processing.FunctionPool.module_name.module  
  }
}
```

**Using a Semaphore to Write EDR Contents to a File for Debugging**

You can change the EDR trace by sending a semaphore to the Output Controller module at run time without stopping the pipeline. You can perform the following changes to the **EdrTrace** entry through a semaphore:

- Enable or disable logging the EDR contents.
- Change **TraceStartPoint** and **TraceEndPoint** for logging the EDR contents.

To change the EDR content logging at run time, send a semaphore with the following syntax:
Sample EDR Content Log File

The following sample output of `EdrTrace` shows EDR contents from Input to Output modules:

```java
ifw.Pipelines.pipeline_name.Output.EdrTrace
{
    TraceStartPoint = new_start_value
    TraceEndPoint = new_end_value
}
```

---

Configuring Pipeline Manager 7-43
ifw.Pipelines.ALL_RATE.Input : DETAIL.CHARGING_START_TIMESTAMP : 20061115101900 :
    : getDate
ifw.Pipelines.ALL_RATE.Input : DETAIL.DURATION : 300 :
    : getDecimal
ifw.Pipelines.ALL_RATE.Input : DETAIL.CHARGING_END_TIMESTAMP : 20061115102400 :
    : setDate
ifw.Pipelines.ALL_RATE.Input : DETAIL.CHARGING_END_TIMESTAMP : 20061115102400 :
    : getDate
ifw.Pipelines.ALL_RATE.Input : DETAIL.CHARGING_START_TIMESTAMP : 20061115101900 :
    : getDate
ifw.Pipelines.ALL_RATE.Input : DETAIL.CHARGING_START_TIMESTAMP : 20061115101900 :
    : getDate
ifw.Pipelines.ALL_RATE.Input : DETAIL.CHARGING_START_TIMESTAMP : 20061115101900 :
    : getDate
ifw.Pipelines.ALL_RATE.Input : DETAIL.CHARGING_START_TIMESTAMP : 20061115101900 :
    : getDate
ifw.Pipelines.ALL_RATE.Input : DETAIL.THRENECHARGING_END_TIMESTAMP : 20061115102400 :
    : setDate
ifw.Pipelines.ALL_RATE.Input : DETAIL.THRENECHARGING_END_TIMESTAMP : 20061115102400 :
    : getDate
ifw.Pipelines.ALL_RATE.Input : DETAIL.RETAIL_CHARGED_AMOUNT_VALUE : 0.0 :
    : getDecimal
ifw.Pipelines.ALL_RATE.Input : DETAIL.WHOLESALE_CHARGED_AMOUNT_VALUE : 0.0 :
    : getDecimal
ifw.Pipelines.ALL_RATE.Input : DETAIL.CHARGING_START_TIMESTAMP : 20061115101900 :
    : getDate
ifw.Pipelines.ALL_RATE.Input : DETAIL.CHARGING_START_TIMESTAMP : 20061115101900 :
    : getDate
ifw.Pipelines.ALL_RATE.Input : DETAIL.CHARGING_START_TIMESTAMP : 20061115101900 :
    : getDate
ifw.Pipelines.ALL_RATE.Input : DETAIL.CHARGING_START_TIMESTAMP : 20061115101900 :
    : getDate
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_TYPE_OF_NUMBER : 0 :
    : getLong
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_MODIFICATION_INDICATOR : 00 :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_NUMBER : 0049100052 :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_NUMBER : 0049100052 :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_NUMBER : 0049100052 :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_NUMBER : 0049100052 :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.B_TYPE_OF_NUMBER : 0 :
    : getLong
ifw.Pipelines.ALL_RATE.Input : DETAIL.B_MODIFICATION_INDICATOR : 00 :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.B_NUMBER : 0049100056 :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.B_NUMBER : 0049100056 :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.B_NUMBER : 0049100056 :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.B_NUMBER : 0049100056 :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.C_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.RECORD_TYPE : 020 :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_TYPE_OF_NUMBER :
    : getLong
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_MODIFICATION_INDICATOR :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.B_TYPE_OF_NUMBER :
    : getLong
ifw.Pipelines.ALL_RATE.Input : DETAIL.B_MODIFICATION_INDICATOR :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.B_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.B_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.B_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.B_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.C_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.RECORD_TYPE :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_TYPE_OF_NUMBER :
    : getLong
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_MODIFICATION_INDICATOR :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.B_TYPE_OF_NUMBER :
    : getLong
ifw.Pipelines.ALL_RATE.Input : DETAIL.B_MODIFICATION_INDICATOR :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.B_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.B_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.B_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.B_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.C_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.RECORD_TYPE :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_NUMBER :
    : getString
ifw.Pipelines.ALL_RATE.Input : DETAIL.A_NUMBER :
    : getString
    DETAIL.ASS_PROCESS_STATUS : 0 :
    : getLong
    DETAIL.ASS_SUSPENSE_EXT.PIPELINE_NAME.0 :
    : ALL_RATE
    : setString
    DETAIL.ASS_SUSPENSE_EXT.PIPELINE_NAME.0 :
    : ALL_RATE
    : setString
    DETAIL.ASS_SUSPENSE_EXT.PIPELINE_NAME.0 :
    : ALL_RATE
    : setString
    DETAIL.ASS_SUSPENSE_EXT.PIPELINE_NAME.0 :
    : ALL_RATE
    : setString
    DETAIL.ASS_SUSPENSE_EXT.PIPELINE_NAME.0 :
    : ALL_RATE
    : setString
    DETAIL.ASS_SUSPENSE_EXT.PIPELINE_NAME.0 :
    : ALL_RATE
    : setString
    DETAIL.ASS_SUSPENSE_EXT.PIPELINE_NAME.0 :
    : ALL_RATE
    : setString
    DETAIL.BASIC_SERVICE : TEL :
    : getString
Using Perl Scripts to Administer Pipeline Manager

Pipeline Manager includes a set of Perl scripts, and associated semaphore files, that you can use to start and stop various types of pipelines and perform other system administration tasks.

Table 7–15 describes the files and scripts used for controlling pipelines:
### Pipeline Manager Administration Perl Scripts

<table>
<thead>
<tr>
<th>Semaphore and Perl script file names</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dump_portal_act_data.reg</td>
<td>Outputs account data for all accounts currently in memory. By default, data is written to the cust.data file, located in the directory where you launch Pipeline Manager. Runs the DAT_Account module PrintData semaphore.</td>
</tr>
<tr>
<td>dump_portal_act_data.pl</td>
<td>Runs the DAT_Account module PrintData semaphore.</td>
</tr>
<tr>
<td>off_queue_buffer.reg</td>
<td>Disables logging of the messages processed by the queue. Sets the DAT_Listener module LogEvents entry to False.</td>
</tr>
<tr>
<td>off_queue_buffer.pl</td>
<td>Runs the DAT_Listener module LogEvents entry to False.</td>
</tr>
<tr>
<td>reload_portal_act_data.reg</td>
<td>Reloads accounts from the BRM database. Runs the DAT_Account module Reload semaphore.</td>
</tr>
<tr>
<td>reload_portal_act_data.pl</td>
<td>Runs the DAT_Account module Reload semaphore.</td>
</tr>
<tr>
<td>reload_price.reg</td>
<td>Reloads all the price models and rate plans. Runs the DAT_Price module Reload semaphore.</td>
</tr>
<tr>
<td>reload_price.pl</td>
<td>Runs the DAT_Price module Reload semaphore.</td>
</tr>
<tr>
<td>reload_zone.reg</td>
<td>Reloads all the zones and zone model data. Runs the DAT_Zone module Reload semaphore.</td>
</tr>
<tr>
<td>reload_zone.pl</td>
<td>Runs the DAT_Zone module Reload semaphore.</td>
</tr>
<tr>
<td>set_call_ass_limit.reg</td>
<td>Sets a new flush limit for call assembly (by default, 30). Runs the FCT_CallAssembling module FlushLimit semaphore.</td>
</tr>
<tr>
<td>set_call_ass_limit.pl</td>
<td>Runs the FCT_CallAssembling module FlushLimit semaphore.</td>
</tr>
<tr>
<td>set_dup_check_limit.reg</td>
<td>Sets a new limit for duplicate checking. If you do not specify any parameter, it sets the BufferLimit entry to three days before the current date, and it sets the StoreLimit entry to seven days before the BufferLimit date. This script creates and runs the set_dup_check_limit.reg semaphore.</td>
</tr>
<tr>
<td>set_dup_check_limit.pl</td>
<td>To modify the default BufferLimit and StoreLimit values, run the script with these two parameters: set_dup_check_limit.pl buffer_limit store_limit</td>
</tr>
<tr>
<td></td>
<td>For example: set_dup_check_limit.pl 5 5</td>
</tr>
<tr>
<td></td>
<td>In this example, if today is November 28, then the buffer limit is set to November 23 and the store limit is set to November 18.</td>
</tr>
</tbody>
</table>
This document explains how to configure the Batch Controller to launch batch handlers to check or process data for your Oracle Communications Billing and Revenue Management (BRM) system.

**About the Batch Controller**

The BRM Batch Controller lets you specify when to run programs or scripts automatically, either at timed intervals or upon creation of certain files, such as log files.

The Batch Controller configuration file (BRM_Home/apps/batch_controller/Infranet.properties) has entries that tell the Batch Controller when to run the specified batch handlers. These programs or scripts can be triggered at specified times or by specified kinds of occurrences, such as creation of a new log file. You can specify scheduled execution, starting the handler at fixed times of the day only, or metronomic execution, starting the handler repeatedly at a fixed interval.

A batch handler can be any executable program or script that can be run from a command line. For more information about the requirements for writing batch handlers, see "Writing Custom Batch Handlers" in BRM Developer’s Guide.

Only one Batch Controller can run on a single computer, but it can control many batch handlers to launch various applications.

BRM installation includes a generic batch handler, called SampleHandler, which you can use with most applications. See "About SampleHandler".

**Setting Activity Times and Triggers**

Certain general parameters apply to all batch activity. Other parameters identify the batch handlers and regulate when those handlers are to be run.

**General Batch Controller Parameters**

The Batch Controller’s Infranet properties file (BRM_Home) specifies how to connect the Batch Controller to the Connection Manager (CM), when to allow high batch traffic, how much logging to record, and how long to wait for handlers.

**Connection Parameters**

Batch Controller requires the following parameters to connect to a CM: a user ID and password (specified in the infranet.connection parameter), the CM’s port number, and the address of the CM’s host computer.
In the `batch.lock.socket.addr` parameter, specify the socket number to lock on. If in doubt, check with your system administrator for the number of an unused socket that can be used exclusively by the Batch Controller. You can use the default value, 11971, unless some other application is using that socket number.

To write a stack trace of any runtime exceptions in the log file, set `infranet.log.logallebuf` to `True`. To disable this feature, set it to `False`.

**Time-to-Run Parameters**

When your system’s heaviest load typically occurs, you might want some handlers to run less often than they do when the load is lighter. The Batch Controller divides the day into high-load and low-load periods to balance the demand for system resources.

Specify the starting time of your system’s busiest period in the `batch.start.highload.time` parameter. Specify the starting time of your system’s slowest period in the `batch.end.highload.time` parameter. For each of these times, specify the hour, minute, and second, in `hhmmss`, using the 24-hour clock. For each handler, you can specify the maximum number of simultaneous actions during each of these two periods.

The `end` parameter must be greater than the `start` parameter. If you do specify `start`, it must be greater than `end`. Specifying the same value for both parameters causes an error.

In the `batch.check.interval` parameter, specify the time, in seconds, between checks for occurrences of the type specified in `batch.timed.events` or `batch.random.events`. Omitting `batch.check.interval` causes an error.

**Log-File Parameter**

For logging, you can specify the level of information reported, the full path of the log file, and whether to print a stack trace of runtime exceptions. Set `infranet.log.level` to 0 for no logging; to 1 for error messages only; to 2 for error messages and warnings; or to 3 for error messages, warnings, and debug messages. Set `infranet.log.file` to the path and file name for the Batch Controller log file.

If you omit `infranet.log.file`, BRM uses `default.pinlog` in the current directory. Omitting `infranet.log.level` causes an error.

**Timeout-Limit Parameters**

You can also set timeout limits for handlers to start their objects and to complete their execution. Set `batch.handler.start.wait` to the number of seconds allowed for the handler to update its own object status from STARTING to STARTED, and set `batch.handler.end.wait` to the number of seconds allowed for updating from STARTED to some other state, such as COMPLETED. See "Writing Custom Batch Handlers" in *BRM Developer’s Guide* for descriptions of all of the handler states.

**Tip:** To only use a user ID, set the `infranet.login.type` parameter to 0; to use both a user ID and a password, set `infranet.login.type` to 1.

**Note:** Omitting either `batch.handler.start.wait` or `batch.handler.end.wait` causes an error.
Example of Parameters

This example demonstrates the general parameters:

```
infranet.connection  pcp://root.0.0.0.1:mypass@myserver:11960/service/pcm_client
infranet.login.type  1
infranet.log.logallebuf true
batch.lock.socket.addr  11971
batch.start.highload.time  083000
batch.end.highload.time  211500
infranet.log.file /opt/portal/7.4/apps/batch_controller/batch.pinlog
infranet.log.level  2
batch.handler.start.wait  600
batch.handler.end.wait  43200
```

In this example, the Batch Controller logs on to the CM host on `myserver`, port 11960, as user `root.0.0.0.1`, using password `mypass`. High usage is expected between 8:30 a.m. and 9:15 p.m. Logging writes a normal level of information to file `batch.pinlog` and prints a stack trace if any program errors are found. Timeouts for updating object status are 10 minutes (600 seconds) for going to STARTED and 12 hours (43,200 seconds) for going to COMPLETED or some other state.

Handler Identification

To identify each of the batch handlers:

1. In the `handler_name.name` parameter, enter a descriptive label for the handler. This name can include spaces or any other characters. It can be of any length, but short names are easier to read.

2. In the `handler_name.max.at.highload.time` parameter, specify the highest number of instances of this batch handler that are permitted to run simultaneously during the time from `batch.start.highload.time` to `batch.end.highload.time`.

3. In the `handler_name.max.at.lowload.time` parameter, specify the highest number of instances of this batch handler that are permitted to run simultaneously during the low-usage time; the time from `batch.end.highload.time` to `batch.start.highload.time`.

4. In the `handler_name.start.string` parameter, specify the command line to start this batch handler.

---

**Note:** When the Batch Controller issues this command, it appends `-p handler_poid -d failed_handler_poid` to the command, as described in "Writing Custom Batch Handlers" in BRM Developer’s Guide. If you are not using custom batch handlers, you can ignore these options.

This example demonstrates the identification parameters:

```
handler1.name Visa Handler #1
               . . .
handler1.max.at.lowload.time  4
handler1.max.at.highload.time  2
handler1.start.string /opt/portal/7.4/apps/visa-handler/visa.pl

handler2.name Discover Handler #1
               . . .
handler2.max.at.lowload.time  5
```

In this example, the internal name Visa Handler #1 applies to the program started by the command string /opt/portal/7.4/apps/visa-handler/visa.pl, which runs a Perl script. The parameters in the above example limit this program to one or two concurrent actions during the specified high-load period or as many as four during the low-load period.

The other batch handler in this example, Discover Handler #1, runs the application /opt/portal/7.4/apps/discover-handler/discover with the additional option -y 2001.

**Occurrence-Driven Execution**

To trigger execution based on specified occurrences:

1. In the *batch.random.events* parameter, specify the event identifiers. If you have two or more event identifiers, separate each with a comma, but no blank space.

2. In the *event_name.name* parameter, enter a descriptive label for the event. This name can include spaces or any other characters. It can be of any length, but short names are easier to read.

3. In the *event_name.handlers* parameter, specify the identifiers of one or more handlers to trigger when the event occurs. You must specify at least one handler. If you have two or more handlers, separate each with a comma; no blank spaces are allowed.

4. In the *event_name.file.location* parameter, specify the full path name of the directory to monitor for the arrival of new files that match the pattern in *event_name.file.pattern*.

5. In the *event_name.file.pattern* parameter, specify the file name pattern to look for. You can use an asterisk (*) to represent zero or more characters in the file name. No other wild cards (metacharacters) are supported.

**Important:** Depending on your configuration, the file pattern might conflict with a file pattern used by another component, such as Rated Event Loader. To prevent conflicts, use a specific pattern, for example, test*.out.

You can also check the Pipeline Manager registry to see if any temporary file output patterns conflict with this entry.

**Caution:** You must specify *batch.timed.events* or *batch.random.events* or both. Specifying neither causes the Batch Controller to shut down just after it starts because it has no tasks to perform.

When the Batch Controller starts, it tests the file name pattern against every file name in the specified directory and runs the batch handler for each file where the name matches the pattern. It then monitors the files entering that directory and runs the batch handler whenever it finds a match.
This example demonstrates the occurrence parameters:

```
batch.random.events event1,event3

...  
event1.name Random Discover file arrival
event1.handlers handler1
event1.file.location /apps/discover/stagingDir
event1.file.pattern *.6011.*

...  
event3.name Random Visa file arrival
event3.handlers handler3
event3.file.location /apps/visa/stagingDir
event3.file.pattern *.dat

...  
handler1.name Discover UEL Handler
handler1.max.at.lowload.time 6
handler1.max.at.highload.time 3
handler1.start.string /apps/discover -uel

...  
handler3.name Visa UEL Handler
handler3.max.at.lowload.time 8
handler3.max.at.highload.time 4
handler3.start.string /apps/visa -uel
```

In this example, event1 is triggered when any file name in the /apps/discover/stagingDir directory matches the pattern *.6011.*. This match causes the Batch Controller to issue the command /apps/discover -uel, which runs handler1, the Discover UEL Handler program. Based on the parameters shown here, the Batch Controller starts no more than six concurrent instances of this program during the specified period of low expected load, or three during the high-load period.

**Timed Execution**

The Batch Controller provides two time-based scheduling options: metronomic execution and scheduled execution.

**Metronomic Execution**

To set up metronomic execution:

1. In the `batch.timed.events` parameter, specify the event identifiers. If you have two or more event identifiers, separate each with a comma; blank spaces are not allowed.

2. In the `event_name.name` parameter, enter a descriptive label for the event. This name can include spaces or any other characters. It can be of any length, but short names are easier to read.

3. In the `event_name.handlers` parameter, specify identifiers for one or more handlers to trigger when the event occurs. If you have two or more handlers, separate each with a comma; blank spaces are not allowed.
4. (Optional) In the `event_name.start` parameter, specify when you want the first execution to occur, in `hhmmss`, using the 24-hour clock. If you omit this parameter, the first execution occurs immediately after the Batch Controller starts.

5. In the `event_name.interval` parameter, specify the frequency, in seconds, for triggering the associated handler. Failing to specify the interval causes an error.

6. (Optional) In the `event_name.count` parameter, specify how many times to execute this batch handler. If you do not specify this limit, batch handlers run repeatedly at the fixed interval for as long as the Batch Controller is running.

---

**Caution:** You must specify `batch.timed.events` or `batch.random.events` or both. Specifying neither causes the Batch Controller to shut down just after it starts because it has no tasks to perform.

This example demonstrates the metronomic parameters:

```
batch.timed.events  event4
  . .
event4.name  Hourly tally
event4.handlers  handler4
event4.start  000000
event4.interval  3600
event4.count  4
```

In this example, the occurrence specified as `event4` is named **Hourly tally**. It runs `handler4` for the first time at midnight (`000000`), and then runs it again every hour (`3600` seconds) after that until it has run four times. If it cannot run at a scheduled time because previous executions are not finished, it runs again immediately as soon as possible. For example, consider this timeline in Figure 8–1 for `event4`, above:

**Figure 8–1  Hourly Tally Run 1 Exceeds 1 Hour**

In this example, the first run of `handler4` continues for more than an hour, taking it past the time when the second run is supposed to begin. The second scheduled run cannot start at the one-hour interval, so it starts as soon as possible after that (1:20 a.m.). The third and fourth scheduled executions start at regular multiples of the interval, measured from the first run.

If the overly long run continues past two scheduled run start times (occurrences), only one run starts on the delayed basis. For example, suppose the midnight run lasts until 2:25 a.m., as shown in Figure 8–2:
Figure 8–2  Hourly Tally Run1 Exceeds 2 Hours

In this case, the run scheduled for 2:00 begins immediately at 2:25. The fourth run, scheduled for 3:00 begins on time. The second run, scheduled for 1:00 is skipped.

Scheduled Execution

To set up scheduled execution:

1. In the batch.timed.events parameter, specify the event identifiers. If you have two or more event identifiers, separate each with a comma; blank spaces are not allowed.

2. In the event_name.name parameter, enter a descriptive label for the event. This name can include spaces or any other characters. It can be of any length, but short names are easier to read.

3. In the event_name.handlers parameter, specify identifiers for one or more handlers that are to be triggered when the event occurs. If you have two or more handlers, separate each with a comma; blank spaces are not allowed.

4. In the event_name.at parameter, specify each time when you want execution to occur, in hhmmss, using a 24-hour clock.

5. In the event_name.file.location parameter, specify the full path name of the directory to monitor for the arrival of new files that match the pattern in event_name.file.pattern.

6. In the event_name.file.pattern parameter, specify the file name pattern to look for. You can use an asterisk (*) to represent zero or more characters in the file name. No other wild cards (metacharacters) are supported.

Important: Depending on your configuration, the file pattern might conflict with a file pattern used by another component, such as Rated Event Loader. To prevent conflicts, use a specific pattern, for example, test*.out.

You can also check the Pipeline Manager registry to see if any temporary file output patterns conflict with this entry.

This example demonstrates the schedule parameters:

```
batch.timed.events  event5

. . .

event5.name  Sporadic tally
event5.handlers  handler5
event5.at  004500, 022500, 045500
event5.file.location  /apps/discover/stagingDir
event5.file.pattern  *.6011.*

. . .

handler5.name  Sporadic Handler
```

Controlling Batch Operations  8-7
In this example, the program specified as `event5` is named **Sporadic tally**. It runs only at 12:45 a.m., 2:25 a.m., and 4:55 a.m. If a program cannot run at a scheduled time because previous executions are not finished, it runs as soon as possible.

### Starting the Batch Controller

Use this command to start the Batch Controller:

```plaintext
start_batch_controller
```

### About SampleHandler

BRM software includes a generic batch handler, called SampleHandler, which you can use with any batch application that processes data from files. The Batch Controller can call this batch handler whenever a specified directory receives a file whose name matches a specified pattern. The input and output files are then moved to directories that you specify.

Preparing SampleHandler for use involves:

1. **Copying SampleHandler**
2. **Customizing SampleHandler**
3. **Configuring the Batch Controller**
4. **Starting the New Batch Handler**

If SampleHandler does not meet your needs, you can write your own batch handler, as described in “Writing Custom Batch Handlers” in **BRM Developer’s Guide**.

### Copying SampleHandler

The directory `BRM_Home/apps/sample_handler` contains these files:

- **pin.conf**: the batch handler’s configuration file for BRM-related parameters.
- **sample_template.xm**: used by Universal Event (UE) Loader only.
- **SampleHandler.pl**: the actual code of the batch handler.
- **SampleHandler_config.values**: the batch handler’s configuration file for application-specific parameters.
- **SampleReload.pl**: used by UE Loader only.

Copy the entire directory and name the copy appropriately. For example, if your new handler is for the Widget application, then you might name the new directory `BRM_Home/apps/<widget_handler>`.

In the new directory, you can rename the **SampleHandler_config.values** file to include the application’s name, such as `<widget_handler>_config.values`. If you do so, you must also edit the **SampleHandler.pl** file to change **SampleHandler_config.values** to the new name.

### Customizing SampleHandler

To configure the new batch handler for the desired application:
1. Open the pin.conf file for the batch handler (BRM_Home/apps/<widget_handler>/pin.conf, for example).

2. Edit the BRM connection parameters. For information, see “Using Configuration Files to Connect and Configure Components”.

3. Save and close the batch handler’s pin.conf file.

4. Open the .values file for the batch handler (BRM_Home/apps/<widget_handler>/SampleHandler_config.values, unless you have renamed the file).

5. Ensure that the $HANDLER_DIR entry specifies the full path to the directory containing the batch application’s log, input, output, and other files.

6. Edit the $FILETYPE entry to specify the file name pattern to look for.

   The batch handler retrieves all files with this file name pattern from the specified directory.

   **Important:** The file name pattern must have the .bc file extension. The Batch Controller automatically appends .bc to each file name before it runs a batch handler.

   **Tip:** You can use an asterisk (*) to represent zero or more characters in the file name. No other wild cards (metacharacters) are supported.

7. (Optional) To change the batch handler’s log file to a directory other than $HANDLER_DIR, edit the $LOGFILE entry to specify the full path to the desired directory.

8. Edit the $pinUEL entry to specify the name of the application to run.

   Ensure that the $pinUELDir entry specifies the full path to the directory containing the application to run.

9. (Optional) To configure the batch handler to get input files from a directory other than $HANDLER_DIR, edit the $STAGING entry to specify the full path to the desired directory.

   The batch handler will move input files from the $STAGING directory to the $PROCESSING directory, where the application will read them.

10. (Optional) To configure the application to get input files from a directory other than $pinUELDir, edit the $PROCESSING entry to specify the full path to the desired directory. This must be the same directory that is specified as the application’s input directory.

   The batch handler will move input files from the $PROCESSING directory to the $ARCHIVE or $REJECT directory, depending on the application’s exit code. Successfully processed files go into the $ARCHIVE directory, and files with problems go into the $REJECT directory.

11. (Optional) To store the application’s processed files somewhere other than $HANDLER_DIR, edit the $ARCHIVE and $REJECT entries to specify the full paths to the desired directories.

12. Save and close the batch handler’s .values file.
Configuring the Batch Controller

You must identify your new batch handler to the Batch Controller. Edit the **Infranet.properties** file of the Batch Controller, as described in "Handler Identification".

Starting the New Batch Handler

As with any batch handler, you start this one by starting or restarting the Batch Controller. The Batch Controller monitors the newly specified file location for the arrival of files and, when a file appears, starts the new batch handler. For more information, see "About the Batch Controller".

Before using the new batch handler for your production system, you should try it on a test system.
This document describes Oracle Communications Billing and Revenue Management (BRM) connection pool functionality.

Overview

A connection pool is a set of connections maintained between an application, such as Content SDK, and the Connection Manager (CM). An incoming request is assigned a connection from this pool and uses the connection to perform operations. When the operation completes, the connection is returned to the pool.

If an incoming request cannot be assigned a connection immediately, the request is queued. The request waits for a connection to become available for a configurable period of time. If a connection doesn’t become available during this time, an exception is thrown indicating that the request timed out.

Connection pooling includes these features:

- Automatic connection pool resizing
- Automatic removal and replacement of bad connections
- Automatic connection attempt retries (failover)
- Timeout management

Configuring the Connection Pool

You configure the connection pool by using attribute/value pairs in the application’s Infranet.properties file.

Table 9–1 describes the configurable connection pool parameters in the Infranet.properties file.

---

**Note:** For the location of the Infranet.properties file, descriptions of any application-specific parameters, and parameter default values, see the appropriate application documentation, such as "Using Content SDK" in BRM Content Manager.

---
Connection Pool Error Handling

Connection pool handles errors by throwing determinate exceptions. These exceptions, derived from the `DeterminateException` class, specify causes for connection failure as shown in Table 9–2:

### Table 9–2  Connection Pool Error Handling

<table>
<thead>
<tr>
<th>Reason for Exception Thrown</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONNECT_PARSE_ERROR</td>
<td>Thrown when parsing the property file name-value pairs results in errors.</td>
</tr>
<tr>
<td>CONNECT_TIMED_OUT</td>
<td>Thrown when a connection cannot be returned from a pool due to request-timeout.</td>
</tr>
<tr>
<td>CONNECTION_COULD_NOT_BE_ESTABLISHED</td>
<td>Thrown when there are no connections available and the pool is expanding in size. The additional connections had problems establishing connections to BRM.</td>
</tr>
<tr>
<td>CONNECTION_NO_LOGIN_INFO</td>
<td>Thrown when the connections created cannot log on to BRM.</td>
</tr>
<tr>
<td>CREATE_ERROR</td>
<td>Thrown by any <code>createInstance</code> APIs if the static instance is already created.</td>
</tr>
<tr>
<td>WAIT_ERROR</td>
<td>Thrown when a connection is in the request queue.</td>
</tr>
</tbody>
</table>

**Tip:** To find the reason for the exception, use this line in your code:

```java
get toString()
```
Monitoring Connection Pooling Events

When a request is assigned a connection or a connection request times out, the connection pool messages are recorded in the log file.

**Note:** If an exception is thrown, the calling code is responsible for issuing a new connection request.
This chapter provides reference information for Oracle Communications Billing and Revenue Management (BRM) system administration utilities.
load_pin_event_record_map

Use this utility to load the event record file (BRM_Home/sys/data/config/pin_event_record_map) into the Oracle Communications Billing and Revenue Management (BRM) database.

The event record file contains a list of event types to exclude from being recorded in the database. For more information, see "Managing database usage".

---

**Note:** You cannot load separate /config/event_record_map objects for each brand. All brands use the same object.

---

**Important:** At the time you load the event record file, if any events of a type specified in the file already exist in the database, these events remain in the database.

---

After running this utility, you must stop and restart the Connection Manager (CM). See "Starting and Stopping the BRM System".

**Location**

BRM_Home/bin

**Syntax**

load_pin_event_record_map [-d] [-v] [-r] pin_event_record_map

**Parameters**

- **-d**
  Log debugging information.

- **-v**
  Verbose. Displays detailed information as the event record map is created.

- **-r**
  Returns a list of the events in the pin_event_record_map file configured to not be recorded.

  **Important:** This option must be used by itself and does not require the file name.

**pin_event_record_map**

The file containing the event types to exclude from the database.

Specify to not record the event type by setting its flag value to 0 in the file. To temporarily record an event type, change the event’s flag value to 1 and reload the file.

This example shows the event record file format where the event type is followed by its flag value:

/event/session : 1
Results

The `load_pin_event_record_map` utility notifies you only if it encounters errors. Look in the `default.pinlog` file for errors. This file is either in the directory from which the utility was started or in a directory specified in the utility configuration file.
**partition_utils**

Use this utility to customize and maintain BRM database partitions.

---

**Note:**
- You can run this utility against only Oracle databases.
- To use this utility to add a partition for a non-event class, partitioning must have been enabled for that class before `pin_setup` was run during the BRM installation process. For more information, see "Editing the pin_setup.values File to Enable Partitioning for Non-Event Tables" in *BRM Installation Guide*.

Before you use the `partition_utils` utility, configure the database connection parameters in the `partition_utils.values` file in `BRM_Home/apps/partition_utils`. See "Configuring a database connection".

**Important:** After you start the utility, do not interrupt it. It might take several minutes to complete, depending on the size of your database.

For more information, see the following topics:
- Partitioning database tables
- About purging data

**Location**

`BRM_Home/bin`

**Syntax for Adding Partitions**

```
partition_utils -o add -t realtime|delayed -s start_date
-u month|week|day -q quantity
[-c storable_class] [-w width] [-f] [-p] [-h]
```

**Parameters for Adding Partitions**

- `-o add`
  Adds partitions.

- `-t realtime|delayed`
  Adds real-time or delayed-event partitions. Only event tables may be set to `delayed`.

  **Note:** Conversion Manager does not support the migration of data to the EVENT_T tables.

- `-s start_date`
  Specifies the starting date for the new partitions. The format is `MMDDYYYY`.  

**start_date** must be the day after tomorrow or later; you cannot create partitions starting on the current day or the next day. For example, if the current date is January 1, the earliest start date for the new partition is January 3 (For example, 01032007.)

- **-u month|weekday**
  Specifies the time unit for the partitions.

- **-q quantity**
  Specifies the number of partitions to add.

  If a partition with a future date already exists in the table, running this utility for adding partitions adds partitions more than the specified quantity.

  For example, you want to create one partition with starting date as February 1 and the table already contains the P_R_02282009 partition. The P_R_02282009 partition is split into two partitions called P_R_02012009 and P_R_02282009.

- **-c storable_class**
  Specifies the class of objects to be stored in the partition. The default storable base class for partition_utils is the /event base class.

- **-w width**
  Specifies the number of units in a partition; for example, 3.

- **-f**
  Forces the creation of partitions even if **start date** is included in the current partition. If this parameter is used on an existing partition, the existing partition is split into two, one containing the old data and the other including the start date given in the command.

---

**Caution:** Before forcing partitions:

For real-time partitions, stop all BRM processes.

For delayed-event partitions, stop all delayed-event loading by stopping Pipeline Manager.

See "Starting and Stopping the BRM System".

---

**Note:** The **-f** parameter works differently when you remove partitions. In that case, it forces the removal of objects associated with open items.

---

- **-p**
  Writes an SQL statement of the operation to the partition_utils.log file but does not perform any action on the database. See "Running the partition_utils utility in test mode".

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

### Syntax for Purging Partitions

```
partition_utils -o purge -e end_date [-t realtime|delayed] [-p] [-h]
```
-s start_date
Specifies the start of the date range for the objects you want to purge. The format is MMDDYYYY.

If the specified dates don’t match the partition boundaries, only objects in partitions that are completely within the date range are purged.

Parameters for Purging Partitions

-0 purge
Purges event objects without removing partitions.

-e end_date
Specifies the cutoff date for keeping objects (on closed items). Objects that are older than that date and associated with closed items are purged; no objects associated with active items are purged.

-t realtime|delayed
Purges real-time or delayed-event partitions. The default is to purge both real-time and delayed-event partitions.

-p
Writes a report of purgeable objects to the partition_utils.log file but does not perform any action on the database. See "Running the partition_utils utility in test mode".

-h
Displays the syntax and parameters for this utility.

Syntax for Removing Partitions

```
partition_utils -o remove -s start_date -e end_date [-c storable_class] [-t realtime|delayed] [-f] [-p] [-h]
```

Parameters for Removing Partitions

-0 remove
Removes partitions.

-s start_date
Specifies the start of the date range for the objects you want to remove. The format is MMDDYYYY.

If the specified dates do not match the partition boundaries, only objects in partitions that are completely within the date range are removed. See "About purging event objects".

-e end_date
Specifies the end of the date range for the objects you want to remove. The format is MMDDYYYY.

By default, you can use this operation to remove only those partitions that are older than 45 days. You can change this limitation by editing the partition_utils.values file in BRM_Home/apps/partition_utils. See "Customizing partition limitations".

-c storable_class
Specifies the partition you are removing by base class. The default is /event.
To remove a non-event table partition, you must use the `-f` parameter. Operations using this option cannot be undone and will remove objects that are being used. Use with caution.

**-t realtime|delayed**
Removes real-time or delayed partitions. The default is to remove both real-time and delayed-event partitions.

**-f**
Forces the removal of partitions that contain objects that are associated with an open item. By default, these partitions aren’t removed.

**Caution:** Use this option carefully.

**Note:** The `-f` parameter works differently when you add partitions. In that case, it forces the splitting of partitions even when they fall within the date range of the current partition.

**-p**
Writes an SQL statement of the operation to the `partition_utils.log` file but does not perform any action on the database. See “Running the partition_utils utility in test mode”.

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

### Syntax for Enabling Delayed Partitions

```
partition_utils -o enable -t delayed -c storable_class [-p] [-h]
```

### Parameters for Enabling Delayed Partitions

**-o enable -t delayed**
Enables delayed-event partitions.

**-c storable_class**
Use the `-c` (class name) parameter to specify the event classes for which you want partitioning.

You can enable partitioning for all subclasses of an event by using the percent sign (%) as a wildcard.

**-p**
Writes an SQL statement of the operation to the `partition_utils.log` file but does not perform any action on the database. See “Running the partition_utils utility in test mode”.

**-h**
Displays the utility’s syntax and parameters.
Syntax for Updating Partitions

```
partition_utils -o update [-c storable_class] [-p] [-h]
```

Parameters for Updating Partitions

- **-o update**
  Aligns partitions across all object tables for a single base class. All real-time and delayed-event partitions get the same real-time partitioning scheme as their base table (EVENT_T for /event base class tables, ITEM_T for /item base class tables, etc.).

- **-c storable_class**
  Specifies the class of objects to be updated. The default storable base class is /event.

- **-p**
  Writes an SQL statement of the operation to the partition_utils.log file, but does not perform any action on the database. See "Running the partition_utils utility in test mode".

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

Syntax for Restarting a partition_utils Job

```
partition_utils -o restart [-b] [-h]
```

Parameters for Restarting a partition_utils Job

- **-o restart**
  Re-executes the last operation that was unsuccessful due to an error or abnormal termination.

- **-b**
  Bypasses executing the last operation but cleans the status of it.

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

Syntax for Finding the Maximum POID for a Date

```
partition_utils -o maxpoid -s date -t realtime|delayed [-p] [-h]
```

Parameters for Finding the Maximum POID for a Date

- **-o maxpoid**
  Returns the maximum POID for the given date.

- **-s date**
  Specifies the date that determines the maximum POID. The format is MMDDYYYY.

- **-t realtime|delayed**
  Gets the maximum POID in only real-time or only delayed partitions.

- **-p**
  Writes an SQL statement of the operation to the partition_utils.log file but does not perform any action on the database. See "Running the partition_utils utility in test mode".
-h
Displays the syntax and parameters for this utility.

Results
If the utility doesn’t notify you that it was successful, look in the partition_utils.log file to find any errors. This file is either in the directory from which the utility was started, or in a directory specified in the utility configuration file. The partition_utils.log file includes SQL statements if you use the -p parameter.
Use the `partitioning.pl` script to manage all the nonpartitioning-to-partitioning upgrade tasks. Run it from a UNIX prompt.

For more information, see "Changing from a nonpartitioned to a partitioned database".

---

**Important:** The `partitioning.pl` script needs the `partition.cfg` configuration file in the directory from which you run the utility.

---

**Location**

`BRM_Home/bin`

**Syntax**

```
perl partitioning.pl [-c | -n | -a | -h ]
```

**Parameters**

- `-c`

Creates the database objects required for the upgrade, including the following:

- The UPG_LOG_T table that logs all the information about the upgrade.
- The `pin_upg_common` package that contains all the common routines for the upgrade.

- `-n`

Displays the event tables that will be partitioned during the upgrade.

Tables selected for partitioning are listed in the `TABLES_TOBE_PARTITIONED_T` table, which is created during the upgrade process. This table contains two columns:

- `table_name`: The name of the table to be partitioned.
- `partition_exchanged`: The value of the exchanged partition. This value is used by the upgrade scripts to perform the table partitioning.

For example:

```
tables_tobe_partitioned
table_name       varchar2(30)
partition_exchanged number(38)
```

Use the `INSERT` statement to partition tables and use `0` for the `partition_exchanged` value. For example, to insert `MY_CUSTOM_EVENT_TABLE`, execute this SQL statement:

```
INSERT INTO TABLES_TOBE_PARTITIONED_T (table_name, partition_exchanged) VALUES ('MY_CUSTOM_EVENT_TABLE',0); COMMIT;
```

**Note:** To prevent a listed table from being partitioned, use the SQL `DELETE` statement to delete its name from `TABLES_TOBE_PARTITIONED_T`.
-a
Partitions the tables listed in the TABLES_TOBE_PARTITIONED_T table.

To partition additional event tables, insert their names into TABLES_TOBE_PARTITIONED_T.

-h
Displays the syntax and parameters for this utility.

Results

The utility doesn’t notify you if it was successful. Look in the UPG_LOG_T table to find any errors.
pin_clean_asos

Use the pin_clean_asos utility to delete closed /active_session objects from Oracle In-Memory Database (IMDB) Cache.

This utility compares an object’s expiration time to the current time to determine if the object needs to be deleted and deletes the objects from one IMDB Cache at a time.

Note: The default pin.conf file for this utility includes the entry -pin_mta multi_db. This utility does not use this entry.

In a multischema environment, you must run this utility separately for each IMDB Cache node. You can create a script that calls the utility with connection parameters to connect to the desired node.

For more information about deleting expired objects from IMDB Cache, see "Purging Old Call Data from Memory" in BRM Telco Integration.

Important: To connect to the BRM database, the pin_clean_asos utility needs a pin.conf configuration file in the directory from which you run the utility.

Location

BRM_Home/bin

Syntax

pin_clean_asos -object "object_type" [-expiration_time number_of_hours] [-help]

Parameters

-object "object_type"
Specifies the type of /active_session object to delete. For example, to delete /active_session/telco/gsm objects from IMDB Cache:

pin_clean_asos -object "/active_session/telco/gsm"

-expiration_time number_of_hours
Sets the expiration time (in hours) for /active_session objects. The utility compares the expiration time with an object’s end time (PIN_FLD_END_T) and deletes objects that are older than the number of hours you specify. For example, if you specify 2 as the value and you run the utility at 10 a.m., the utility deletes objects that were closed on or before 8 a.m.

If not specified, the expiration time defaults to 0. This results in the removal of all /active_session objects that have an end time less than the current time.

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

If the utility does not notify you that it was successful, look in the default.pinlog file to find and errors. This file is either in the directory from which the utility was started or in a directory specified in the utility configuration file.
Use this utility to free space in Oracle In-Memory Database (IMDB) Cache or the database by deleting objects that remain because of a session’s abnormal termination, such as a missed stop accounting or cancel authorization request.

This utility performs the following functions:

- Releases expired reservations.
- Rates active sessions that are in a started or an updated state and updates balances in the database.
- Deletes expired active session objects.

This utility compares an object’s expiration time with the current time to determine if the object needs to be deleted and deletes objects from one IMDB Cache at a time.

Note: This utility requires a `pin.conf` file to provide entries for connecting to the Connection Manager (CM) and DM, login name and password, log file information, and performance tuning. The values for these entries are obtained from the information you provided during installation, or the entries are set with a default value.

The default `pin.conf` file for this utility includes the entry `-pin_mta multi_db`. This utility does not use this entry. In a multischema environment, you must run this utility separately for each IMDB Cache node. You can create a script to call the utility with connection parameters to connect to the desired node.

### Location

`BRM_Home/bin`

### Syntax

```
pin_clean_rsvns [-help] [-object object_type] [-account]
    [-expiration_time number_of_hours] [-cause user_defined]
    [-bytes_uplink volume] [-bytes_downlink volume]
```

### Parameters

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

Note: This utility is installed with Resource Reservation Manager. See "About Resource Reservation Manager" in *BRM Configuring and Collecting Payments.*
-**object** _object_type_

Specifies the object storage location. In an IMDB Cache DM environment, _object_type_ can only be 1. Use `-object 1` to delete expired objects in IMDB Cache.

-**account**

Calls `STOPACCOUNTING` to delete the active session objects in both created and update states, and starts rating the session.

Use this parameter to rate active sessions if your network supports only the create state, and not start and update states, for sessions. For example, networks using the Message Based Interface (MBI) protocol send only authorization and reauthorization requests, so the active session objects remain in a created state even during usage.

---

**Note:** When you run the utility without this parameter, it calls `STOPACCOUNTING` to delete the active session objects in the update state.

---

-**expiration_time** _number_of_hours_

Sets back the expiration time for the objects, in hours.

The default is the current time when the utility runs. The utility deletes objects that are in an expired state up to the time you specify. For example, if you specify 2 as the value and you run the utility at 10 a.m., the utility deletes objects that expired by 8 a.m.

-**cause** _user_defined_

Specifies the reason for releasing reservations and terminating the session.

You can define any value for the cause, and the value is stored in the event object. You can define different rate plans depending on the reason for the session termination.

---

**Note:** You use the rate plan selector to specify the rate plan for calculating the charges for the event.

---

-**bytes_uplink** _volume_

When the `/active_session` object’s `PINFld_BYTES_UPLINK` field is set to 0 or not specified, this parameter specifies to populate the field with the specified volume.

-**bytes_downlink** _volume_

When the `/active_session` object’s `PINFld_BYTES_DOWNLINK` field is set to 0 or not specified, this parameter specifies to populate the field with the specified volume.

---

**Results**

The utility releases any expired reservations and deletes any expired session objects. If there are any session objects in a started or updated state, the utility rates the objects:

- For duration RUMs, it calculates the duration to rate by using the end time specified in the session object, the expiration time in the reservation object, the `pin_virtual_time`, or the current system time.

- For volume RUMs, it calculates the volume to rate by using the volume in the session object or the volume passed in at the command line.
Use this utility to start and stop Oracle Communications Billing and Revenue Management (BRM) components.

Before you read this document, you should be familiar with how BRM works. See "BRM System Architecture" in *BRM Concepts*.

---

**Important:** To connect to the BRM database and configure the different processes, the `pin_ctl` utility needs a configuration file in the directory from which you run the utility. This configuration file must be called `pin_ctl.conf`, which is different from most BRM configuration file names.

For information on setting up and configuring the processes that `pin_ctl` controls, see "Configuring the pin_ctl Utility".

For general information on creating configuration files, see "Creating Configuration Files for BRM Utilities".

---

**Syntax**

```
pin_ctl action component [-c file_name] [-collectdata] [-debug] [-i]
```

**Parameters**

*action*

Specifies the type of action to be executed. See "action Parameter".

For example, to start the Connection Manager (CM), use this command:

```
pin_ctl start cm
```

* -c file_name

Specifies a configuration file to use instead of the default. Use this parameter to run different configurations of the same system.

*component*

Specifies the process on which the action is performed. See "component Parameter".

* -collectdata

Gets diagnostic data when starting, stopping, or checking the status of a component.

* -debug

Displays debugging information.

* -i

Interaction mode. Use this parameter to allow the utility to stop and ask if you want to proceed. This is especially useful when running `stop`, `halt`, and `clear`.

---

**action Parameter**

* clear

Deletes log entries associated with the component (not the file).
cstart
C clears the component logs and, if the component isn’t running, starts the component. If the component is already running, the command clears the log file; the component continues running.

Note: You are not prompted to clear logs.

halt
Searches for the specified component and runs the `kill -9` command.

restart
Stops the component, waits for completion, then restarts the component.

snmpget action
Gets an SNMP value. To use this command, you must add `snmpget` actions by editing the `pin_ctl.conf` file. See "Customizing snmpset and snmpget Actions".

snmpset action
Sets an SNMP value:
- `rebalanceAaaGw`. Rebalances AAA Gateway Manager CMs. See "Rebalancing CM Connections" in BRM AAA Gateway Manager.
- `addServerInstance`. Rebalances DM connection load when you restart a failed DM in the DM pool or when you add a new DM to the DM pool.
- `refreshConnections`. Rebalances DM connections when you restart a failed DM in the pool.

You can add `snmpset` actions by editing the `pin_ctl.conf` file. See "Customizing snmpset and snmpget Actions".

start
Starts the component if it is not running. If you specify `all for component`, it starts the components specified in the `pin_ctl.conf` file. For information, see "Customizing the components included in “all”".

status
Returns the status `Running` or `NotRunning` of component.

stop
Starts the component if it is running. If you specify `all for component`, it stops the components specified in the `pin_ctl.conf` file. For information, see "Customizing the Components Included in ‘all’".

component Parameter
You can perform an action on any of the following components:
all
Applies the action to a customizable set of components. By default, the components are:

- Oracle Data Manager (DM)
- Email DM
- Connection Manager
- CM Master Process
- Invoice formatter

To customize the set of components, see "Customizing the components included in "all"."

aaa
AAA Gateway Manager

answer
Paymentech answer simulator

batch_controller
Batch Controller

bre
Pipeline Manager

cm
Connection Manager

cm_proxy
CM Proxy

cmmpp
Connection Manager Master Process

dm_eai
Enterprise Application Interface (EAI) DM

dm_email
Email DM

dm_fusa
Paymentech DM

dm_ifw_sync
Account Synchronization DM (Oracle AQ)

dm_invoice
Invoice DM

dm ldap
LDAP DM

dm oracle
Oracle DM

dm tt
Oracle In-Memory Database (IMDB) Cache DM
**dm_vertex**  
Vertex tax calculation DM

**eai_js**  
EAI Java Server

**formatter**  
Invoice formatter

**infmgr**  
System Manager

**nmgr**  
Node Manager

**rtp**  
Real-time pipeline
Use this utility to monitor the following database key performance indicators (KPIs) in Oracle databases:

- Age of event and audit tables
- Size of audit tables
- Invalid or missing procedures, triggers, or indexes

You configure the `pin_db_alert.pl` utility to alert you when one of these components has returned a certain status.

For more information, see "Using the pin_db_alert utility to monitor key performance indicators".

---

**Important:** To connect to the BRM database, the `pin_db_alert.pl` utility needs a configuration file in the directory from which you run the utility. See "Creating Configuration Files for BRM Utilities".

---

**Location**

`BRM_Home/diagnostics/pin_db_alert`

**Syntax**

`pin_db_alert.pl`

**Parameters**

This utility has no parameters.
Use the `pin_purge` utility to delete old bills, items, journals, and expired account sub-balances from Oracle IMDB Cache.

This utility compares an object’s expiration time to the current time to determine if the object needs to be deleted.

This utility deletes objects from one IMDB cache or logical partition at a time. If the system has multiple logical partitions, you need to run this utility for each logical partition. In a high availability configuration, you must run the utility for each high-availability node. You can create a script to call the utility with relevant connection parameters to connect to the desired IMDB cache nodes.

**Important:** Before running this utility, unset the ORACLE_HOME environment variable.

**Note:** This utility requires a `pin.conf` file in the directory from which you run the utility with entries for connecting to the Connection Manager (CM) and IMDB Cache DM, login name and password to connect to the data store, batch size for deleting, and log file information. The values for these entries are obtained from the information you provided during installation or the entries are set with a default value. The `pin.conf` file for this utility is installed in the `BRM_Home/apps/pin_subscription` directory.

### Location

`BRM_Home/bin`

### Syntax

```
pin_purge [-l username/password@DatabaseAlias] -c {bill | item | subbalance} (-n number_of_days | -d MM/DD/YYYY) [-help]
```

### Parameters

- **-l username/password@DatabaseAlias**
  Specifies how to connect to the database. If you omit this option, the utility uses the information provided in the `pin.conf` file to establish the connection.

- **-c {bill | item | subbalance}**
  Specifies the object to be deleted.
  - `-c bill` deletes bills and all related items and journals that do not have any pending or open items, due amounts set to zero, and bill status set to closed (bill in finalized)

  Deleting bill objects does not also delete the associated bill items. The items must be deleted separately using the `-c item` parameter.

  **Note:** The bill objects are deleted from the Oracle IMDB Cache only.
-c **item** deletes billable and special items (such as payments, adjustments, and disputes) in closed status and all journals related to these items.

**Note:** The item and journal objects are deleted from Oracle IMDB Cache *only*.

- **c subbalance** deletes account sub-balances.

**Note:** Account sub-balances are deleted from Oracle IMDB Cache. The objects are eventually deleted from the BRM database when updates from Oracle IMDB Cache are propagated to the BRM database.

- **n number_of_days**
  Specifies the number of days prior to which bills, items, or sub-balances are deleted. For example, specify **90** to delete bills, items, or sub-balances older than 90 days.

- **d MM/DD/YYYY**
  Specifies the date prior to which bills, items, or sub-balances are deleted. For example, specify **03/01/2009** to delete bills, items, or sub-balances older than March 1, 2009.

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

**Results**

If the utility does not notify you that it was successful, look in the pin_purge.pinlog file to find any errors. This file is either in the directory from which the utility was started or in a directory specified in the utility configuration file.
pin_sub_balance_cleanup

Use this utility to purge expired account sub-balances from the Oracle Communications Billing and Revenue Management (BRM) database.

Caution: When you delete sub-balances from the database, events that impacted those sub-balances cannot be rerated. Make sure you no longer need the expired sub-balances before deleting them.

For more information, see "About purging account sub-balances".

To connect to the BRM database, the pin_sub_balance_cleanup utility needs a configuration file in the directory from which you run the utility. See "Creating Configuration Files for BRM Utilities".

Important: You must run the pin_sub_balance_cleanup utility from the BRM_Home/apps/pin_subscription directory. This directory contains the pin.conf file that has the parameters required for this utility.

Location

BRM_Home/bin

Syntax

pin_sub_balance_cleanup -n | -d

Parameters

-n
The number of days prior to which sub-balances are deleted. For example, specify 60 to delete expired sub-balances older than 60 days.

-d
The date prior to which sub-balances are deleted. Use the format MM/DD/YYYY. For example, specify 06/30/2003 to delete expired sub-balances older than June 30, 2003.

Results

The pin_sub_balance_cleanup utility doesn’t notify you that it was successful. To check for errors, look in the utility log file (BRM_Home/apps/pin_subscription/pin_subscription.pinlog).
Use the pin_tt_schema_gen utility to connect to the BRM database and generate SQL scripts to create and initialize the BRM cache groups in Oracle In-Memory Database (IMDB) Cache. These SQL scripts contain all the required cache group definitions. You can run these SQL scripts against the appropriate IMDB Cache nodes to load the cache groups with the required schema and data. See "Creating and Initializing the BRM Cache Group Schema".

Note: The pin_tt_schema_gen utility has been certified for Perl 5.8.0 and Oracle 11g.

Before running this utility, you must configure the database connection parameters in one of the following files in BRM_Home/bin.

- pin_tt_schema_gen.values, which generates scripts for the default cache groups.
- A custom configuration values file, which generates scripts for custom cache groups.

See "Configuring the pin_tt_schema_gen.values File" for more information.

Important: After you start this utility, do not interrupt it. It might take several minutes to complete, depending on the size of your database.

Note: When you run this utility, the following warning messages are logged in the pin_schema_gen.log file. These warnings are reported for storable classes that do not have associated tables. You can ignore these warning messages.

'/reservation/active' mentioned in array local_tables_class_def does not have any table.

'/active_session/telco/gprs/master' mentioned in array local_tables_class_def does not have any table.

'/active_session/telco/gprs/subsession' mentioned in array local_tables_class_def does not have any table.

Requirements

The pin_tt_schema_gen utility requires the following versions of the Perl modules:

- DBI version 1.605
- DBD-Oracle version 1.16
- Bit-Vector version 7.1

Location

BRM_Home/bin
Syntax

```
```

Parameters

- `-t` Generates the `tt_schema.sql` script file, which you can run against the appropriate IMDB Cache node to create the cache groups and transient table schema.

- `-l` Generates the `tt_load.sql` script file, which you can run against the appropriate IMDB Cache node to load data from the BRM database.

- `-d` Generates the `tt_drop.sql` script file, which you can run against the appropriate IMDB Cache node to drop the cache groups.

- `-o` Updates the BRM database with unique indexes and not-null constraints.

- `-a` Runs the `-t`, `-l`, `-d`, and `-o` parameters.

- `-h` Displays the syntax and parameters for this utility.

- `-f configuration_values_files` Specifies the name of the configuration values files to be used by the utility. The default is `pin_tt_schema_gen.values`. You can provide another configuration values file; for example `my_custom_schema_gen.values`.

Results

If the utility does not notify you that the files were generated, look in the `pin_tt_schema_gen.log` file to find any errors. This file is either in the directory from which the utility was started or in a directory specified in the utility configuration file.
pin_virtual_gen

Use the pin_virtual_gen utility to convert /event classes in the BRM schema to use virtual columns. After you run the utility, the poid_type columns of event tables in the BRM database are virtual-column enabled.

For more information, see "Generating Virtual Columns on Event Tables".

To connect to the BRM database and to specify logging information, the pin_virtual_gen utility uses the Infranet.properties file in the directory from which you run the utility.

Specify the log level by setting the infranet.log.level property in the Infranet.properties file. The default value is 1. Valid values are 1, 2, and 3. Regardless of the log level set, status messages are printed to stdout and to the log file. Errors are logged and printed to stderr.

Location

BRM_Home/apps/pin_virtual_columns

Syntax

pin_virtual_gen { { -gentasks [-execute] | -readtasks [-execute] } { [create|pre_export|post_export|verify_types|create_types] } | -showtasks [minTaskId maxTaskId] }

Parameters

-gentasks create [-execute]
Generates tasks and stores them in the database.
Executes tasks after saving to the database.

-readtasks create [-execute]
Reads previously stored tasks from the database.
Executes tasks after reading from the database.

-gentasks create
Creates virtual columns and supporting columns in the BRM database.
Use this option in conjunction with the showtasks option if you want to display the tasks that will be executed for creating the virtual columns before executing them. This example shows how to create the tasks for creating virtual columns, display them, and then execute them:

pin_virtual_gen -gentasks create
pin_virtual_gen -showtasks
pin_virtual_gen -readtasks create -execute

-showtasks minTaskId maxTaskId
Reads corresponding tasks from the database and displays task details.

minTaskId and maxTaskId specify the tasks to show within an ID range. The command shows tasks that have an ID greater than minTaskId or lower than maxTaskId. minTaskId and maxTaskId are optional.
All tasks are displayed when an ID range is not provided.
-gentasks pre_export [-execute]
Removes virtual columns temporarily.
```
pin_virtual_gen -gentasks pre_export -execute
```

-gentasks post_export [-execute]
Restores virtual columns that were temporarily removed.
```
pin_virtual_gen -gentasks post_export -execute
```

-gentasks verify_types [-execute]
Verifies whether storable class type names exist in the data dictionary of the BRM database schema.
```
pin_virtual_gen -gentasks verify_types -execute
```

-gentasks create_types [-execute]
Creates the names of custom storable class types and stores them in the data dictionary of the BRM database schema.
```
pin_virtual_gen -gentasks create_types -execute
```

-help
Displays the syntax (usage) for this utility.

Results

The **pin_virtual_gen** utility notifies you when it runs successfully. Otherwise, look in the `vcol.pinlog` file for errors. This file is either in the directory from which the utility was started or in a directory specified in the utility configuration file (`Infranet.properties`).
purge_audit_tables.pl

Use this Perl script to archive unneeded shadow objects in audit tables. Use it to:

- Generate audit table reports
- Create history tables
- Move audit tables to history tables
- Archive audit tables
- Restore archived audit tables

The different versions of shadow objects that are valid for archiving are moved to the history tables so they can be accessed for future reference if necessary. In addition, when versions of an object are removed from audit tables, all subclasses of the shadow objects are also removed automatically by the script.

Note: The purge_audit_tables.pl script doesn’t delete objects from the database; it only purges the object rows stored in a table.

Important: To connect to the BRM database, the purge_audit_tables.pl script needs a configuration file in the directory from which you run the utility. See "Creating Configuration Files for BRM Utilities".

For more information, see "Archiving Audit Data" in BRM Developer’s Guide.

For information on audit trails and shadow objects, see "About Tracking Changes to Object Fields" in BRM Developer’s Guide.

Location

BRM_Home/sys/archive/oracle

Syntax

purge_audit_tables.pl report -t objects -d date -l login/pswd@connection
create -t objects -l login/pswd@connection
archivedirect -t objects -d date -c commit_size -l login/pswd@connection
archiveindirect -t objects -d date -c commit_size -l login/pswd@connection
renametohist -t objects -l login/pswd@connection
updffromhist -t objects -d date -c commit_size -l login/pswd@connection
help

The following purging actions are supported:

- report Syntax
- create Syntax
- archivedirect Syntax
- archiveindirect Syntax
- renametohist Syntax
- updfromhist Syntax

**report Syntax**

Generates audit table reports.

The utility generates a file named `purge_tables.report`, which provides information about the tables for the specified objects, including the number of rows in each table that are eligible for purging, and whether history tables exist for them. You create a report to determine which mode of archiving to use for the specified object: **archivedirect** or **archiveindirect**.

```
purge_audit_tables.pl report -t objects -d date -l login/pswd@connection
```

- **-t objects**
  Specifies a comma-separated list of shadow objects on which to report.

  Shadow objects use an **au** prefix. For example, a change to a field marked for auditing in the `/profile` object results in the `/au_profile` shadow object.

  **Note:** Do not specify child objects for an object; they are included automatically by the script. For example, the `/au_profile/serv_extracting` object is reported on when you list the `/au_profile` object.

- **-d date**
  Specifies the cutoff date for purging data.

  This date determines which versions of the audit object are eligible for purging. If a version of an object is valid at the cutoff date, and there is at least one older version of the same object, the valid object is kept and all older versions are marked for purging and moved to the history tables.

  The format is **YYYY:MM:DD**.

- **-l login/pswd@connection**
  Specifies your standard Pipeline Manager user name and database password. This parameter is required to connect to the database.

  **Sample report command**

  ```
  perl purge_audit_tables.pl report -t au_account -d 2005:02:23 -l pin/p1N@subdb
  ```

- create Syntax

  Creates empty history tables for the specified objects and their child objects.

  ```
purge_audit_tables.pl create -t objects -l login/pswd@connection
```

- **-t objects**
  Specifies a comma-separated list of objects for which to create history tables. History tables are prepended by **H_** as shown in Table 10–1.

  **Table 10–1  Audit and History Tables Format**

<table>
<thead>
<tr>
<th>/service object audit tables</th>
<th>/service object history tables</th>
</tr>
</thead>
<tbody>
<tr>
<td>AU_SERVICE_T</td>
<td>H_SERVICE_T</td>
</tr>
<tr>
<td>AU_SERVICE_ALIAS_T</td>
<td>H_SERVICE_ALIAS_T</td>
</tr>
<tr>
<td>AU_SERVICE_EXTRACTING_T</td>
<td>H_SERVICE_EXTRACTING_T</td>
</tr>
</tbody>
</table>
-l login/pswd@connection
Specifies your standard Pipeline Manager user name and database password. This parameter is required to connect to the database.

Sample create command
purge_audit_tables.pl create -t au_account -l pin/p1N@subdb

archivedirect Syntax
Archives audit tables for the specified objects and their child objects by copying the data directly to the history tables and then removing it from the audit tables.

purge_audit_tables.pl archivedirect -t objects -d date -c commit_size -l login/pswd@connection

-t objects
Specifies a comma-separated list of objects to archive audit tables for.

Shadow objects use an au prefix. For example, a change to a field marked for auditing in the /profile object results in the /au_profile shadow object.

Important: Do not specify the child objects for a table; they are handled automatically by the script.

-d date
Specifies the cutoff date for purging data.
This date determines which versions of the audit object are eligible for purging. If a version of an object is valid at the cutoff date, and there is at least one older version of the same object, the valid object is kept and all older versions are marked for purging and moved to the history tables.
The format is YYYY:MM:DD.

-c commit_size
Specifies the number of rows to save to the database at one time.

Note: Do not specify child objects for an object; they are included automatically by the script. For example, the /au_profile/serv_extracting object is reported on when you list the /au_profile object.

Sample archivedirect command
purge_audit_tables.pl archivedirect -t au_account -d 2005:03:29 -c 1000 -l pin/p1N@subdb

archiveindirect Syntax
Archives audit tables for the specified objects and their child objects by copying the data first to temporary tables, then to the history tables. If successful, the old audit table data is removed.

Important: Do not specify the child objects for a table; they are handled automatically by the script.

Important: Do not specify the child objects for a table; they are handled automatically by the script.

Note: Do not specify child objects for an object; they are included automatically by the script. For example, the /au_profile/serv_extracting object is reported on when you list the /au_profile object.
purge_audit_tables.pl archiveindirect -t objects -d date -c commit_size -l login/pswd@connection

- **t objects**
  Specifies a comma-separated list of objects to archive audit tables for.

  Shadow objects use an au prefix. For example, a change to a field marked for auditing in the /profile object results in the /au_profile shadow object.

- **d date**
  Specifies the cutoff date for purging data.

  This date determines which versions of the audit object are eligible for purging. If a version of an object is valid at the cutoff date, and there is at least one older version of the same object, the valid object is kept and all older versions are marked for purging and moved to the history tables.

  The format is YYYY:MM:DD.

- **c commit_size**
  Specifies the number of rows to save to the database at one time.

- **l login/pswd@connection**
  Specifies your standard Pipeline Manager user name and database password. This parameter is required to connect to the database.

  Displays the syntax and parameters for this utility.

  This shows a sample archiveindirect command:

  ```
purge_audit_tables.pl archiveindirect -t au_account -d 2005:02:23 -c 1000 -l pin/pl1N@subDb
  ```

**renametohist Syntax**

Renames the specified audit tables to their corresponding history tables and recreates the audit tables without any indexes. This option also creates the script files used to create, rename, rebuild, and drop indexes that were in the audit tables. You can run these scripts manually when necessary.

- `- create_index_script.sql`
- `- rename_index_script.sql`
- `- rebuild_index_script.sql`
- `- drop_index_script.sql`

```
purge_audit_tables.pl renametohist -t objects -l login/pswd@connection
```
**-t objects**
Specifies a comma-separated list of objects for which to rename audit tables to history tables and recreate empty audit tables.

---

**Note:** You do not need to specify child objects for an object; they are included automatically by the script. For example, the /au_profile/serv_extracting child object is reported on if you list the /au_profile object.

---

**-l login/pswd@connection**
Specifies your standard Pipeline Manager user name and database password. This parameter is required to connect to the database.

This shows a sample **renametohist** command:

```
purge_audit_tables.pl renametohist -t au_account -l pin/p1N@subdb
```

### updfromhist Syntax

Retrieves the data for a given object and its child objects from the history tables and transfers it back to the audit tables.

```
purge_audit_tables.pl updfromhist -t objects -d date -c commit_size -l login/pswd@connection
```

**-t objects**
Specifies a comma-separated list of shadow objects to retrieve the data from the history tables and update in the corresponding audit tables.

Shadow objects use an au prefix. For example, a change to a field marked for auditing in the /profile object results in the /au_profile shadow object.

---

**Note:** Do not specify child objects for an object; they are included automatically by the script. For example, the /au_profile/serv_extracting object is reported on when you list the /au_profile object.

---

**-d date**
Specifies the cutoff date for retrieving data.

The format is YYYY:MM:DD.

**-c commit_size**
Specifies the number of rows to save to the database at one time.

**-l login/pswd@connection**
Specifies your standard Pipeline Manager user name and database password. This parameter is required to connect to the database.

This shows a sample **updfromhist** command:

```
purge_audit_tables.pl updfromhist -t au_account -d 2005:03:29 -c 1000 -l pin/p1N@subdb
```

### help Syntax

Displays the syntax for the **purge_audit_tables.pl** utility.

```
help syntax
purge_audit_tables.pl help
```
Results

The `purge_audit_tables.pl` utility notifies you only if it encounters errors.
This document describes the syntax and parameters for the AGENT++ SNMP utilities that are installed with Oracle Communications Billing and Revenue Management (BRM).

**Note:** The `snmpTrap` and `snmpInform` utilities are not supported.

**Important:** Because the AGENT++ SNMP implementation uses dynamic OIDs, you cannot use symbolic SNMP names in SNMP commands. For example, instead of using this command:

```
snmpWalk <hostname> 10.196.129.31
portal.components.mtf.connectionConfigurations.dmoTable.dmoEntry
-P20761 -S
```

You must use this:

```
snmpWalk <hostname> 10.196.129.31 1.3.6.1.4.1.3512.1.5.2.2.1 -P20761 -S
```

For more information about SNMP commands and options, see the SNMP documentation provided by your OS vendor.

See also "Sample SNMP input and output".
snmpBulk

Retrieves a subtree of OIDs. Uses the SNMPv2 GETBULK request to retrieve information from an SNMPv2C agent. If the node is an SNMPv1-only agent, this utility automatically turns the GETBULK request into an SNMPv1-supported GETNEXT request.

Syntax

snmpBulk IP_address | DNS_name [OID1 [OID2...]] [option1 option2...]

Parameters

IP_Address | DNS_Name
The location of the SNMP process.

OID
Specifies which part of the MIB is searched. All objects in the subtree below the given OID are queried.
The default is sysDescr (description of the host on which the SNMP agent is running).

options
Can be any of the following, separated by a space:

- -vN
  Use SNMP version 1, 2, or 3. The default is 1.
- -Pport
  Remote port to use.
- -Ccommunity_name
  Specifies the community. The default is public.
- -rN
  Specifies the number of retries to be used in the requests. The default is 1.
- -tN
  Specifies the timeout in hundredths of a second between retries. The default is 100.
- -nN
  Specifies non-repeaters (number of object instances that should be retrieved no more than once from the beginning of the request. The default is 0.
- -mN
  The maximum number of times that other variables beyond those specified by the non-repeaters field should be retrieved). The default is 1.
snmpDiscover

Broadcasts a network discovery request to find out if the SNMP master agent is running.

**Syntax**

```
snmpDiscover Broadcast_IP_address [option1 [option2...]]
```

**Parameters**

- **Broadcast_IP_address**
  - IP address of the network.

- **options**
  - Can be any of the following, separated by a space:
    - `-vN`
      - Use SNMP version 1, 2, or 3. The default is 1.
    - `-Pport`
      - Remote port to use.
    - `-Ccommunity_name`
      - Specifies the community. The default is public.
    - `-rN`
      - Specifies the number of retries to be used in the requests. The default is 1.
snmpGet

Uses the SNMP GET request to query for information on a network entity.

Syntax

\[ \text{snmpGet } IP\_address \mid DNS\_name \ [OID] \ [option1 \ [option2\ldots]] \]

Parameters

**IP\_Address | DNS\_Name**

The location of the SNMP process.

**OID**

Specifies which part of the MIB is searched. All objects in the subtree below the given OID are queried.

The default is `sysDescr` (description of the host on which the SNMP agent is running).

**options**

Can be any of the following, separated by a space:

- `-vN`
  
  Use SNMP version 1, 2, or 3. The default is 1.

- `-Pport`
  
  Remote port to use.

- `-Ccommunity_name`
  
  Specifies the community. The default is `public`.

- `-rN`
  
  Specifies the number of retries to be used in the requests. The default is 1.

- `-tN`
  
  Specifies the timeout in hundredths of a second between retries. The default is 100.
snmpNext

Gets data about the next object in the MIB. This utility sends an SNMP GETNEXT request to the master agent and waits for a response back before it proceeds.

Syntax

```
snmpNext  IP_address | DNS_name [OID] [option1 [option2...]]
```

Parameters

**IP_address | DNS_name**
The location of the SNMP process.

**OID**
Specifies which part of the MIB is searched. All objects in the subtree below the given OID are queried.

The default is `sysDescr` (description of the host on which the SNMP agent is running).

**options**
Can be any of the following, separated by a space:

- `-vN`
  Use SNMP version 1, 2, or 3. The default is 1.
- `-Pport`
  Remote port to use.
- `-Ccommunity_name`
  Specifies the community. The default is `public`.
- `-rN`
  Specifies the number of retries to be used in the requests. The default is 1.
- `-tN`
  Specifies the timeout in hundredths of a second between retries. The default is 100.
snmpNextAsync

Gets data about the next object in the MIB. This utility sends an SNMP GETNEXT request to the master agent and receives the response from a callback function. The main process does not have to wait for the response before it proceeds.

Syntax

```
snmpNextAsync  IP_address | DNS_name [OID] [option1 [option2...]]
```

Parameters

**IP_address | DNS_name**
The location of the SNMP process.

**OID**
Specifies which part of the MIB is searched. All variables in the subtree below the given OID are queried.

The default is `sysDescr` (description of the host on which the SNMP agent is running).

**options**
Can be any of the following, separated by a space:

- **-vN**
  Use SNMP version 1, 2, or 3. The default is 1.

- **-P**port
  Remote port to use.

- **-C**community_name
  Specifies the community. The default is public.

- **-rN**
  Specifies the number of retries to be used in the requests. The default is 1.

- **-tN**
  Specifies the timeout in hundredths of a second between retries. The default is 100.
snmpPasswd

Sets a password.

Syntax

```
snmpPasswd  IP_address | DNS_name  user  new_password  [option1  [option2...]]
```

Parameters

**IP_address | DNS_name**
The location of the SNMP process.

**user**
The user name.

**new_password**
The new password.

**options**
Can be any of the following, separated by a space:

- **-vN**
  Use SNMP version 1, 2 or 3. The default is 1.

- **-Pport**
  Remote port to use.

- **-Ccommunity_name**
  Specifies the community. The default is `public`.

- **-rN**
  Specifies the number of retries to be used in the requests. The default is 1.

- **-tN**
  Specifies the timeout in hundredths of a second between retries. The default is 100.
snmpSet

Sets a value in an object.

**Syntax**

```
snmpSet IP_address | DNS_name [OID] [option1 [option2...]]
```

**Parameters**

- **IP_address | DNS_name**
  The location of the SNMP process.

- **OID**
  Specifies which part of the MIB is searched. All objects in the subtree below the given OID are queried.
  The default is `sysDescr` (description of the host on which the SNMP agent is running).

- **options**
  Can be any of the following, separated by a space:
  - `-vN`
    Use SNMP version 1, 2, or 3. The default is 1.
  - `-Pport`
    Remote port to use.
  - `-Ccommunity_name`
    Specifies the community. The default is `public`.
  - `-Gcommunity_name`
    Specifies the GET community. The default is the SET community value.
  - `-rN`
    Specifies the number of retries to be used in the requests. The default is 1.
  - `-tN`
    Specifies the timeout in hundredths of a second between retries. The default is 100.
**snmpWalk**

Retrieves a subtree of OIDs.

**Syntax**

```
snmpWalk IP_address | DNS_name [OID] [option1 [option2...]]
```

**Parameters**

- **IP_address | DNS_name**
  The location of the SNMP process.

- **OID**
  Specifies which part of the MIB is searched. All objects in the subtree below the given OID are queried.
  The default is `sysDescr` (description of the host on which the SNMP agent is running).

- **options**
  Can be any of the following, separated by a space:
  - **-vN**
    Use SNMP version 1, 2, or 3. The default is 1.
  - **-Pport**
    Remote port to use.
  - **-S**
    Walk only within the subtree.
  - **-Ccommunity_name**
    Specifies the community. The default is public.
  - **-rN**
    Specifies the number of retries to be used in the requests. The default is 1.
  - **-tN**
    Specifies the timeout in hundredths of a second between retries. The default is 100.
snmpWalkThreads

Retrieves a subtree of OIDs. The start OID is 1.

This utility gets all OID values from one or multiple SNMP master agents (one master agent per thread when processing the SNMP WALK request). `snmpWalk` gets the values on a subtree of OIDs from one SNMP master agent.

**Syntax**

```
snmpWalkThreads host/port [host/port ...] [option1 [option2...]]
```

**Parameters**

*host/port*
Host name and port of the system you are getting information about.

*options*
Can be any of the following, separated by a space:

- `-vN`
  Use SNMP version 1, 2, or 3. The default is 1.

- `-Pport`
  Remote port to use.

- `-Ccommunity_name`
  Specifies the community. The default is `public`.

- `-rN`
  Specifies the number of retries to be used in the requests. The default is 1.

- `-tN`
  Specifies the timeout in hundredths of a second between retries. The default is 100.
Part II describes how to configure an Oracle Communications Business and Revenue Management (BRM) high-availability system. It contains the following chapters:

- Understanding a High-Availability System
- Configuring a High-Availability System
Understanding a High-Availability System

This document provides an overview of the Oracle Communications Billing and Revenue Management (BRM) high-availability architecture. In addition, it explains how the BRM components in a high-availability system handle failover.

About High-Availability BRM Systems

A high-availability system is designed to continue functioning when one or more of its components fail. To do this, it contains backup components to which it automatically switches (fails over) when an active component stops working. The failover should appear seamless to users and should not interrupt service.

In a BRM system, high availability is required primarily for real-time processing of prepaid and postpaid services and for operations performed by customer service representatives (CSRs). Batch processes, such as invoicing and billing, typically do not require high availability because they do not involve real-time interaction with users. You can handle failure in batch processing by restarting the component or by accepting slower performance rather than by switching to a backup component. Therefore, this document covers high-availability systems for real-time processing only.

About the Architecture of a High-Availability BRM System

A high-availability BRM system contains one or more backup components for each of the following components:

- AAA Gateway Manager
- Connection Manager (CM)
- In-Memory Database Cache Data Manager (IMDB Cache DM)
- Oracle IMDB Cache (the data store)
- Oracle Clusterware (includes a built-in backup mechanism)
- Oracle Real Application Clusters (Oracle RAC) instance

The primary and backup components should not run on the same physical host or use the same power supply.

Each instance—primary and backup—of a component should be connected to all instances of the component’s server-side peer (the component that it calls). For example, each instance of AAA Gateway Manager should be connected to each CM, each CM should be connected to each IMDB Cache DM, and so on. This ensures that if one instance of a component fails, another one is available to process data.
In addition to backups, the level of availability of the components in a BRM system depends on the sizing that you use (that is, the number of computers in the system). The more computers that you spread your components across, the fewer components that are affected if one of the computers fails.

Figure 12–1 shows the architecture of the basic set of components required to create a high-availability BRM system. Dashed lines represent backup connections.

**Figure 12–1 Basic High-Availability BRM System**

Each component in a high-availability system starts the failover process in the following situations:

- A network connection to the underlying host server is lost.
- A response to a request is not received within the timeout period specified for the component that sent the request. The delay can be due to hardware, software, or network problems.

The following sections explain how each component in a high-availability BRM system handles failover:

- About AAA Gateway Manager in a High-Availability System
- About Connection Managers in a High-Availability System
- About IMDB Cache DMs and Data Stores in a High-Availability System
About AAA Gateway Manager in a High-Availability System

The following sections explain how failures related to AAA Gateway Manager are handled in a high-availability system:

- How the Network Handles AAA Gateway Manager Failure
- How AAA Gateway Manager Handles CM Failure

For an overview of AAA Gateway Manager, see BRM AAA Gateway Manager.

For information about configuring AAA Gateway Manager for a high-availability system, see "Configuring a High-Availability System".

How the Network Handles AAA Gateway Manager Failure

In a high-availability system containing two or more instances of AAA Gateway Manager, the network service control point (SCP) should respond to gateway failure in one of the following ways:

- If the SCP is configured to send requests to all instances of AAA Gateway Manager in the system, it distributes the workload handled by the failed gateway to the remaining available gateways.
- If the SCP is configured to send requests only to the primary instance of AAA Gateway Manager, it switches to the designated backup gateway when the primary gateway fails.

**Note:** The SCP is not part of the BRM system. To configure the SCP, see the SCP documentation.

How AAA Gateway Manager Handles CM Failure

In a high-availability system, AAA Gateway Manager handles the following types of CM failure:

- Connection Failure
- Request Processing Failure

**Connection Failure**

If any CM fails, AAA Gateway Manager fails over to the next available CM in its CM connection pool.

For information about CM connection pools, see the discussion of connection pools in BRM AAA Gateway Manager.

**Request Processing Failure**

If AAA Gateway Manager does not receive a response from a CM within the SocketTimeout timeout period, it starts the failover process to another CM in its connection pool.
**About Connection Managers in a High-Availability System**

In a high-availability system, the CMs detect failures in the pipelines and in the IMDB Cache DMs to which they connect. This section describes how CMs handle such failures:

- **How CMs Handle Real-Time Pipeline Failure**
- **How CMs Handle IMDB Cache DM Failure**

For information about configuring CMs for a high-availability system, see "Understanding a High-Availability System".

### How CMs Handle Real-Time Pipeline Failure

When a CM receives a request for rating that requires discounts to be applied or zoning to be considered by Pipeline Manager, the CM forwards the request to the instance of Pipeline Manager running real-time discounting and zoning pipelines. If the Pipeline Manager instance fails and does not send a response to the CM, the request times out, and the CM sends an error message to AAA Gateway Manager. AAA Gateway Manager considers the CM failed and starts the failover process.

If a request has no discounting or zoning requirements, the CM sends the request to the IMDB Cache DM.

### How CMs Handle IMDB Cache DM Failure

In a high-availability system, the failure of an IMDB Cache DM is detected by a CM. CMs handle the following types of IMDB Cache DM failure:

- **Initial Connection Failure**
- **Session Failure**

For an overview of the IMDB Cache DM, see the discussion of IMDB Cache DM in *BRM Concepts*

For information about IMDB Cache DMs in a high-availability system, see "About IMDB Cache DMs and Data Stores in a High-Availability System".

### Initial Connection Failure

In a high-availability system, each CM connects to an active and a standby IMDB Cache DM. If the active DM does not respond when the CM tries to establish a connection to it, the CM performs the following actions:

1. Checks the `dm_pointer` entries in the CM `pin.conf` file for the host name and port number of the standby IMDB Cache DM and connects to the standby DM.
2. Logs a failover message in the CM log file.
3. Initiates the IMDB Cache DM failover process by sending a login request to the standby DM with the `PCM_OPFLG_RETRY` flag.
   
   The standby IMDB Cache DM accepts the login request and starts the failover process only if its associated Oracle IMDB Cache (data store) is active.
   
   Otherwise, it rejects the login request with the `PIN_ERR_NOT_ACTIVE` error and forces the CM to retry the formerly active DM. (The `PIN_ERR_NOT_ACTIVE` error is recorded as `PIN_ERR_NOT_PRIMARY` in the CM log file.)
4. If the standby IMDB Cache DM rejects the CM’s login request, the CM tries to connect to the formerly active IMDB Cache DM at intervals specified in the `pcm_`
bad_connection_retry_delay_time_in_secs entry in the CM pin.conf and IMDB Cache DM pin.conf files.

**Session Failure**
If a CM loses its connection to the active IMDB Cache DM during a session or if a request from the CM to the active DM times out, the CM performs the following actions:

1. Closes the connection to the active IMDB Cache DM.
2. Logs a failure message in the CM log file.
3. Checks the dm_pointer entries in its pin.conf file for the name and port number of the standby IMDB Cache DM.
4. Initiates the IMDB Cache DM failover process by sending a login request to the standby DM with the PCM_OPFLG_RETRY flag.
   - If the failed entry opcode is not in an explicit transaction, the CM retries the opcode with the PCM_OPFLG_RETRY flag.
     Explicit transaction means that AAA Gateway Manager sent a transaction open request to the CM before retrying the failed entry opcode.
   - If the failed entry opcode is in an explicit transaction, the CM returns the PIN_ERRCLASS_SYSTEM_RETRYABLE error to AAA Gateway Manager.
     AAA Gateway Manager ends the transaction and then retries the same transaction.
     The standby DM accepts the request and starts the failover process only if the Oracle IMDB Cache (data store) that it connects to is active.
     Otherwise, it rejects the request with the PIN_ERR_NOT_ACTIVE error and forces the CM to resend its request to the formerly active DM. (The PIN_ERR_NOT_ACTIVE error is recorded as PIN_ERR_NOT_PRIMARY in the CM log file.)
5. If the standby DM rejects the request, the CM tries to reconnect to the formerly active DM and resume the interrupted session.

**About IMDB Cache DMs and Data Stores in a High-Availability System**

The basic BRM high-availability architecture has one pair of active and standby IMDB Cache DM instances and one pair of active and standby Oracle IMDB Cache instances for each BRM database schema. Larger high-availability systems can have several IMDB Cache DMs and Oracle IMDB Cache pairs for each schema.

---

**Note:** An Oracle IMDB Cache instance is also called a data store

*Figure 12–2 shows a typical relationship between the IMDB Cache DMs and data stores in a high-availability system. The dashed line represents a backup connection.*
As the preceding figure illustrates, the active IMDB Cache DM is connected to the active data store and the standby IMDB Cache DM is connected to the standby data store. Both IMDB Cache DM instances are also directly connected to the Oracle RAC-enabled BRM database. The replication agent associated with the active data store propagates the updates to the standby data store. Updates in the standby data store are propagated to the BRM database.

When an IMDB Cache DM starts up, it connects to its associated data store and checks the state of that data store. If the data store is active, the IMDB Cache DM sets its own processing state to active. If the data store is on standby, the IMDB Cache DM sets its processing state to standby.

The active IMDB Cache DM processes all requests, and only the data in the active data store is updated. The standby data store receives the updates from the active data store and propagates them to the BRM database.

The standby IMDB Cache DM processes requests only when the state of its data store changes to active.

In a high-availability system, the data store ensures data availability and integrity.

**How IMDB Cache DMs Fail Over**

An IMDB Cache DM failure can occur in the following situations:

- The node on which the active IMDB Cache DM and its data store reside fails. See "About Active Node Failure".
- The IMDB Cache DM associated with the active data store fails. See "About Active IMDB Cache DM Failure".

**About Active Node Failure**

In a high-availability system, an IMDB Cache DM instance and its associated data store reside on the same physical server (node).

When an active node fails, the failover process is as follows:

1. Oracle Clusterware detects the active data store failure and does the following:
About IMDB Cache DMs and Data Stores in a High-Availability System

Changes the state of the standby data store to active.

Changes the state of the formerly active data store to standby.

2. The CM sends requests to the standby IMDB Cache DM instance.

3. The standby IMDB Cache DM checks the state of its associated data store.
   - If the data store is active, the standby DM changes its own processing state to active and processes the request.
   - If the data store is on standby or if the standby DM’s connection to the data store fails, the standby DM rejects the CM request with the PIN_ERR_NOT_ACTIVE error. (The error is recorded as PIN_ERR_NOT_PRIMARY in the CM log file.)

About Active IMDB Cache DM Failure

IMDB Cache DM failover is managed by Oracle Clusterware. When Oracle Clusterware detects an IMDB Cache DM failure, the failover process is as follows:

1. Oracle Clusterware tries to restart the formerly active DM.

2. While Oracle Clusterware is trying to restart the DM, the CM continues trying to connect to the DM at intervals for the duration of the specified retry time period (pcm_bad_connection_rety_delay_time_in_secs).
   - If the CM cannot connect within the specified time period, it redirects requests to the standby IMDB Cache DM.

3. If Oracle Clusterware is able to restart the formerly active DM, the DM checks the state of its associated data store.
   - If the data store is active, the DM sets its processing state to active and starts processing CM requests.
   - If the data store is on standby, the IMDB Cache DM sets its processing state to standby.

4. If Oracle Clusterware is unable to restart the formerly active DM, you must manually activate the standby DM.

Manually Activating a Standby IMDB Cache DM

When an active IMDB Cache DM fails, its associated data store is not notified of the failure, so the data store’s status remains active. This prevents the standby data store from becoming active.

Because its associated data store is still on standby, the standby DM rejects all CM requests with the PIN_ERR_NOT_ACTIVE error to indicate that it is in standby mode and not accepting requests. (The PIN_ERR_NOT_ACTIVE error is recorded as PIN_ERR_NOT_PRIMARY in the CM log file.)

Therefore, if an internal IMDB Cache DM error prevents Oracle Clusterware from restarting a DM, you must manually change the standby data store’s state to active. This enables the standby DM to switch its state to active and process the requests redirected to it by the CM.

All CM requests will fail until either the active or standby IMDB Cache DM establishes a connection with an active data store.
About IMDB Cache DMs and Data Stores in a High-Availability System

How IMDB Cache DMs Handle Data Store Failure

When the active IMDB Cache DM receives a request from the CM, it passes the request to its associated data store only if that data store’s state is active.

The data store is considered to have failed in the following situations:

- When the initial IMDB Cache DM connection to the data store fails. See "Initial Connection Failure".
- When the connection to the data store is lost during a transaction or a request has timed out. See "Transaction Failure".

Initial Connection Failure

When the active IMDB Cache DM’s attempt to connect to its data store fails, the following actions occur:

1. The DM logs a connection failure message in the IMDB Cache DM log file.
2. The DM retries the connection to the data store.
3. If the retry connection fails, the DM rejects the CM request and logs a PIN_ERR_STORAGE error.
4. Oracle Clusterware detects the active data store failure and does the following:
   - Changes the state of the standby data store to active.
   - Changes the state of the formerly active data store to standby.
5. The CM sends the request to the standby IMDB Cache DM.
6. The standby DM checks the state of its associated data store.
   - If the data store is active, the standby DM sets its processing state to active and processes the request.
   - If the data store is on standby or if the connection to the data store fails, the standby DM rejects the CM request with the PIN_ERR_NOT_ACTIVE error. (The error is recorded as PIN_ERR_NOT_PRIMARY in the CM log file.)
   - If the standby DM’s connection to its data store fails, the CM tries to reconnect to the active IMDB Cache DM.

Transaction Failure

If the active IMDB Cache DM loses its connection with its data store during a transaction or if a request from the active DM to its data store times out, the following actions occur:

1. The DM logs a connection failure message in the IMDB Cache DM log file.
2. The DM retries the connection to its data store.
3. If the connection fails, the DM sets its processing state to standby and rejects the CM request with the PIN_ERR_NOT_ACTIVE error.
4. The CM sends the request to the original standby IMDB Cache DM.
5. Oracle Clusterware detects the active data store failure and does the following:
   - Changes the state of the original standby data store to active.
   - Changes the state of the formerly active data store to standby.
6. The original standby DM checks the state of its associated data store.
If the data store is active, the original standby DM sets its processing state to active and processes the request.

If the data store is on standby or if the connection to the data store fails, the original standby DM rejects the CM request with the PIN_ERR_NOT_ACTIVE error. (The error is recorded as PIN_ERR_NOT_PRIMARY in the CM log file.)

**How IMDB Cache DMs Handle Oracle RAC Failure**

In a high-availability system, each IMDB Cache DM maintains connections to a primary Oracle RAC node and to a backup Oracle RAC node through connect descriptors in the `tnsnames.ora` file referenced by the DM. The database service associated with the connect descriptor specifies which Oracle RAC node is primary and which is the backup for that D.

In Figure 12–3, solid lines represent primary connections and dashed lines represent backup connections.

![IMDB Cache DM Connections to Oracle RAC Nodes](image)

In normal processing, each IMDB Cache DM sends requests to its primary Oracle RAC instance. If that Oracle RAC instance becomes unavailable, the IMDB Cache DM connects to its backup Oracle RAC instance. This section describes how IMDB Cache DMs handle Oracle RAC failure.

In a high-availability system, IMDB Cache DMs handle the following types of Oracle RAC failure:
Initial Connection Failure
Connection failures can be due to hardware, software, network, or listener problems or to a lack of response from the Oracle RAC instance. If an IMDB Cache DM’s initial attempt to connect to its primary Oracle RAC instance fails, the IMDB Cache DM connects to its backup Oracle RAC instance.

Session Failure
When an Oracle RAC instance fails while processing a request, the IMDB Cache DM detects the failure and connects to its backup Oracle RAC instance. Any transaction that is active when the failure occurs is rolled back. Information about the failover is logged in the IMDB Cache DM pinlog file.

If an IMDB Cache DM loses the connection to its primary Oracle RAC instance in the middle of a session, it performs the following actions:

1. Tries to reestablish the connection to its primary Oracle RAC instance.
2. Performs one of the following actions:
   - If the reconnection attempt is successful, continues processing the request.
   - If the reconnection attempt is unsuccessful, clears any incomplete connection to the primary Oracle RAC instance and tries to connect to its backup Oracle RAC instance.

In the following situations, the IMDB Cache DM returns a PIN_ERRCLASS_SYSTEM_RETRYABLE error and PIN_ERR_STORAGE error code to the CM so the client can retry the transaction:

   —The failover occurs in the middle of a transaction.
   —The failover occurs outside a transaction, and the IMDB Cache DM cannot finish the operation.

If the failover happens outside a transaction and the IMDB Cache DM can finish the operation, the failover is transparent to the CM.

About the BRM Database in a High-Availability System
This section describes the database components of a high-availability system:

- About Oracle Real Application Clusters and Oracle Clusterware
- About Multischema High-Availability Systems

For an overview of multischema systems, see this discussion about multischema systems in BRM Concepts.

About Oracle Real Application Clusters and Oracle Clusterware
For a high-availability system, you must use Oracle Real Application Clusters (Oracle RAC), which consists of multiple Oracle RAC instances. Each Oracle RAC instance has the following characteristics:

- Runs on its own cluster node and server
- Is typically associated with only one schema
Concurrently processes data for a single database with all the other Oracle RAC instances in the cluster.

Oracle RAC requires a highly available, high-speed storage system. A storage area network (SAN) running on clustered hardware is recommended. The cluster nodes are connected through a high-performance grid.

Oracle Clusterware is used to manage Oracle RAC servers. It also facilitates state management of Oracle IMDB Cache instances (data stores) and manages the failover of IMDB Cache DM instances by restarting the IMDB Cache DM process when it detects a failure.

For information about installing and configuring Oracle RAC instances and Oracle Clusterware, see the Oracle RAC and Oracle Clusterware documentation.

About Multischema High-Availability Systems

BRM supports both multischema and multidatabase systems. For high-availability systems, however, a multischema database must be used.

Multischema high-availability systems are built on an Oracle RAC system with one primary Oracle RAC instance for every schema in your system plus at least one backup Oracle RAC instance. The backup Oracle RAC instance can take over for any primary Oracle RAC instance that fails. Each IMDB Cache DM is connected to a primary Oracle RAC instance and to the backup Oracle RAC instance. Because a primary Oracle RAC instance never handles the load of more than one server, all primary Oracle RAC servers can be sized to run at 80% capacity during peak processing times. This reduces your system’s overall spare idle capacity.

You can increase the number of backup Oracle RAC instances to meet business requirements.

Figure 12–4 shows the configuration for a multischema high-availability system with two schemas. Solid lines represent primary connections and dashed lines represent backup connections.
Figure 12–4 Multischema High-Availability System

Connection Key
Primary
Backup
This document provides guidelines for configuring an Oracle Communications Billing and Revenue Management (BRM) high-availability system for real-time processing of prepaid and postpaid services.

For an overview of a high-availability BRM system, see "Understanding a High-Availability System".

About Setting Up a High-Availability System

To create a high-availability system, you must install and configure at least two instances of each component and then connect each instance to all instances of the component’s server-side peer.

To create a high-availability BRM system, follow these guidelines:

- Install and configure the following components:
  - A single-schema or multischema BRM database.
  - One Oracle Real Application Clusters (Oracle RAC) instance for each database schema, and at least one additional Oracle RAC instance to use as a backup.
  - Oracle Clusterware.
  - One or more pairs of active and standby Oracle In-Memory Database Cache (Oracle IMDB Cache) instances for each database schema. (Oracle IMDB Cache instances are also called data stores.)
  - One or more pairs of active and standby IMDB Cache Data Managers (DMs) for each database schema.
  - At least two Connection Managers (CMs). Use as many as you estimate will support your workload, and then add one more.
  - One real-time pipeline for each CM.
  - At least two instances of AAA Gateway Manager. Use as many as you estimate will support your workload, and then add one more.
**Important:** Duplicate components should reside on different physical hosts and use different power sources. They should not be located in the same rack. If you use blade servers, install duplicate components in different blade chassis. (This also applies to active and standby Oracle IMDB Cache and IMDB Cache DM instances.)

For more information about configuring components in a high-availability system, see the following sections:

- Configuring the BRM Database for High Availability
- Configuring IMDB Cache Manager for High Availability
- Configuring Connection Managers for High Availability
- Configuring AAA Gateway Manager for High Availability

---

- Connect each instance of a component to all instances of the component’s server-side peer. For example, configure each AAA Gateway Manager to connect to all the CMs, and configure each CM to connect to all the IMDB Cache DMs.

- When setting timeout periods for processing a request, ensure that each client-side (or calling) component has a longer timeout period than the server-side component that it calls.

- The timeout period (also called latency) is the time taken by a component to respond to a request from the moment the request is sent to the component. The *timeout period* (also called latency) is the time taken by a component to respond to a request from the moment the request is sent to the component.

  The timeout period should be long enough to accommodate slow responses due to overload, in which case there is an occasional response rather than no response.

  Table 13–1 lists the timeout settings that affect component failover in a high-availability system. The components are listed in the order that they process requests received from the network.
Configuring the BRM Database for High Availability

For a diagram of the data processing flow in a high-availability system, see "About the Architecture of a High-Availability BRM System".

**Important:** Timeout periods must be large enough to accommodate slow responses because of overload.

### Configuring the BRM Database for High Availability

The BRM database in high-availability systems consists of the following components:

- Oracle RAC. See "About Oracle Real Application Clusters and Oracle Clusterware".
- A single-schema or multishema BRM database. See "About Multishema High-Availability Systems".

The following sections explain how to configure Oracle RAC for failover in a high-availability system:

---

**Table 13–1 Timeout Settings for High Availability Systems**

<table>
<thead>
<tr>
<th>Component</th>
<th>Timeout setting</th>
<th>Description</th>
<th>Suggested value</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAA Gateway Manager</td>
<td>SocketTimeOut</td>
<td>Specifies the amount of time that the gateway waits for a response from the CM before failing over to another CM. See &quot;Configuring AAA Gateway Manager for High Availability&quot;.</td>
<td>120 seconds</td>
</tr>
<tr>
<td>Connection Manager</td>
<td>pcm_timeout_in_msecs</td>
<td>Specifies the amount of time that the CM waits for a response from an IMDB Cache DM before failing over to the next IMDB Cache DM in the dm_pointer list. This entry is known as the CM's long (failover) timeout. See &quot;Connecting CMs to IMDB Cache DMs&quot;.</td>
<td>100 seconds</td>
</tr>
<tr>
<td>Connection Manager</td>
<td>pcm_bad_connection_retry_delay_time_in_secs</td>
<td>Specifies the interval at which the CM tries again to connect to the active IMDB Cache DM after it fails to connect. The timeout should be long enough for the CM to re-establish a connection with the IMDB Cache DM. See &quot;Connecting CMs to IMDB Cache DMs&quot;.</td>
<td>-</td>
</tr>
<tr>
<td>Oracle IMDB Cache</td>
<td>LockWait</td>
<td>Specifies the number of seconds that Oracle IMDB Cache (the data store) waits to acquire a lock for the BRM database before returning a timeout error to the IMDB Cache DM. See &quot;Creating and Configuring Active Data Stores&quot;.</td>
<td>30 seconds</td>
</tr>
<tr>
<td>Oracle RAC database</td>
<td>FAST_START_MTTR_TARGET</td>
<td>Specifies the time limit for Oracle RAC instance recovery. See &quot;Minimizing Recovery Time of Oracle RAC Instances&quot;.</td>
<td>30 seconds</td>
</tr>
</tbody>
</table>
Setting Up Oracle RAC for Failover in a High-Availability System

Figure 13–1 shows the configuration of Oracle RAC in a basic high-availability system. Dashed lines represent backup connections.

To configure Oracle RAC for failover in a high-availability system:

1. Set up an Oracle RAC instance for each database schema in your system and then add at least one more Oracle RAC instance to function as the backup. For example, if you have three schemas, set up at least four Oracle RAC instances.

   For information about how to set up an Oracle RAC system, see the Oracle RAC documentation.
The Oracle RAC instances should reside on different physical hosts and use different power sources.

2. Configure Oracle database services. See "Configuring Oracle Database Services".

3. Add entries for the Oracle database services to all tnsnames.ora files that will be referenced by the IMDB Cache DMs in your system. See "Defining Connections to the Oracle Database Services".

4. Configure your IMDB Cache DMs to connect to the Oracle RAC instances.

Configuring Oracle Database Services

You use Oracle database services to connect IMDB Cache DMs to Oracle RAC instances. To create a high-availability system, you must map each database service to one primary Oracle RAC instance and to one backup Oracle RAC instances.

For example, if your system has four database schemas and five Oracle RAC instances, configure the database services as shown in Table 13–2:

<table>
<thead>
<tr>
<th>Database service</th>
<th>Primary Oracle RAC instance</th>
<th>Backup Oracle RAC instance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service1</td>
<td>Oracle RAC instance 1</td>
<td>Oracle RAC instance 5</td>
</tr>
<tr>
<td>Service2</td>
<td>Oracle RAC instance 2</td>
<td>Oracle RAC instance 5</td>
</tr>
<tr>
<td>Service3</td>
<td>Oracle RAC instance 3</td>
<td>Oracle RAC instance 5</td>
</tr>
<tr>
<td>Service4</td>
<td>Oracle RAC instance 4</td>
<td>Oracle RAC instance 5</td>
</tr>
</tbody>
</table>

To create the services in the preceding table, log on to any Oracle RAC node as the Oracle database administrator, and run the following commands:

```
srvctl add service -d racDatabaseName -s service1 -r racInstanceName1 -a racInstanceName5 -P Basic
srvctl add service -d racDatabaseName -s service2 -r racInstanceName2 -a racInstanceName5 -P Basic
srvctl add service -d racDatabaseName -s service3 -r racInstanceName3 -a racInstanceName5 -P Basic
srvctl add service -d racDatabaseName -s service4 -r racInstanceName4 -a racInstanceName5 -P Basic
```

For information about the `srvctl` command, see the Oracle RAC documentation.
You must also configure each service to be notified if the backup Oracle RAC instance becomes unreachable or network connections fail. To do this, enable Fast Application Notification (FAN) for each database service. For information about FAN, see *Automatic Workload Management with Oracle Real Application Clusters* on the Oracle Technology Network (http://www.oracle.com/technetwork/index.html).

 Optionally, you can enable Transparent Application Failover (TAF) in Basic failover mode for the database services. TAF is not required by IMDB Cache DMs, but it benefits other BRM applications that connect directly to the database, such as Rated Event Loader. In Basic mode, applications connect to a backup Oracle RAC node only after their connection to the primary Oracle RAC node fails. This approach has low overhead, but end users might experience a delay while the new connection is created. For more information about TAF, see the Oracle database documentation.

### Defining Connections to the Oracle Database Services

Perform the following procedure in the appropriate `tnsnames.ora` files.

To define connections to the Oracle database services:

1. Open the `tnsnames.ora` file in a text editor.
   
   By default, that file is in the `Oracle_Home/network/admin/` directory.

2. For each database service, add the following connect descriptor:

   ```
   connectionString =
   (DESCRIPTION =
   (ADDRESS = (PROTOCOL = TCP)(HOST = primaryRacInstanceHostName)(PORT = oraHostPortNo))
   (ADDRESS = (PROTOCOL = TCP)(HOST = backupRacInstanceHostName)(PORT = oraHostPortNo))
   (LOAD_BALANCE = OFF)
   (CONNECT_DATA =
   (SERVER = DEDICATED)
   (SERVICE_NAME = serviceName)
   (FAILOVER_MODE =
   (TYPE = SELECT)
   (METHOD = BASIC)
   (RETRIES = 180)
   (DELAY = 5)
   )
   )
   )
   ```

   where:

   - `connectionString` is the connection name. The `sm_database` entry in the IMDB Cache DM `pin.conf` file must match this entry.
   - `ADDRESS` defines a single listener protocol address. Add an entry for the primary Oracle RAC instance’s listener and the backup Oracle RAC instance’s listener.
   - `primaryRacInstanceHostName` is the name of the computer on which the service’s primary Oracle RAC instance resides.
   - `backupRacInstanceHostName` is the name of the computer on which the service’s backup Oracle RAC instance resides.
   - `oraHostPortNo` is the port number for the Oracle database on the host computer. Typically, this number is 1521.
- **LOAD_BALANCE** specifies whether to distribute connection requests across the listeners specified in the **ADDRESS** entries. For high-availability BRM systems, set this to **OFF**.

- **serviceName** specifies the name of the database service. The **sm_svcname** entry in the IMDB Cache DM **pin.conf** file must match this entry.

- For **FAILOVER MODE**,
  - **TYPE = SELECT** specifies that work in progress is not lost when failover occurs from the primary to the backup Oracle RAC node.
  - **METHOD = BASIC** specifies that applications connect to a backup Oracle RAC node only after their connection to the primary Oracle RAC node fails. Backup connections are not preestablished.

3. Save and close the file.

**Minimizing Recovery Time of Oracle RAC Instances**

To minimize the recovery time of Oracle RAC instances, use the **FAST_START_MTTR_TARGET** initialization parameter to reduce the size of the redo log file. When setting this parameter, you must balance system performance against failure recovery time. Use the following values:

- **0**—Disables the parameter. In this case, blocks containing modified data not yet written to disk (dirty blocks) are flushed mainly during checkpoints triggered by redo log switches, so instance recovery time depends primarily on the amount of redo log data to apply at the time of failure. If the redo logs are huge and the current log is almost full, recovery might take several hours.

- **1 through 3600**—Specifies the time limit in seconds for database instance recovery. To meet this target, the Oracle database adjusts the frequency of checkpoint creation. It might need to proactively flush dirty blocks from the database cache to disks. This requires additional input/output operations, which can degrade performance.

For example:

```sql
alter system set FAST_START_MTTR_TARGET=30;
```

The preceding command sets database instance recovery time to 30 seconds. This value is a recommended starting point, but you should test it in your environment to find the optimal setting.

---

**Important:** The timeout period of all BRM components in a high-availability system should be greater than **FAST_START_MTTR_TARGET**.

---

When **FAST_START_MTTR_TARGET** is set to a short time period, such as 30 seconds, you can further reduce service downtime by lowering the database cluster heartbeat interval. (By default, Oracle RAC waits 30 seconds before resetting a node after the loss of its heartbeat.) A very short heartbeat interval, however, might result in unnecessary node resets due to network blips.

---

**Note:** Failure of one Oracle RAC node interrupts service in all Oracle RAC nodes because Oracle RAC must remaster internal services and restore the database by using the current state of the redo log file.
For more information about redo log files, see "Assigning storage for redo log files" in BRM Installation Guide.

**Configuring IMDB Cache Manager for High Availability**

This section provides information about configuring IMDB Cache Manager for high availability.

The basic BRM high-availability architecture has one pair of *active* and *standby* IMDB Cache DM instances and one pair of *active* and *standby* Oracle IMDB Cache instances (data stores) for each BRM database schema. Larger high-availability systems can have several IMDB Cache DMs and Oracle IMDB Cache pairs for each schema.

For more information, see "About IMDB Cache DMs and Data Stores in a High-Availability System".

Figure 13–2 shows a basic configuration of IMDB Cache DMs and their data stores in a high-availability system with two logical partitions and one BRM database schema. Dashed lines represent backup connections.
This section explains how to set up a high-availability BRM system that contains the following components:

- Two logical partitions for a single-schema database
- The ttGrid cache grid
- Data stores tt_0.0.0.1 and tt_0.1.0.1

For more information about installing IMDB Cache Manager, including hardware and software requirements, see "Installing and configuring BRM with IMDB Cache Manager" in BRM Installation Guide.
For an overview of IMDB Cache Manager, including information about cache grids and logical partitions, see "Using Oracle IMDB Cache Manager".

To set up a high-availability system for data stores tt_0.0.0.1 and tt_0.1.0.1:

1. Install and configure the BRM database with Oracle RAC. See "Configuring the BRM Database for High Availability".
2. Install Oracle Clusterware. See the Oracle Clusterware documentation.
3. On each node on which you plan to configure a data store, install an instance of Oracle IMDB Cache. See the Oracle TimesTen In-Memory Database Installation Guide.

To support the example data stores tt_0.0.0.1 and tt_0.1.0.1, you must configure a total of four instances of Oracle IMDB Cache (that is, an active and a standby instance for each data store). Each instance must reside on a different node.

---

**Important:** The primary group of the Oracle IMDB Cache owner should be the same as the primary group of the Oracle Clusterware owner. Their user names, however, can be different.

---

4. Install base BRM 7.4. See "Installing BRM" in BRM Installation Guide.
5. Install BRM 7.4 Patch Set 8. See the BRM 7.4 Patch Set 8 documentation.
6. On each node on which you plan to configure a data store, install an instance of the IMDB Cache DM. See "Installing and Configuring BRM with IMDB Cache Manager" in BRM Installation Guide.
7. Install any optional components that you want to add to your system.
8. Create and configure the active data stores tt_0.0.0.1 and tt_0.1.0.1. See "Creating and Configuring Active Data Stores".
9. If you have existing BRM data that was created in a BRM system before IMDB Cache Manager was installed, run the load_pin_uniqueness utility to prepare the data for migration to an IMDB Cache Manager–enabled system.

---

**Note:** Stop and restart the CM and DM.

---

10. Create the schema and load BRM objects into the active data stores. See "Initializing an Active Data Store".
11. Configure clusters for the active data stores. See "Configuring the Cluster for an Active Data Store".
12. Configure the standby data stores. See "Configuring Standby Data Stores".
13. Create the active/standby data store pairs, and register them with Oracle Clusterware. See "Creating Active and Standby Data Store Pairs".
14. Configure IMDB Cache DM instances to connect to the active and standby data stores. See "Associating IMDB Cache DM Instances with Data Stores".
15. Configure the CMs to connect to the IMDB Cache DM instances. See "Configuring Connection Managers for High Availability".
16. Register the IMDB Cache DM instances with Oracle Clusterware. See "Registering IMDB Cache DM Instances with Oracle Clusterware".
17. Perform post-installation tasks. See "IMDB Cache Manager Post Installation Tasks" in BRM Installation Guide.

Creating and Configuring Active Data Stores

To create and configure data stores for high availability, perform the following procedure on each node on which you want an active data store to reside.

1. Log on to the node for the active data store.
2. Create a directory for storing database files:
   
   ```
   mkdir BRM_Home/Database_Files_Location
   
   For example:
   ```
   ```
   mkdir BRM_Home/database_files
   ```

   **Note:** Oracle recommends using a local disk for database files instead of a network-mounted disk.

3. Go to the following directory:
   ```
   cd IMDB_Home/info
   ```
   
   where IMDB_Home is the directory in which Oracle IMDB Cache is installed.

4. Add the data store attributes to the `sys.odbc.ini` data store configuration file.
   ```
   [DSN]
   DataStore=/opt/portal/7.4/brm_database_files/Data_Store_Name
   OracleNetServiceName=PinDB
   DatabaseCharacterSet=UTF8
   ConnectionCharacterSet=UTF8
   PLSQL=1
   OracleNetServiceName=pin_db
   oraclepwd=pin01
   Driver=IMDB_Home/lib/libtten.so
   #Shared-memory size in megabytes allocated for the data store.
   PermSize=32
   #Shared-memory size in megabytes allocated for temporary data partition, generally half the size of PermSize.
   TempSize=16
   PassThrough=0
   #Use large log buffer, log file sizes
   LogFileSize=512
   #Async repl flushes to disk before sending batches so this makes it faster
   #on Linux
   LogFlushMethod=2
   #Limit Ckpt rate to 10 mb/s
   CkptFrequency=200
   CkptLogVolume=0
   CkptRate=10
   ```

   **Note:** You can edit the `sys.odbc.ini` file in the `IMDB_Home/info` directory by commenting out the default configurations.
Connections=200
#Oracle recommends setting LockWait to 30 seconds.
LockWait=30
DurableCommits=0
CacheGridEnable=1

where:

- **DSN** is the data source name, which is the same as the data store name. The DSN must also be the same as the database alias name in the `tnsnames.ora` file. For more information about setting database alias names, see "Making a Data Store Accessible to IMDB Cache DM".

- **Data_Store_Name** is the name of the data store.

- **IMDB_Home** is the directory in which Oracle IMDB Cache is installed.

5. Save and close the file.

6. Go to the **IMDB_Home** directory and source the `ttenv.csh` file:

   ```bash
   cd IMDB_Home/bin
   source ttenv.csh
   ```

   where **IMDB_Home** is the directory in which Oracle IMDB Cache is installed.

7. Set up the Oracle IMDB Cache grid privileges in the BRM database:
   a. Connect to the BRM database as a system administrator:

   ```sql
   cd IMDB_Home/oraclescripts
   sqlplus sys as sysdba
   ```

   b. Run the following SQL scripts:

   ```sql
   @IMDB_Home/oraclescripts/initCacheGlobalSchema.sql "Data_Store_User";
   @IMDB_Home/oraclescripts/grantCacheAdminPrivileges.sql "Data_Store_User"
   ```

   where **Data_Store_User** is the IMDB Cache data store user.

   c. Run the following commands to grant privileges:

   ```sql
   grant all on TIMESTEN.TT_GRIDID to "Oracle_DB_User";
   grant all on TIMESTEN.TT_GRIDINFO to "Oracle_DB_User";
   ```

   where **Oracle_DB_User** is the BRM database user.

   For more information, see the *Oracle TimesTen In-Memory Database Cache User’s Guide*.

8. Create the data store:

   ```bash
   cd IMDB_Home/bin
   ttIsql Data_Store_Name
   ```

   where **Data_Store_Name** is the name of the data store, such as `tt_0.0.0.1` and `tt_0.1.0.1`.

9. Create the data store user and grant all permissions:

   ```bash
   ttIsql Data_Store_Name
   create user Data_Store_User identified by Data.Store_Password;
   grant all to Data_Store_User;
   ```

   where:
Configuring IMDB Cache Manager for High Availability

Important: The IMDB Cache data store user must be the same as the BRM database user. However, the data store user password can be different from the database user password.

10. Set the data store user and password, and make the data store grid-enabled:

```sql
sqlite "uid=Data_Store_User;pwd=Data_Store_Password;dsn=Data_Store_Name"
call ttcacheuidpwdset('Cache_Admin_User', 'Cache_Admin_User_Pwd');
call ttGridCreate('ttGrid');
call ttGridNameSet('ttGrid');
```

where:

- **Data_Store_User** is the Oracle IMDB Cache data store user name, which must be the same as the BRM database user name.
- **Data_Store_Password** is the password for the Oracle IMDB Cache user name.
- **Data_Store_Name** is the name of the data store.
- **Cache_Admin_User** and **Cache_Admin_User_Pwd** are the cache user name and password.

Important: Run the call `ttGridCreate('ttGrid')` command only once per database schema. For example, if you are configuring multiple active data stores for a single schema, run this command only when you configure first active data store.

For more information about creating a cache grid, see the Oracle TimesTen In-Memory Database Cache User’s Guide.

Note: To initialize the data stores for a multischema setup, you must generate the schema and load SQL files for each database schema by using the pin_tt_schema_gen utility. Then follow the preceding steps to initialize the data stores for each schema. See "Creating and Initializing the BRM Cache Group Schema".

**Initializing an Active Data Store**

To load BRM objects into an active data store, you must perform the following procedures:

1. Generate the BRM cache groups schema using the BRM database, and extract the data from the BRM database for caching. See "Generating the Schema and Load SQL Files for the Active Data Store".

2. Create and initialize the BRM cache groups schema in the active data store. See "Initializing the Data Store".

Data_Store_Name is the name of the data store.

Data_Store_User is the IMDB Cache data store user.

Data_Store_Password is the password for the IMDB Cache data store user.

Data_Store_Name is the name of the data store.

Data_Store_User is the IMDB Cache data store user.

Data_Store_Password is the password for the IMDB Cache data store user.

Important: The IMDB Cache data store user must be the same as the BRM database user. However, the data store user password can be different from the database user password.

10. Set the data store user and password, and make the data store grid-enabled:

```sql
sqlite "uid=Data_Store_User;pwd=Data_Store_Password;dsn=Data_Store_Name"
call ttcacheuidpwdset('Cache_Admin_User', 'Cache_Admin_User_Pwd');
call ttGridCreate('ttGrid');
call ttGridNameSet('ttGrid');
```

where:

- **Data_Store_User** is the Oracle IMDB Cache data store user name, which must be the same as the BRM database user name.
- **Data_Store_Password** is the password for the Oracle IMDB Cache user name.
- **Data_Store_Name** is the name of the data store.
- **Cache_Admin_User** and **Cache_Admin_User_Pwd** are the cache user name and password.

Important: Run the call `ttGridCreate('ttGrid')` command only once per database schema. For example, if you are configuring multiple active data stores for a single schema, run this command only when you configure first active data store.

For more information about creating a cache grid, see the Oracle TimesTen In-Memory Database Cache User’s Guide.

Note: To initialize the data stores for a multischema setup, you must generate the schema and load SQL files for each database schema by using the pin_tt_schema_gen utility. Then follow the preceding steps to initialize the data stores for each schema. See "Creating and Initializing the BRM Cache Group Schema".

**Initializing an Active Data Store**

To load BRM objects into an active data store, you must perform the following procedures:

1. Generate the BRM cache groups schema using the BRM database, and extract the data from the BRM database for caching. See "Generating the Schema and Load SQL Files for the Active Data Store".

2. Create and initialize the BRM cache groups schema in the active data store. See "Initializing the Data Store".
Generating the Schema and Load SQL Files for the Active Data Store

Use the `pin_tt_schema_gen` utility to generate the schema SQL file with the BRM cache groups schema and the load SQL file with the BRM data. For more information, see "Creating and Initializing the BRM Cache Group Schema".

To generate the schema and load SQL files, perform the following procedure on the node on which the active data store resides:

1. Open the `BRM_Home/bin/pin_tt_schema_gen.values` file, and configure the values in the file.

   **Note:** You must generate the load SQL for each active data store.

   For example, generate `tt_load_0.0.0.1.sql` with `$db_no_for_load_sql` set to 0.0.0.1 and generate `tt_load_0.1.0.1.sql` with `$db_no_for_load_sql` set to 0.1.0.1 in the `pin_tt_schema_gen.values` file.

2. Save and close the file.

3. Run the following command:

   ```
   source BRM_Home/source.me.csh
   ```

4. Run the `pin_tt_schema_gen` utility with the `-a` parameter:

   ```
   ./pin_tt_schema_gen -a
   ```

   See `pin_tt_schema_gen`.

   **Note:** If you do not specify the values for `MAIN_DB['user']` and `MAIN_DB['password']` in the `pin_tt_schema_gen.values` file, the `pin_tt_schema_gen` utility prompts you to enter those values.

This updates the BRM database with unique indexes and non-null constraints and generates the following files:

- `tt_schema.sql`
- `tt_load_Logical_Partition.sql`
- `tt_drop.sql`

where `Logical_Partition` is the name of the logical partition in which the data store resides, such as `0.0.0.1` and `0.1.0.1`.

**Initializing the Data Store**

Use the schema and load SQL files to create the BRM cache groups schema and to load the BRM data.

To initialize an active data store, perform the following procedure on the node on which the active data store resides:

1. Set the data store user and password:
ttSql "uid=Data_Store_User; pwd=Data_Store_Password; dsn=Data_Store_Name"
call ttcacheidpwdset('Cache_Admin_User', 'Cache_Admin_User_Pwd');

where:
Data_Store_User is the Oracle IMDB Cache data store user name, which must be
the same as the BRM database user name.
Data_Store_Password is the password for the Oracle IMDB Cache user name.
Data_Store_Name is the name of the data store, such as tt_0.0.0.1 and tt_0.1.0.1.
Cache_Admin_User and Cache_Admin_User_Pwd are the cache user name and
password.

2. Start the cache agent:
call ttcachestart;

3. Create the schema:
run BRM_Home/bin/tt_schema.sql;

4. Create stored procedures:
run BRM_Home/sys/dm_tt/data/tt_create_pkg_pin_sequence.plb;
run BRM_Home/sys/dm_tt/data/tt_create_procedures.plb;
run BRM_Home/sys/dm_tt/data/create_tt_wrappers.plb;

---

**Note:** Load the stored procedures in tt_create_pkg_pin_sequence.plb before the procedures in tt_create_procedures.plb.

---

### Configuring the Cluster for an Active Data Store

To set up the cluster configuration for an active data store:

1. Log on to the node on which the active data store resides.
2. Go to the following directory:
   ```
cd IMDB_Home/info
   ```
   where IMDB_Home is the directory in which Oracle IMDB Cache is installed.
3. Add the following entries to the `cluster.oracle.ini` data store configuration file.
   You can edit the `cluster.oracle.ini` file in the `IMDB_Home/info` directory by
   commenting out the default configurations.

   **Note:** The `MasterHosts` entry identifies the nodes on which a pair of
   active/standby data stores resides. The order in which the nodes are
   specified sets the default states (active, standby) of the data stores.

   ```ini
   [DSN]
   MasterHosts = Active_Node, Standby_Node
   ScriptInstallDir = /export/home/ttinstaller/TimesTen/tt1121_HA/info/crs_scripts
   CacheConnect = Y
   ReturnServiceAttribute = RETURN TWOSAFE
   GridPort = Active_Port, Standby_Port
   ```
where:

- **DSN** is the data source name, which is the same as the data store name. The DSN must also be the same as the database alias name in the `tnsnames.ora` file. For more information about setting database alias names, see "Making a Data Store Accessible to IMDB Cache DM".

- **Active_Node** is the host name for the active data store.

- **Standby_Node** is the host name for the standby data store.

- **Active_Port** is an unused port number assigned to the active data store.

- **Standby_Port** is an unused port number assigned to the standby data store.

**Note:**  *Active_Port* and *Standby_Port* are ports on the private network that is set up as part of Oracle Clusterware installation.

4. Save and close the file.

### Configuring Standby Data Stores

To configure a standby data store for an active data store, perform this procedure on the node on which you want the standby data store to reside.

Before performing this procedure, obtain the following information:

- The name and location of the directory used to store database files for the active data store. See step 2 in "Creating and Configuring Active Data Stores".

- A copy of the entries added to `sys.odbc.ini` file for the active data store. See step 4 in "Creating and Configuring Active Data Stores".

- A copy of the entries added to `cluster.oracle.ini` file for the active data store. See step 3 in "Configuring the Cluster for an Active Data Store".

To create and configure a standby data store for an active data store:

1. Log on to the node on which you want the standby data store to reside.

2. Create a directory for storing database files:

   ```
   mkdir BRM_Home/Database_Files_Location
   ```

3. Go to the following directory:

   ```
   cd IMDB_Home/info
   ```

   where **IMDB_Home** is the directory in which Oracle IMDB Cache is installed.

4. In the `sys.odbc.ini` data store configuration file, add the *same* entries that you added to the active data store’s `sys.odbc.ini` file. See step 4 in "Creating and Configuring Active Data Stores".

5. Save and close the file.
6. In the `cluster.oracle.ini` data store configuration file, add the same entries that you added to the active data store’s `cluster.oracle.ini` file. See step 3 in “Configuring the Cluster for an Active Data Store”.

7. Save and close the file.

Creating Active and Standby Data Store Pairs

Use this procedure to perform the following tasks:

- Create a pair of active and standby data stores.
- Register the TimesTen agent and the data stores with Oracle Clusterware.
- Replicate the active data store’s BRM cache groups schema and data in the standby data store.

Perform the procedure on each active data store node.

1. Go to the `bin` directory on the node on which the active data store resides:
   ```
   cd IMDB_Home/bin
   ```
   where `IMDB_Home` is the directory in which Oracle IMDB Cache is installed.

2. Register the cluster information on the host as a root user by entering the following command:
   ```
   ttCWAdmin -ocrconfig
   ```

3. Start the Oracle IMDB Cache cluster agent using the TimesTen administrator user login:
   ```
   ttCWAdmin -init
   ```

4. Create an active/standby pair replication scheme:
   ```
   ttCWAdmin -create -dsn Data_Store_Name
   ```
   where `Data_Store_Name` is the name of the data store, such as `tt_0.0.0.1` or `tt_0.1.0.1`.

5. Provide the required information such as the admin user ID and password. A confirmation message is displayed when the registration is complete.

6. Start the active/standby pair replication scheme:
   ```
   ttCWAdmin -start -dsn Data_Store_Name
   ```
   Oracle Clusterware automatically starts the data stores.

7. Load data into the active/standby pair replication scheme:
   ```
   ttisql Data_Store_Name
   run BRM_Home/bin/tt_load_Logical_Partition.sql;
   ```
   where `Logical_Partition` is the database number of the logical partition in which the data stores reside, such as `0.0.0.1` or `0.1.0.1`.

To initialize the data stores for a multischema setup, you must generate the schema and load SQL files for each database schema by using the `pin_tt_schema_gen` utility. Then follow the steps in this section to initialize the data stores for each schema.

For more information about registering data stores with Oracle Clusterware, see the Oracle Clusterware Administration and Deployment Guide.
For more information about TimesTen replication, see the *TimesTen to TimesTen Replication Guide*.

**Associating IMDB Cache DM Instances with Data Stores**

For high availability, you need an active and a standby instance of each IMDB Cache DM. The active DM instance is connected to the active data store, and the standby DM instance is connected to the standby data store.

To connect active and standby IMDB Cache DM instances to the active and standby data stores, perform the following procedures on each node on which an active or a standby data store resides:

1. **Making a Data Store Accessible to IMDB Cache DM**
2. **Configuring an IMDB Cache DM Instance for a Data Store**
3. **Configuring pin_ctl to Start and Stop IMDB Cache DM**

**Note:** These procedures assume that you have installed an instance of IMDB Cache Manager on each node on which an active or a standby data store resides.

### Making a Data Store Accessible to IMDB Cache DM

To configure a data store so that IMDB Cache DM can directly connect to it, perform the following procedure on the node on which the data store resides:

1. Open the `tnsnames.ora` file located in the directory specified by `$TNS_ADMIN`.
2. Add the following entry:

   
   ```
   Database_Alias_Name = (DESCRIPTION = (ADDRESS = (PROTOCOL = ) (HOST = ) (PORT = ))
   (CONNECT_DATA =
   (SERVICE_NAME = Data_Store_Name)
   (SERVER = timesten_direct )))
   ```

   where:
   - `Database_Alias_Name` is the data store name specified in the `sys.odbc.ini` file.
   - `Data_Store_Name` is the data store name specified in the `sys.odbc.ini` file.
   - `timesten_direct` indicates that instances of IMDB Cache DM on the same node as the data store can directly connect to the data store.

   **Note:** You must add a separate entry for each logical partition.

For example:

```plaintext
tt_0.0.0.1 = (DESCRIPTION = (ADDRESS = (PROTOCOL = ) (HOST = ) (PORT = ))
(COMPRESS_DATA =
(SERVICE_NAME = tt_0.0.0.1)
(SERVER = timesten_direct )))
```

```plaintext
tt_0.1.0.1 = (DESCRIPTION = (ADDRESS = (PROTOCOL = ) (HOST = ) (PORT = ))
(COMPRESS_DATA =
(SERVICE_NAME = tt_0.1.0.1)
(SERVER = timesten_direct )))
```
3. Save and close the file.

**Configuring an IMDB Cache DM Instance for a Data Store**

Use the existing IMDB Cache DM installation to configure the settings for the active data store.

To configure IMDB Cache DM for the active data store:

1. Open the `BRM_Home/sys/dm_tt/pin.conf` file.
2. Set the `tt_ha_enabled` entry to 1:
   ```
   - dm tt_ha_enabled 1
   ```
3. Set the `sm_database_tt` entry to the data store:
   ```
   - dm sm_database_tt Data_Store_Name
   ```
   where `Data_Store_Name` is the name of the data store, such as `tt_0.0.0.1` or `tt_0.1.0.1`.
4. Set the `sm_pw_tt` entry to the data store password:
   ```
   - dm sm_pw_tt Data_Store_Password
   ```
   `Data_Store_Password` is the password for the IMDB Cache data store user.

   **Important:** The IMDB Cache data store user must be the same as the BRM database user. However, the data store user password can be different from the database user password.

5. Set the `logical_partition` entry to 1 to enable logical partitioning:
   ```
   - dm logical_partition 1
   ```
6. Save and close the file.

**Configuring pin_ctl to Start and Stop IMDB Cache DM**

To configure the `pin_ctl` utility to start and stop an instance of IMDB Cache DM, perform the following procedure on the node on which the DM and its associated data store reside:

1. Set the `TIMESTEN_HOME` environment variable to the directory in which the IMDB Cache DM is installed on the node. For example:
   ```
   /export/home/ttinstaller/TimesTen/tt1121_HA
   ```
2. Go to the directory in which the IMDB Cache DM is installed and source the `source.me` file:
   ```
   - Bourne shell:
     ```
     . source.me.sh
     ```
   - C shell:
     ```
     source source.me.csh
     ```
3. Open the `BRM_Home/bin/pin_ctl.conf` file in a text editor.
4. Add the following line to the startup configuration section of the file:

```
Start_DMTTInstance_Service_Name c1pidproc:DMTTInstance_Service_Name:
cport:DMTT_Port host:DMTT_Host dbno:Data_Store_DB_Number
```

where:

- **Start_DMTTInstance_Service_Name** is the name of the start command for the IMDB Cache DM instance.
- **DMTTInstance_Service_Name** is a simple process name matching filter.
- **DMTT_Port** is the IMDB Cache DM port number.
- **DMTT_Host** is the IMDB Cache DM host name.
- **Data_Store_DB_Number** is the data store database number.

For example:
```
start_dm_tt c1pidproc:dm_tt: cport:1234 host:vm31230 dbno:0.0.0.1
```

5. Save and close the file.

6. To ensure that pin_ctl is configured correctly, run the following commands:

   - `pin_ctl start dm_tt`
   - `pin_ctl stop dm_tt`

   For more information about configuring pin_ctl for high-availability, see "Using the pin_ctl Utility to Monitor BRM".

**Configuring Connection Managers for High Availability**

Figure 13–3 shows a basic configuration of CMs in a high-availability system with two logical partitions. Dashed lines represent backup connections.
To configure CMs for high availability:

1. Install at least two CMs. Use as many as you estimate will support your workload, and then add one more.

2. Install one real-time pipeline for each CM. See "Configuring Pipeline Manager".

3. Connect each CM to all the active and standby IMDB Cache DMs in your BRM System. See "Connecting CMs to IMDB Cache DMs".

Connecting CMs to IMDB Cache DMs

To configure a CM to connect to the active and standby IMDB Cache DM instances in a high-availability system, set the CM configuration file (`BRM_Home/sys/cm/pin.conf`) entries shown in Table 13–3. These settings minimize CM request failures and enable CM connections to succeed in the event of an IMDB Cache DM failover.

For information about how CMs handle IMDB Cache DM failure, see "About Connection Managers in a High-Availability System".
To enable Oracle Clusterware to manage IMDB Cache DM high availability, you must register each IMDB Cache DM instance with Oracle Clusterware.

**Table 13–3 CM pin.conf Entries for High Availability System**

<table>
<thead>
<tr>
<th>Configuration entry</th>
<th>Description</th>
</tr>
</thead>
</table>
| **dm_pointer**            | Specifies the host and port number of the IMDB Cache DM instances to connect to. Include an entry for each active and standby pair of IMDB Cache DMs in your system. Oracle recommends that the active DM be listed first in each entry. The CM pin.conf file should contain one dm_pointer entry per logical partition. Therefore, because the active and standby DM in each pair support the same logical partition, they must be on the same dm_pointer line:  
  - cm dm_pointer lp_number ip active_dm_host active_dm_port ip standby_dm_host standby_dm_port  
  **Example:**  
  - cm dm_pointer 0.0.0.1 ip 156.151.2.168 33950 ip 168.35.37.128 12960  
  - cm dm_pointer 0.1.0.1 ip 156.151.2.168 32250 ip 168.35.37.128 12850 |
| **pcm_timeout_in_msecs**  | Specifies the amount of time in milliseconds that the CM waits for a response from an IMDB Cache DM before failing over to the standby DM. This entry is called the CM’s long (failover) timeout (see "Configuring Multilevel CM Timeout for Client Requests"). The default value is 120000 (120 seconds). For high-availability systems, the recommended value is 100000 (100 seconds).  
  **Important:** In a high-availability system, each client-side component should have a longer timeout period than the server-side component that it calls. For example, to minimize CM request failures, make this timeout long enough for Oracle Clusterware to restart the IMDB Cache DM. For suggested timeout values, see "About Setting Up a High-Availability System".  
  **Example:**  
  - cm pcm_timeout_in_msecs 100000 |
| **pcm_op_max_retries**    | Specifies the maximum number of times an opcode is retried in the Portal Communications Model (PCM). The default value is 1. For high availability, the value must be at least 2.  
  **Example:**  
  - cm pcm_op_max_retries 2 |
| **cm_op_max_retries**     | Specifies the maximum number of times an opcode is retried in the CM. The default value is 1. For high availability, the value must be at least 2.  
  **Example:**  
  - cm cm_op_max_retries 2 |
| **pcm_bad_connection_retry_delay_time_in_secs** | Specifies the interval, in seconds, at which the CM tries to connect to the active IMDB Cache DM after it fails to connect. See "How CMs Handle IMDB Cache DM Failure".  
  **Important:** In a high-availability system, each client-side component should have a longer timeout period than the server-side component that it calls. For suggested timeout values, see "About Setting Up a High-Availability System".  
  **Example:**  
  - cm pcm_bad_connection_retry_delay_time_in_secs 100 |

**Registering IMDB Cache DM Instances with Oracle Clusterware**

To enable Oracle Clusterware to manage IMDB Cache DM high availability, you must register each IMDB Cache DM instance with Oracle Clusterware.
To register an IMDB Cache DM instance with Oracle Clusterware:

1. Log in to the Oracle Clusterware instance.
2. Create an action program.

Oracle Clusterware uses an action program to interact with IMDB Cache DM. It provides a method for starting, stopping, and checking the status of IMDB Cache DM.

You can create your own action program by using any existing action program, or you can use the sample action program `pin_ctl_dmtt.sh` located in `BRM_Home/bin`.

3. Set the following environment variables to customize the action program:
   - ORACLE_HOME
   - ORACLE64_HOME
   - PATH
   - LD_LIBRARY_PATH
   - LIBPATH_64
   - CLASSPATH
   - TNS_ADMIN
   - PIN_LOG

   **Note:** You must set the environment variable to the directory where BRM is installed.

   You must set the Oracle Clusterware environment for Oracle, IMDB Cache DM and BRM binaries, and IMDB Cache DM and BRM libraries properly. Otherwise, you must modify the action script to set the environment. A modified sample script is available in the `BRM_Home/bin/pin_ctl_dmtt.sh` file.

   You must provide the required permissions to the Oracle Clusterware user.

4. Export the following environment variables:
   - ORACLE_HOME
5. Go to the directory where you installed the Third-Party package and source the `source.me.sh` file.

   Bourne shell:
   ```
   . source.me.sh
   ```

   C shell:
   ```
   source source.me.csh
   ```

6. To verify that you can start, stop, and check the IMDB Cache DM services, log in as the Oracle Clusterware user and execute the following commands:

   ```
   sh /export/pinhome/pin9711/opt/portal/7.4/bin/pin_ctl_dmtt.sh start
   sh /export/pinhome/pin9711/opt/portal/7.4/bin/pin_ctl_dmtt.sh stop
   sh /export/pinhome/pin9711/opt/portal/7.4/bin/pin_ctl_dmtt.sh check
   ```

7. Create a profile for the IMDB Cache DM instance.

   The profile specifies how the IMDB Cache DM instance is registered inside the Oracle Cluster Registry.

   For example:

   ```bash
   $ORA_CRS_HOME/bin/crs_profile \
   -create IMDB_Cache_DM_Instance_Name \ 
   -t application \ 
   -d "In-Memory Database Cache DM" \ 
   -a BRM_Home/bin/pin_ctl_dmtt.sh \ 
   -o ci=5,ra=60,as=always,ut=20
   ```

   where `IMDB_Cache_DM_Instance_Name` is a unique name within your BRM system for the IMDB Cache DM instance. For example, you might use the following names for the four IMDB Cache DM instances created to support the `tt_0.0.0.1` and `tt_0.1.0.1` data stores: `tt_0.0.0.1_1`, `tt_0.0.0.1_2`, `tt_0.1.0.1_1`, and `tt_0.1.0.1_2`.

   **Note:** For information about how to configure the profile resources, such as those specified in the `-o` entry, see the Oracle Clusterware documentation.

8. Register the IMDB Cache DM profile with Oracle Clusterware.

   Registering inserts information from the IMDB Cache DM profile into the Oracle Cluster Registry.

   For example:

   ```bash
   $ORA_CRS_HOME/bin/crs_register IMDB_Cache_DM_Instance_Name
   ```
9. Start the IMDB Cache DM instance:

```bash
crs_start IMDB_Cache_DM_Instance_Name
```

where `IMDB_Cache_DM_Instance_Name` is the name of the IMDB Cache DM instance.

For more information, see the *Oracle Clusterware Administration and Deployment Guide*.

---

**Note:** Give the following permissions to the Oracle Clusterware user:

- Read and execute permissions for the `BRM_Home/opt` directory.
- Read, write, and execute permissions for the `BRM_Home/var` directory.

---

**Configuring AAA Gateway Manager for High Availability**

This section provides information only about configuring settings related to high availability. For information about configuring other AAA Gateway Manager settings, see "Using BRM AAA Gateway Manager" in *BRM AAA Gateway Manager*.

For information about how AAA Gateway Manager handles failure, see "About AAA Gateway Manager in a High-Availability System".

**Figure 13–4** shows the basic configuration of AAA Gateway Manager in a high-availability system. Dashed lines represent backup connections.
To configure AAA Gateway Manager for high availability:

1. Install as many instances of AAA Gateway Manager as you need to support your workload, and then install one more.

   For information about installing AAA Gateway Manager, see "About installing AAA Gateway Manager" in BRM AAA Gateway Manager.

2. Configure the entries listed in Table 13–4 in the registry file BRM_Home/opt/ifw/MBI.reg or BRM_Home/opt/ifw/diameter_charge.reg of each AAA Gateway Manager instance in your BRM system.

   For more information about AAA Gateway Manager registry files, see "About configuring AAA Gateway Manager" in BRM AAA Gateway Manager.
Configure your network service control point (SCP)—which is not part of the BRM system—to interact with the AAA Gateway Manager instances by doing one of the following:

- (Recommended) Send requests to all the instances and distribute the workload handled by any failed instance to the remaining available instances.
- Connect only to a primary (default) instance and switch to a designated standby instance if the primary gateway fails.

For more information about configuring the SCP, see the SCP documentation.

---

**Important:** The SCP must be able to detect AAA Gateway Manager failure and to redistribute its workload or switch to the designated standby gateway if the primary one fails.

---

**Restoring a High-Availability System After Failover**

After failed components in a high-availability system are fixed, return the system to its original configuration by switching the workload back to the primary components. This enables the system to use its optimal architecture.

The following sections explain how to restore components in a high-availability system:

- **Rebalancing AAA Gateway Manager and CM Connections**
Switching Back to the Primary Oracle RAC Instance

Ensuring All Accounts are Billed

**Note:** The `pin_ctl` utility is the main BRM rebalancing tool. To use it, you must install and run the SNMP service on all the servers in your high-availability system. See “Using the SNMP Instrumentation Protocol to Monitor and Control BRM Components”.

For an overview of how these components function in a high-availability system, see "Understanding a High-Availability System".

**Rebalancing AAA Gateway Manager and CM Connections**

When a CM fails, its request load is automatically rebalanced among the remaining CMs. When the failed CM comes back online, the load is not automatically rebalanced. To rebalance the load, run the following command:

```
pin_ctl snmpset rebalanceAaaGw aaa
```

For more information, see "Rebalancing CM Connections" in *BRM AAA Gateway Manager*.

**Switching Back to the Primary Oracle RAC Instance**

To restart a failed Oracle RAC instance, run the following command:

```
srvcctl start instance -d racDatabaseName -i primary_racInstanceName
```

where:

- `racDatabaseName` is the name of the Oracle database.
- `primary_racInstanceName` is the name of the primary (preferred) Oracle RAC instance.

For information about the `srvcctl` command, see the Oracle RAC documentation.

After a failed database instance is restarted, the services for which it is the primary database instance do not automatically switch back to it from the backup instance.

To switch a database service from its backup database instance to its primary database instance, run the following command:

```
srvcctl relocate service -d racDatabaseName -s serviceName -i backup_racInstanceName -t primary_racInstanceName -f
```

where:

- `racDatabaseName` is the name of the Oracle database.
- `serviceName` is the name of the database service that the primary Oracle RAC instance originally supported.
- `backup_racInstanceName` is the name of the backup (available) Oracle RAC instance.
- `primary_racInstanceName` is the name of the primary (preferred) Oracle RAC instance.
Note: Switching database services back to their primary Oracle RAC instance causes a service interruption. Usually, however, switching back to the primary node takes less time than failing over to the backup node.

Ensuring All Accounts are Billed

If a system failure occurs while the pin_bill_day application is running, some operations might fail and thus bills might not be generated for all accounts.

To ensure that all accounts are billed, Oracle recommends rerunning pin_bill_day after the system is restored.
Part III describes how to improve performance for an Oracle Communications Business and Revenue Management (BRM) system. It contains the following chapters:

- Improving BRM Performance
- Optimizing Pipeline Manager Performance
- Optimizing BRM for Prepaid and Postpaid Convergence
This document provides information on evaluating and improving the performance of your Oracle Communications Billing and Revenue Management (BRM) system.

Before you read this document, you should be familiar with BRM system architecture. See "BRM System Architecture" in BRM Concepts.

For information on troubleshooting BRM, see "Resolving Problems in Your BRM System".

For information specifically about tuning Pipeline Manager, see "Optimizing Pipeline Manager Performance".

### Monitoring Performance

You can use BRM diagnostics tools to monitor performance. For example:

- You can use HP OpenView to display graphs and reports showing system CPU and memory usage.
- You can monitor opcode latency.
- You can monitor event data record (EDR) throughput in a pipeline.

See "About Monitoring BRM".

### Improving Connection Manager Performance

For information about Connection Managers (CMs), see "The Connection Management Tier" in BRM Concepts.

To improve CM performance, see the following topics:

- Increasing CM Login Performance
- Load Balancing CMs
- Improving Performance for Loading Large Price Lists
- Improving Real-Time Rating Performance
- Logging Non-Currency Events
- Specifying the Number of Connections to CMs
- Setting the CM Time Interval between Opcode Requests
Increasing CM Login Performance

To increase CM login performance, see the following topics:

- Using CM Proxy to Allow Unauthenticated Log On
- Turning Off Session-Event Logging
- Turning Off the Checking of Logons and Passwords

Using CM Proxy to Allow Unauthenticated Log On

Use CM Proxy to provide unauthenticated connections to BRM, typically for providing connections for incoming mail messages. Connections made through CM Proxy do not require you to log on, which increases performance.

Caution: Only connections that do not require authentication should use CM Proxy.

To increase security, you can restrict the types of operations performed by CM Proxy by specifying the opcodes that perform allowed operations.

To use CM Proxy:

1. Open the CM Proxy configuration file (`BRM_Home/sys/cm_proxy/pin.conf`).
2. Configure CM Proxy according to the guidelines in that file, and save the file.

   The following are some of the more important entries:

   - Use the `oplist` entry to specify the opcodes that can be performed by CM Proxy.
   - Use the `allowed` entry to specify the hosts that can use CM Proxy.
   - Use the `queue manager` entries to manage front-end and back-end connections. See "Improving Data Manager and Queue Manager Performance".
   - Use the `standard connection` entries to connect to a CM or CMMP (Connection Manager Master Process). See "About Connecting BRM Components".

3. Start CM Proxy. See "Starting and Stopping the BRM System".
4. Open the configuration file for each application you want to use CM Proxy. See "Locations of Configuration and Properties Files".
5. Change the `cm_ptr` entry to point to the machine running CM Proxy and change the `login_type` entry to 0 (no login name or password required).

Turning Off Session-Event Logging

By default, the CM writes a session-event storable object for each client connection, but you can suppress the creation of these session storable objects if you do not need these records of logins. You can turn off session-event logging for some CMs for better performance while keeping this feature for other CMs dedicated to applications that require session objects, such as the mail and terminal servers.

To turn off session-event logging:

1. Open the CM configuration file (`BRM_Home/sys/cm/pin.conf`).
2. Change the value for the `loginAudit` entry to 0 and make sure that the entry is not commented.

3. Stop and restart the CM. See "Starting and Stopping the BRM System".

**Turning Off the Checking of Logons and Passwords**

When an application tries to log on to BRM, the CM verifies the service specified by the application, asks for a login name, and verifies the login password. To improve performance, set up the CM to not ask for a login name or password. See "Configuring the CM to Verify Application Logins with the Service Only".

To turn off login name and password verification:

1. Open the CM configuration file (`BRM_Home/sys/cm/pin.conf`).

2. Change the value for the `cm_login_module` entry to read:
   ```
   - cm cm_login_module ./cm_login_null.extension
   ```
   where `extension` is the file type specific to your operating system: `so` for Solaris, Linux, or HP-UX IA64; or `a` for AIX. For example:
   ```
   - cm cm_login_module ./cm_login_null.so
   ```

3. Stop and restart the CM. See "Starting and Stopping the BRM System".

**Load Balancing CMs**

You can use two methods for balancing the load among multiple CMs:

- You can use a CMMP to provide additional reliability and to balance the load among multiple CMs. See "Using Connection Manager Master Processes".

- You can provide a simple failover system for connections to the CM by giving the application a list of CMs on the system. If the first CM in the list is unavailable, the application tries the next CM.

For example:

- `nap cm_ptr ip cm_host1 11960`
- `nap cm_ptr ip cm_host1 11961`

If the CMs are on separate machines:

- `nap cm_ptr ip cm_host1 11960`
- `nap cm_ptr ip cm_host2 11960`

**Tip:** You can point to multiple CMMPs rather than to individual CMs. If the CM provided by the first CMMP is not available, the application asks for a CM from the second CMMP.

For example:

- `nap cm_ptr ip CMMP_host1 11959`
- `nap cm_ptr ip CMMP_host2 11959`

**Improving Performance for Loading Large Price Lists**

If you have a large price list, you can improve performance in the following ways:
Cache pricing data, such as G/L IDs and resource IDs. In a test environment where you modify your price list often, caching pricing data improves performance because there is no need to load price reference objects every time you commit the price list to the database.

**Important:** Pricing data is created every time the CM starts. Whenever the pricing data is changed in the database, the CM must be stopped and restarted to place the new information into the cache.

In a production system where you rarely modify your price list, you do not need to cache pricing data. This reserves the CM cache for other uses and eliminates the need to stop and restart the CM to update the cache if you change the price list.

Turn off event logging for price list creation events.

**Note:** When you turn off event logging, BRM still stores audit trail information for products, deals, and plans; however, there will not be an event log of when the price plans were modified and who modified them.

To improve loading performance:

1. Open the CM configuration file (`BRM_Home/sys/cm/pin.conf`).
2. Edit the `cache_references_at_start` entry. The default is 1 (cache data).
   - `fm_price cache_references_at_start 1`
3. Edit the `fm_price_prod_provisioning_cache` entry. The default entries are usually sufficient. For more information, see the instructions in the configuration (`pin.conf`) file.
   - `cm_cache fm_price_prod_provisioning_cache 100, 102400, 13`
4. Edit the `fm_price_cache_beid` entry. The default entries are usually sufficient. For more information, see the instructions in the configuration (`pin.conf`) file.
   - `cm_cache fm_price_cache_beid 200, 524288, 32`
5. Edit the `log_price_change_event` entry. The default is 0 (events are not logged).
   - `fm_price log_price_change_event 0`
6. Save the file.
7. Stop and restart the CM. See "Starting and Stopping the BRM System".

**Improving Real-Time Rating Performance**

To improve real-time rating performance, see the following topics:

- **Changing the Precision of Rounded and Calculated Values**
- **Setting the Interval for Checking for Price List Changes**
- **Setting the Interval for Updating Zone Maps**
- **Filtering the ERAs Considered During Rating and Discounting**
- **Enabling and Disabling the Caching of Customized Products**
Changing the Precision of Rounded and Calculated Values

To improve performance, you can change the precision of rounded values and of values calculated by real-time rating. You change the precision by adding or modifying entries in the CM `pin.conf` file:

- To change the precision of rounded values, add or change the `rating_quantity_rounding_scale` entry. The value of this entry determines the number of digits to the right of the decimal place for rounding quantities. The default is 8.

- To change the precision of calculated values, add or change the `rating_max_scale` entry. The value of this entry determines the number of digits to the right of the decimal place that are used. The default is 10.

---

**Important:** You must stop and restart the CM after you change these values. See "Starting and Stopping the BRM System".

---

Setting the Interval for Checking for Price List Changes

You can set the interval at which BRM checks for changes to the price list. If you change the price list frequently, you may want to use a shorter interval. If your price list is less volatile, you can increase the interval.

To change the interval:

1. Open the CM configuration file (`BRM_Home/sys/cm/pin.conf`).
2. Edit the following entry:
   - `fm_rate refresh_product_interval 3600`
   
   The value of this entry determines the interval in seconds. The default is 3600.
3. Save the file.
4. Stop and restart the CM. See "Starting and Stopping the BRM System".

Setting the Interval for Updating Zone Maps

To specify how frequently BRM checks for changes to zone maps and updates them in the database:

1. Open the CM configuration file (`BRM_Home/sys/cm/pin.conf`).
2. Edit the following entry:
   - `fm_zonemap_pol update_interval 3600`
   
   The value of this entry determines the interval in seconds. The default is 3600.
3. Save the file.
4. Stop and restart the CM. See "Starting and Stopping the BRM System".

Filtering the ERAs Considered During Rating and Discounting

By default, real-time rating checks for both account-level extended rating attributes (`/profile/acct_extrating` object) and service-level ERAs (`/profile/serv_extrating` object) when it searches for rating and discounting criteria. You can improve real-time rating performance by filtering the types of ERAs that BRM considers when it searches for
rating and discounting criteria. For example, you can configure BRM to search for service-level ERAs only or to omit the ERA search altogether.

You can specify the types of ERAs to consider by modifying a field in the rating instance of the /config/business_params object.

You modify the /config/business_params object by using the pin_bus_params utility. See BRM Developer’s Guide.

To specify the ERA types:

1. Use the following command to create an editable XML file from the rating instance of the /config/business_params object:

   ```
   pin_bus_params -r BusParamsRating bus_params_rating.xml
   ```

   This command creates the XML file named `bus_params_rating.xml.out` in your working directory. If you do not want this file in your working directory, specify the full path as part of the file name.

2. Search the XML file for following line:

   ```xml
   <EnableEras>serviceAndAccount</EnableEras>
   ```

3. Change `serviceAndAccount` to one of the following:
   - `account`: Limits the rating and discounting criteria search to account-level ERAs by retrieving only the `/profile/acct_extrating` object.
   - `service`: Limits the rating and discounting criteria search to service-level ERAs by retrieving only the `/profile/serv_extrating` object.
   - `disabled`: Omits ERAs from the rating and discounting criteria. Because neither object is retrieved, this option provides the best performance.

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

4. Save the file.

5. Change the file name from `bus_params_rating.xml.out` to `bus_params_rating.xml`.

6. Use the following command to load the change into the /config/business_params object:

   ```
   pin_bus_params bus_params_rating.xml
   ```

   You should execute this command from the `BRM_Home/sys/data/config` directory, which includes support files used by the utility. To execute it from a different directory, see `pin_bus_params` in BRM Developer’s Guide.

7. To verify that all fields are correct, you can display the /config/business_params object by using Object Browser or by using the `robj` command with the `testnap` utility.

   For more information, see the following topics in BRM Developer’s Guide:
   - Using testnap
   - Reading Objects by Using Object Browser
8. Stop and restart the CM. For more information, see "Starting and Stopping the BRM System".

9. For multiple databases, run the pin_multidb script with the -R CONFIG parameter. For more information on this script, see "pin_multidb".

**Enabling and Disabling the Caching of Customized Products**

When you use advanced customization to create customized products, BRM uses the customized products for rating. You can control whether customized products are cached for use by the real-time rating engine.

- If you choose not to cache customized products (the default setting), the real-time rating engine retrieves customized product data from the database during rating. This slows rating performance but minimizes the memory impact of customized products.
- If you choose to cache customized products, the CM size grows as customized products are created. Because the products are cached in memory, however, rating performance is increased.

You enable product caching by changing the `EnableTailormadeCache` field in the `rating` instance of the `/config/business_params` object from 0 to 1. You can disable caching by changing the field back to 0.

You modify the `/config/business_params` object by using the `pin_bus_params` utility. See BRM Developer’s Guide.

To enable caching of customized products:

1. Use the following command to create an editable XML file from the `rating` instance of the `/config/business_params` object:
   ```
   pin_bus_params -r BusParamsRating bus_params_rating.xml
   ```
   This command creates the XML file named `bus_params_rating.xml.out` in your working directory. If you do not want this file in your working directory, specify the full path as part of the file name.

2. Search the XML file for following line:
   ```xml
   <EnableTailormadeCache>0</EnableTailormadeCache>
   ```

3. Change 0 to 1.

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

4. Save the file.

5. Change the file name from `bus_params_rating.xml.out` to `bus_params_rating.xml`.

6. Use the following command to load the change into the `/config/business_params` object:
   ```
   pin_bus_params bus_params_rating.xml
   ```
You should execute this command from the `BRM_Homesys/data/config` directory, which includes support files used by the utility. To execute it from a different directory, see `pin_bus_params` in BRM Developer’s Guide.

7. To verify that all fields are correct, you can display the `/config/business_params` object by using Object Browser or by using the `robj` command with the `testnap` utility.

   For information on using `testnap`, see "Using testnap" in BRM Developer’s Guide.
   For information on how to use Object Browser, see "Reading Objects by Using Object Browser" in BRM Developer’s Guide.

8. Stop and restart the CM. For more information, see "Starting and Stopping the BRM System".

9. For multiple databases, run the `pin_multidb` script with the `-R CONFIG` parameter. For more information on this script, see "pin_multidb".

### Configuring the Maximum Number of Products and Discounts Cached

Products and discounts that are used during the real-time rating process are automatically stored in the CM cache. This improves rating performance, but, over time, it can consume a large amount of memory. To prevent the CM cache from growing too large, you can set a maximum number of products and discounts that can be stored in the CM cache.

- When you set the maximum to a non-zero value, BRM prevents the real-time rating engine from storing more than the specified number of products and discounts in CM cache. When the maximum number is reached, BRM flushes 10% of the products and discounts from cache that have been used the least.

---

**Note:** The maximum number of products and discounts that should be stored in CM cache is dependent on the your business needs.

---

- When you set the maximum to zero, BRM does not regulate the number of products and discounts stored in the CM cache. This is the default.

You configure the maximum number of products and discounts that can be cached by configuring the `ProductsDiscountsThreshold` field in the `rating` instance of the `/config/business_params` object.

To set a maximum number of products and discounts that can be cached:

1. Use the following command to create an editable XML file from the `rating` instance of the `/config/business_params` object:

   ```
   pin_bus_params -r BusParamsRating bus_params_rating.xml
   ```

   This command creates the XML file named `bus_params_rating.xml.out` in your working directory. If you do not want this file in your working directory, specify the full path as part of the file name.

2. Search the XML file for following line:

   ```
   <ProductsDiscountsThreshold>0</ProductsDiscountsThreshold>
   ```

3. Change `0` to the maximum number of products and discounts that you would like stored in cache. The default value of `0` specifies to not regulate the number of products and discounts in the CM cache.
Improving Connection Manager Performance

Improving BRM Performance

4. Save the file.

5. Change the file name from `bus_params_rating.xml.out` to `bus_params_rating.xml`.

6. Use the following command to load the change into the `/config/business_params` object:

   ```
   pin_bus_params bus_params_rating.xml
   ```

   You should execute this command from the `BRM_Home/sys/data/config` directory, which includes support files used by the utility. To execute it from a different directory, see `pin_bus_params` in BRM Developer’s Guide.

7. To verify that all fields are correct, you can display the `/config/business_params` object by using Object Browser or by using the `robj` command with the `testnap` utility.

   For more information, see the following topics in BRM Developer’s Guide:
   - Using `testnap`
   - Reading Objects by Using Object Browser

8. Stop and restart the CM. For more information, see "Starting and Stopping the BRM System".

9. For multiple databases, run the `pin_multidb` script with the `-R CONFIG` parameter. For more information on this script, see "pin_multidb".

Improving the Performance of your Multithreaded Applications

You can improve the performance of multithreaded applications by controlling the thread load and monitoring the thread activity.

Controlling Thread Load on Your Multithreaded Application

The number of child threads allowed is controlled by the `children` entry in the application `pin.conf` file. You can set the number of threads manually by editing the `pin.conf` file or automatically by using the `pin_mta_monitor` monitoring utility.

To improve performance, you can limit the number of child threads. For example, you can limit the number of threads during the business day to prevent excessive load on the network and increase the number of threads during off-peak hours.

To control the number of threads for an MTA by using `pin_mta_monitor`:

   ```
   Note: After entering a command at the monitor prompt, press the spacebar once and then press Enter. If you do not enter a space after the command, the utility does not run the command.
   ```

1. Go to the directory from which you run the MTA.

2. Start the `pin_mta_monitor` utility using the following command:
pin_mta_monitor mta_application

where mta_application is the MTA that pin_mta_monitor tracks.

3. The (mon)> prompt appears. Provide the number by which you want to increase or decrease the number of threads that mta_application uses as the value for number in the appropriate command (press the spacebar and then press Enter).
   - To increase the number of threads that mta_application uses, enter:
     (mon)> t+number
   - To decrease the number of threads that mta_application uses, enter:
     (mon)> t-number

For example, the following commands increase the thread pool of pin_inv_export by two threads:

pin_mta_monitor pin_inv_export
(mon)> t+2

Monitoring the Thread Activity of Your Multithreaded Application

Use the pin_mta_monitor utility to monitor the thread activity of your MTA.

Note: After entering a command at the monitor prompt, press the spacebar once and then press Enter. If you do not enter a space after the command, the utility does not run the command.

To monitor the thread activity of an MTA:

1. Go to the directory from which you run the MTA.
2. Start the pin_mta_monitor utility using the following command:
   
   pin_mta_monitor mta_application

   where mta_application is the MTA that pin_mta_monitor tracks. For example, the following command is used to monitor the activity of pin_inv_export:

   pin_mta_monitor pin_inv_export

3. The (mon)> prompt appears.

To print the thread activity of the mta_application to stdout:

   (mon)> p

The following is a sample output of the thread activity that pin_mta_monitor prints for pin_inv_export:

   Required:2, cmd:print
   Thread_id:2, Working POID:0.0.0.1 /invoice 221084 0, Flag:0
   Thread_id:3, Working POID:0.0.0.1 /invoice 220860 0, Flag:0

4. To stop printing the thread activity of the mta_application to stdout:

   (mon)> q
Logging Non-Currency Events

By default, BRM is configured to give maximum account-creation performance by logging only events that have a balance impact associated with a currency. To log noncurrency events at the expense of system performance, change the `creation_logging` entry in the CM configuration file (`pin.conf`).

By default, these events are logged:
- `/event/billing/charge`
- `/event/billing/deal`
- `/event/billing/product`

By default, these events are not logged:
- `/event/customer/billinfo`
- `/event/customer/nameinfo`
- `/event/customer/login`
- `/event/customer/password`
- `/event/customer/status`
- `/event/billing/limit`

To log non-currency events:

1. Open the CM configuration file (`BRM_Home/sys/cm/pin.conf`).
2. Edit the `creation_logging` entry:
   - `fm_cust creation_logging 1`
3. Save and close the file.

The new value becomes effective immediately and applies to the next account created. You do not need to restart the CM to enable this entry.

Specifying the Number of Connections to CMs

The `cm_max_connects` entry in the CM configuration file (`pin.conf`) tells the CM how many client applications can connect at one time. If client applications are having trouble connecting to the database, you can increase the number of connections. Performance degrades when too many CMs are running on the same machine, depending on the system load.

The maximum number of connections is 1000.

---

**Note:** Normally, `cm_max_connects` is commented out. This type of connection is best handled by CM Proxy. See "Using CM Proxy to Allow Unauthenticated Log On".

---

1. Open the CM configuration file (`BRM_Home/sys/cm/pin.conf`).
2. Edit the `cm_max_connects` entry:
   - `cm cm_max_connects 300`
3. Save and close the file.
4. Stop and restart the CM.
Setting the CM Time Interval between Opcode Requests

When a client application requests a connection, the CM spawns a child process to handle the connection. The child process or thread communicates with the client application by receiving requests in the form of opcodes.

By default, the child process or thread waits infinitely for each opcode request, but you can set a time interval after which the child process or thread terminates if no request has arrived.

1. Open the CM configuration file (`BRM_Home/sys/cm/pin.conf`).
2. Edit the `cm_timeout` entry.
   
   ```
   cm cm_timeout 3
   ```

   The value of this entry specifies the time interval in minutes.
3. Save and close the file.
4. Stop and restart the CM. See "Starting and Stopping the BRM System".

Using Connection Manager Master Processes

The Connection Manager Master Process (CMMP) routes connections from a single BRM client to multiple CMs. See "About Connection Manager Master Processes (CMMPs)" in BRM Concepts.

To specify which CMs to route connections to, you list the CMs in the CMMP configuration file. (See "Setting Up a CMMP".) You can also specify how the CMMP chooses a CM:

- By default, the CMMP chooses a CM randomly from the list of CMs.
- You can specify that the CMMP choose CMs sequentially from the first entry to the last, and then back to the first (known as round-robin selection).

The CMMP does not validate whether the CMs it points to are running. If the CMMP sends a transaction to a nonfunctioning CM, the client can tell that the CM is offline and tries to connect to a different CM or CMMP. Therefore, you should include multiple `cm_ptr` entries to the same CMMP or to more than one CMMP.

Because it is the client’s responsibility to ensure a connection to a CM, there must be at least one entry in the CMMP `pin.conf` file for each CM to which it is connected.

Sample CMMP Configuration

In this example shown in Figure 14–1:

- There are two CMMP host machines, CMMP_host1 and CMMP_host2.
- There are four CM host machines, CM_host1, CM_host2, CM_host3, and CM_host4.
- Each CM points to its own DM.
Client configuration file

Using the connection parameters shown below, the clients attempt to connect to all CMs through both CMMPs:

- nap cm_ptr ip CMMP_host1 11959
- nap cm_ptr ip CMMP_host1 11959
- nap cm_ptr ip CMMP_host1 11959
- nap cm_ptr ip CMMP_host1 11959
- nap cm_ptr ip CMMP_host2 11959
- nap cm_ptr ip CMMP_host2 11959
- nap cm_ptr ip CMMP_host2 11959
- nap cm_ptr ip CMMP_host2 11959

CMMP configuration files

The following redirect entries are required for client-to-CM connectivity. The parameters should be the same for both CMMP configuration files:

- cm redirect - CM_host1 11960
- cm redirect - CM_host2 11960
- cm redirect - CM_host3 11960
- cm redirect - CM_host4 11960

If the CMs are on the same host, the port numbers need to be unique:

- cm redirect - CM_host1 11961
- cm redirect - CM_host2 11962
- cm redirect - CM_host3 11963
- cm redirect - CM_host4 11964

Note: Make sure you also set the - cm cmmp_algorithm parameter.
Setting Up a CMMP

To set up a CMMP:

1. Open the CMMP configuration file (`BRM_Home/sys/cmmp/pin.conf`).
2. Configure the CMMP according to the guidelines in the file:
   - Use the `redirect` entry to list the CMs on your system. For example:
     ```
     - cm redirect - CM_host1 11960
     - cm redirect - CM_host2 11960
     - cm redirect - CM_host3 11960
     - cm redirect - CM_host4 11960
     ```
   - Use the `cmmp_algorithm` entry to specify whether the CMMP chooses CMs randomly (the default) or sequentially.
3. Save and close the file.
4. Start the CMMP. See “Starting and Stopping the BRM System”.
5. Open the configuration file for each application you want to use CMMP. See "Locations of Configuration and Properties Files".
6. Change the `cm_ptr` entry in the client application `pin.conf` file to point to the machine running the CMMP.
7. Save and close each configuration file you change.

Improving Data Manager and Queue Manager Performance

For information about Data Managers (DMs), see "The Data Management Tier” in BRM Concepts.

To improve DM performance, see the following topics:

- About Queuing-Based Processes
- Configuring DM Front Ends and Back Ends
- Setting DM Shared Memory Size
- Reducing Resources Used for Search Queries
- Load Balancing DMs
- Optimizing Memory Allocation during Database Searches
- Increasing DM CPU Usage
- Examples of DM Configurations

About Queuing-Based Processes

Queuing improves system performance by lowering the number of connections to the database. This reduces the number of processes and therefore reduces the system load required to handle the connections.

Queuing is used in two different types of system components:

- The RADIUS Manager, CM Proxy, and Web Interface daemons use queuing to connect incoming client connections to CMs. In this case, queuing reduces the number of client connections to CMs.
- All DMs use queuing internally. Front-end processes pass requests and data through a queue to back-end processes. In this case, queuing reduces the number of connections to the database.

CMs and Connection Manager Master Processes (CMMPs) do not use queuing.

Figure 14–2 shows an example of where queuing takes place in the BRM system architecture. Note that queuing occurs in two locations. In this example, connections are queued in the RADIUS Manager and in the database DM.

**Figure 14–2  BRM Queuing Locations**

[Diagram showing queuing locations]

**Example of Queuing in a Client-to-CM Connection**

Figure 14–3 shows the RADIUS daemon running on a system. Front-end processes pass the connections to a shared-memory queue where the connections wait for available back ends. The back ends connect to CMs.
Configuring DM Front Ends and Back Ends

You configure DM performance by specifying the number of front-end and back-end processes and the amount of shared memory the DM uses.

Note: Queue Manager (QM) components; for example, LDAP Manager, use the same types of configuration entries, but they have different names. For example, instead of `dm_max_fe`, the entry is named `qm_max_fe`. The functionality is the same.

Use the following DM and QM `pin.conf` entries to tune performance:

- `dm_n_fe`: Specifies the number of DM front-end processes.
- `dm_n_be`: Specifies the maximum number of DM back-end processes.
- `dm_max_per_fe`: Specifies the maximum number of connections for each front end.
- `dm_trans_be_max`: Specifies the maximum number of back ends that can be used for processing.
- `dm_trans_timeout`: Specifies the time in minutes that DM back-end processes wait for the next opcode in a transaction.

To change one or more of these parameters:

1. Open the DM configuration file (`BRM_Home/sys/dm_oracle/pin.conf`).
2. Change the configuration entry associated with the parameter. For tuning guidelines, see the topics following this procedure. For the syntax of each configuration entry, follow the guidelines in the configuration file.
3. Save and close the file.

4. Stop and restart the DM. See "Starting and Stopping the BRM System".

---

**Important:** Besides configuring the number of connections for best performance, remember to keep the number of connections within the terms of your database license agreement.

---

**Ratio of Front Ends to Back Ends**

Oracle recommends that the total number of front ends (specified in the `dm_n_fe` and `dm_max_per_fe` entries) should be two to four times the number back ends (specified in the `dm_n_be` entry).

In this example, the total number of front ends is 64 (4 times 16), which is 4 times the number of back ends.

- `dm dm_n_fe 4`
- `dm dm_max_per_fe 16`
- `dm dm_n_be 16`

**Providing Enough Front-End Connections**

If connection errors occur between the CM and DM, increase the values in the `dm_n_fe` and `dm_max_per_fe` entries. If there are not enough front ends, BRM reports an error; for example:

```
DMfe #3: dropped connect from 194.176.218.1:45826, too full
W Thu Aug 06 13:58:05 2001 dmhost dm:17446 dm_front.c(1.47):1498
```

Check the `dm_database.log` and `dm_database.pinlog` files for errors.

You must have enough DM front-end connections (number of processes times the number of connections for each process) to handle the expected number of connections from all of the CMs. Otherwise, you will see errors when the applications cannot connect to BRM.

Connections might be required for the following:

- One connection for each terminal server thread (`pin_radiusd`).
- One connection for each CM Proxy thread.
- One connection for each Web interface thread, plus one additional connection if customers create accounts with a Web interface.
- One connection for each billing application thread plus one additional connection for the master search thread.
- Two connections for each instance of Customer Center.

The maximum number of connections each front-end process can handle depends on the activity of the connection and, on multi-processor machines, the processor speed. For intensive connections, such as a heavily utilized terminal server, a front end might be able to handle only 16 connections. For intermittent connections, such as through certain client tools, a single front end can handle 256 or 512 connections. For systems that use a combination of these activities; for example, real-time processing with some client tool activity, you can configure an intermediate value for the maximum connections per front end.

For a given number of connections, if you have too many front ends (too few connections for each front end), the DM process uses too much memory and there is
Improving Data Manager and Queue Manager Performance

too much context switching. Conversely, if you have too few front ends (too many connections for each front end), the system performs poorly.

Determining the Required Number of Back Ends
You configure the number of back ends to get maximum performance from your system, depending on the workload and the type of BRM activity. Here are some guidelines for various activities:

- **Authentication/authorization**: For processing terminal server requests, which consist of many single operations without an explicit transaction, size the number of back ends to handle the traffic and leave the percentage of back ends available for transactions at the default value (50%).

  For example:
  
  ```
  -dm dm_n_be 48
  -dm dm_trans_be_max 24
  ```

  Normally, however, you configure the DM to perform a variety of tasks.

- **Account creation**: This activity uses one transaction connection for a long time and a second regular connection intermittently. You must provide two back ends for each of the accounts you expect to be created simultaneously. Your system might lock up if you do not have enough back ends. You can leave the percentage of back ends available for transactions at the default.

  For example:
  
  ```
  -dm dm_n_be 48
  -dm dm_trans_be_max 46
  ```

  The example above allows you to have 23 account creation sessions active simultaneously.

- **Billing**: Because all billing operations are transactions, make sure there is at least one back end capable of handling transactions for each billing program thread, plus one additional back end for the master thread searches.

  For example:
  
  ```
  -dm dm_n_be 24
  -dm dm_trans_be_max 22
  ```

  The example above allows you to have approximately 20 billing sessions (children) active simultaneously.

In general, if you need rapid response times, reduce the number of transactions waiting to be processed by adding more back ends, devoting a larger number of them to transactions, or both. For example, try increasing the number of back ends to 3 to 4 times the number of application processes. For performance, dedicate at least 80% of the back ends to processing transactions. For heavy updating and inserting environments, especially when billing is running, dedicate all but two of the back ends to transaction processing.

For example:

  ```
  -dm dm_n_fe 4
  -dm dm_max_per_fe 16
  -dm dm_n_be 24
  -dm dm_trans_be_max 22
  ```
If you configure too many back ends, the DM process uses too much memory and there is too much context switching. Conversely, if you have too few back ends, the system performs poorly and the network is overloaded as terminal servers retry the connection.

**Note:** If there are not enough DM back ends, BRM may stop responding without reporting an error message.

On small BRM systems, where you might use a single DM for multiple activities, you can calculate the peak requirements for a combination of those activities and size the back ends accordingly. For example, you might need 32 connections for authentication and authorization and another 8 for the Web interface. If you run billing at hours when the rest of the system is relatively quiet, you do not need additional back ends.

**Note:** The number of back ends is independent of the number of front ends. That is, front ends are not tied to particular back ends because requests are transferred via the shared memory queue.

To help gauge the correct number of back ends, monitor database utilization. If it is under-utilized, you can increase the number of back ends.

### Determining the Maximum Number of Back Ends Dedicated to Transactions

The maximum number of back ends dedicated to transactions (specified in the `dm_trans_be_max` entry) should be at least 80% of the number of back ends specified in the `dm_n_be` entry. For heavy transaction loads, such as when running billing, use a value that is 2 less than the `dm_n_be` entry. For example:

- `dm dm_n_be 48`
- `dm dm_trans_be_max 46`

**Note:** You cannot specify more transaction back ends than there are total back ends.

### Setting the DM Time Interval between Opcode Requests

By default, the DM back-end processes wait an infinite amount of time for each opcode request, but you can set a time interval after which the DM back-end terminates if no opcode request has arrived. The following DM `pin.conf` entry specifies the maximum amount of time to wait, in minutes, for an opcode call before aborting the transaction:

- `dm dm_trans_timeout 4`

**Note:** To have DM back-end processes wait forever, set this entry to 0.

### Setting DM Shared Memory Size

BRM queuing increases system performance by lowering the number of connections to the database. This reduces the number of processes, which reduces the system load required to handle the connections. All DMs use shared memory for internal queuing.
Front-end processes pass connections through a shared-memory queue to back-end processes.

To specify DM shared memory, you use the following entries in the DM configuration file (`pin.conf`):

- **`dm_shmsize`**: Specifies the size of the shared memory segment, in bytes, that is shared between the front ends and back ends. The maximum allowed value of `dm_shmsize` in the DM’s `pin.conf` file is 2 GB.

- **`dm_bigsize`**: Specifies the size of shared memory for “big” shared memory structures, such as those used for large searches (with more than 128 results) or for `PIN_FLDT_BUF` fields larger than 4 KB.

The maximum allowed value of `dm_bigsize` in the Data Manager’s (DM’s) `pin.conf` file is now `1610612736` bytes (1.5 GB). The value of `dm_bigsize` should always be set less than the value of `dm_shmsize`.

To specify DM shared memory:

1. Open the DM configuration file (`BRM_Home/sys/dm_oracle/pin.conf`).
2. Change the configuration entry associated with each parameter. For tuning guidelines, see the discussions following this procedure. For the syntax of each configuration entry, follow the guidelines in the configuration file.
3. Save and close the file.
4. Stop and restart the DM. See "Starting and Stopping the BRM System".

**Note:** You may have to increase the `shmmax` kernel parameter for your system. It should be at least as large as the `dm_shmsize` entry in the DM configuration file on any computer running a DM. Otherwise, the DM will not be able to attach to all of the shared memory it might require and BRM will fail to process some transactions. See your vendor-specific system administration guide for information about how to tune the `shmmax` parameter.

**Note:** Besides configuring the number of connections for best performance, remember to keep the number of connections within the terms of your database license agreement.

### Determining DM Shared Memory Requirements

The amount of shared memory required by a DM depends on:

- **Number of front ends**: Each front end takes about 32 bytes of shared memory for its status block.

- **Number of connections per front end**: Each connection to a front end takes at least one 8-KB block of shared memory.

- **Number of back ends**: Each back end takes about 32 bytes of shared memory for its status block.

- **Size and type of DM operations**: Most of the shared memory used is taken by DM operations, and particularly by large searches; for example:
  
  - Running the `pin_ledger_report` utility.
Improving Data Manager and Queue Manager Performance

- Running searches that return large numbers or results.
- Using large zone maps. Allocate 1 MB of memory in the `dm_bigsize` entry for every 3000 lines in a zone map.

Operations that read objects, read fields, or write fields and involve a large BUF field can also be significant, but they are rare. Normal operations take 0 to 16 KB above the 8-KB-per-connection overhead.

You can also reduce the requirements for shared memory by using the PCM_OP_STEP_SEARCH opcode instead of the PCM_OP_SEARCH opcode.

You should monitor the shared memory usage and the transaction queues for each DM. See "Monitoring DM Shared Memory Usage" and "Monitoring DM Transaction Queues".

How BRM Allocates Shared Memory for Searches

The `dm_shmsize` entry sets the total size of the shared memory pool. The `dm_bigsize` entry sets the size of the portion of the shared memory reserved for “big” shared memory structures. Therefore, the memory available to front ends, back ends, and normal (not “big”) operations is the value of the `dm_shmsize` entry minus the value of the `dm_bigsize` entry.

For example, with these entries, the shared memory available to normal operations is 25165824:

- `dm dm_shmsize 33554432`
- `dm dm_bigsize 8388608`

To allocate memory for a search, BRM uses regular shared memory until the search returns more than 128 results. At that point, BRM reallocates the search to use the memory set aside for “big” structures. When allocating this type of memory, BRM doubles the size of the initial memory requirement in anticipation of increased memory need.

For example, consider a search that returns the POIDs of accounts that need billing. For 100,000 accounts, the memory allocated to the search is as follows:

- Memory used by “big” structures: 3.2 MB.
  The 3.2 MB figure is derived by taking the size of a POID and the anticipated number of accounts read in a billing application and then doubling the amount of memory as a safety margin.
  100,000 x 16 x 2 = 3,200,000 (3.2 MB), which is rounded up to a multiple of 8192. For example, `dm_bigsize` would be set to 3203072 or 391 x 8192.
  As a general rule, `dm_shmsize` should be approximately 4 to 6 times larger than `dm_bigsize`.

- Memory used by “small” structures: 4 MB.
  This memory is allocated for the following:
  - 2 MB for the result account POIDs (100,000 accounts x 20-byte chunks).
  - 2 MB for the POID types (100,000 accounts x 20-byte chunks).

- Total memory use: 7.2 MB.

Note: The value for `dm_shmsize` must be a multiple of 1024. The value of `dm_bigsize` must be a multiple of 8192.
Shared Memory Guidelines

DM shared memory is limited to 512 MB. Billing applications, internet telephony, and searching can affect DM shared memory requirements. It is usually best to start with a lower amount of shared memory to keep system resource usage minimal.

Shared memory for database servers can be from 512 MB for medium scale installations to several GB or more for the largest installations, depending upon activities. Some experimentation is necessary because more than 1 GB may not provide a performance increase, especially if there is a lot of update activity in the BRM database.

This example shows Solaris 2.6 kernel tuning parameters (for /etc/system) for the database server:

```plaintext
set bufhwm=2000
set autoup=600
set shmsys:shminfo_shmmax=0xffffffff
set shmsys:shminfo_shmseg=32
set semsys:seminfo_semmns=600
set semsys:seminfo_semmnu=600
set semsys:seminfo_semume=600
set semsys:seminfo_semmsl=100
forceload:drv/vxio
forceload:drv/vxspec
```

**Note:** This example of a Solaris kernel configuration essentially sets the maximum shared memory limit to infinity. When this setting is used, the system can allocate as much RAM as required for shared memory.

Reducing Resources Used for Search Queries

You can increase performance for search queries that retrieve objects with multiple rows from the database (for example, account searches for multiple customers) by setting the value of the `dm_in_batch_size` entry in the DM configuration file (`pin.conf`).

BRM interprets the value of `dm_in_batch_size` as the number of matching rows to retrieve in one search. When you start a search, BRM executes n+1 searches, where n is the number of searches performed to retrieve the number of rows set in `dm_in_batch_size`. For example, if `dm_in_batch_size` is set to 25 and the search retrieved 100 matching rows, five searches were performed (25 x 4)+1). The default setting is 80, indicating that BRM executes two searches to retrieve up to 80 matching rows. The maximum value is 160.

To preserve resources, you set the value in `dm_in_batch_size` to correlate to the size of the data set being searched. To increase performance when searching large data sets, you increase the number of retrieved rows in `dm_in_batch_size`. The larger the value set in `dm_in_batch_size`, the more resources are used to perform the search query. For example, if a typical user search query returns 10 rows from the database and `dm_in_batch_size` is set to 100, more resources than necessary are being used to complete the search.
Load Balancing DMs

The `dm_pointer` entry in the CM configuration file (`pin.conf`) tells the CM which DM to connect to. Having pointers to several DMs provides reliability because the system will switch to another DM if one DM fails.

You can ensure a more even load among the available DMs by adding several identical pointers to each DM, even if the DMs are on the same machine. When a CM receives a connection request, it chooses one of the pointers at random. Or, you can increase the load on a particular DM by increasing the relative number of pointers to that DM.

For example, if you have two DMs and you want to make sure that most activity goes to one with the most powerful hardware, make three or four pointers to that DM and only one or two to the other DM. When new child CM processes or threads are created, more of them are configured to point to the first DM:

```
- cm dm_pointer 0.0.0.1 ip 127.0.0.1 15950
- cm dm_pointer 0.0.0.1 ip 127.0.0.1 15950
- cm dm_pointer 0.0.0.1 ip 127.0.0.1 15950
- cm dm_pointer 0.0.0.3 ip 127.0.0.3 11950
```

Optimizing Memory Allocation during Database Searches

You can configure the Oracle DM to optimize memory allocation during database searches by using the `extra_search` entry in the DM configuration file. When this entry is set, the Oracle DM performs an extra search in the BRM database to calculate the number of database objects meeting the search criteria and then allocates the optimal amount of memory for the results.

To optimize memory allocation by performing an extra search:

1. Open the Oracle DM configuration file (`BRM_Home/sys/dm_oracle/pin.conf`).
2. Change the `extra_search` entry to 1:
   ```
   - dm extra_search 1
   ```
3. Save and close the file.
4. Stop and restart the Oracle DM. See "Starting and Stopping the BRM System".

Important: Performing the extra search slows database search performance.

Improving BRM Performance During Database Searches

Oracle databases could access tables that have non-bitmap indexes by performing an internal conversion from ROWIDs to bitmap and then from bitmap back to ROWIDs. This internal conversion process can significantly decrease BRM performance when querying a large number of rows.

To increase search performance, Oracle recommends that you prevent the database from using bitmap access paths for non-bitmap indexes. To do so, add the following parameter to your database’s `init.ora` file or `spfile` and then restart your database:

```
_b_tree_bitmap_plans=false
```
Improving Data Manager and Queue Manager Performance

Increasing DM CPU Usage

If the CPU usage on a DM machine reaches 75% over a 60-second average, increase the CPU capacity by using a faster CPU, adding CPUs, or adding another machine to run the same type of DM.

Examples of DM Configurations

These examples show DM pin.conf file settings used with a variety of multiple CPU configurations. These examples are intended as guidelines; your settings depend on your system resources and workload.

Example 1: BRM 16-CPU database server configuration

The example in Table 14–1 shows a BRM system that uses:

- A 16x450 MHz CPU database server.
- Four 6x450 MHz CPU CM/DM/EM systems.

<table>
<thead>
<tr>
<th>Daemon/program</th>
<th>pin.conf entry</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>dm_oracle</td>
<td>dm_n_fe</td>
<td>6</td>
</tr>
<tr>
<td>dm_oracle</td>
<td>dm_n_be</td>
<td>22</td>
</tr>
<tr>
<td>dm_oracle</td>
<td>dm_max_per_fe</td>
<td>16</td>
</tr>
<tr>
<td>dm_oracle</td>
<td>dm_trans_be_max</td>
<td>20</td>
</tr>
<tr>
<td>dm_oracle</td>
<td>dm_shmsize</td>
<td>67108864</td>
</tr>
<tr>
<td>dm_oracle</td>
<td>dm_bigsize</td>
<td>1048576</td>
</tr>
</tbody>
</table>

Note: The dm_shmsize entry is set to 64 MB to handle a larger billing load.

Example 2: BRM 36-CPU database server configuration

The example in Table 14–2 shows a BRM system that uses:

- A 36x336 MHz CPU database server.
- Four 4x400 MHz CPU CM/DM/EM systems.

<table>
<thead>
<tr>
<th>Daemon/program</th>
<th>pin.conf entry</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>dm_oracle</td>
<td>dm_n_fe</td>
<td>4</td>
</tr>
<tr>
<td>dm_oracle</td>
<td>dm_n_be</td>
<td>24</td>
</tr>
<tr>
<td>dm_oracle</td>
<td>dm_max_per_fe</td>
<td>16</td>
</tr>
<tr>
<td>dm_oracle</td>
<td>dm_trans_be_max</td>
<td>22</td>
</tr>
<tr>
<td>dm_oracle</td>
<td>dm_shmsize</td>
<td>20971520</td>
</tr>
<tr>
<td>dm_oracle</td>
<td>dm_bigsize</td>
<td>6291456</td>
</tr>
</tbody>
</table>
Improving Interprocess Communication (IPC) Performance

By default, CM and DM processes communicate through AF_INET sockets. You can increase your system’s interprocess communication (IPC) performance by configuring it to use AF_UNIX sockets between CMs and DMs that reside on the same machine and AF_INET sockets between CMs and DMs that reside on separate machines.

Both socket types are described in Table 14–3.

<table>
<thead>
<tr>
<th>Socket type</th>
<th>Description</th>
</tr>
</thead>
</table>
| AF_UNIX     | Provides communication through a local socket file that the DM creates each time it starts.  
**Note:** If the DM finds a socket file when it starts, it deletes the existing file and creates a new one.  
These sockets provide the fastest IPC performance but can be used only by processes located on the same machine. |
| AF_INET     | Provides communication through an IP address. These sockets are slower than AF_UNIX sockets, but they allow communication between processes on separate machines. |

To improve IPC performance by configuring your system to use both AF_UNIX and AF_INET sockets, perform these steps:

1. On machines containing both a CM and a DM, set the following entry in your CM configuration file (BRM_Home/sys/cm/pin.conf):
   - `cm dm_pointer 0.0.0.1 local BRM_Home/sys/dm/dm_port`

   If your CM also connects to DMs on other machines, such as in a multidatabase system, add the following entry:

   ```
   cm dm_pointer 0.0.0.x ip HostName PortNumber
   ```

   where
   - `HostName` is the associated DM machine’s host name or IP address.
   - `PortNumber` is the associated DM’s port number. The default port number is 12950.

   For an example of how to configure this file for multidatabase systems, see "Sample Configuration Settings for a Multidatabase System".

2. On machines containing both a CM and a DM, set the following entries in your DM configuration file (BRM_Home/sys/dm/pin.conf):
   - `dm dm_port PortNumber`
   - `dm dm_local_socket_name BRM_Home/sys/dm/dm_port`

   where
   - `PortNumber` is the DM’s port number. The default port number is 12950.

3. On machines containing a DM but no CM, set the following entry in your DM configuration file:
where PortNumber is the DM’s port number. The default number is 12950.

4. Save and close all files.

Sample Configuration Settings for a Multidatabase System

This section shows how to set your CM and DM configuration files for the following sample multidatabase system as shown in **Figure 14–4**:

**Figure 14–4  Sample Multidatabase System**

CM pin.conf file

- cm dm_pointer 0.0.0.1 local BRM_Home/sys/dm/dm_port
- cm dm_pointer 0.0.0.2 ip HostName2 PortNumber2
- cm dm_pointer 0.0.0.3 ip HostName3 PortNumber3

DM1 pin.conf

- dm dm_port PortNumber1
- dm dm_local_socket_name BRM_Home/sys/dm/dm_port

DM2 pin.conf

- dm dm_port PortNumber2

DM3 pin.conf

- dm dm_port PortNumber3
Improving Performance by Disabling Unused Features

You can improve performance by disabling features that you do not use. To do so, you edit `/config/business_params` objects. See `pin_bus_params` in BRM Developer’s Guide.

Table 14–4 lists the features enabled by `/config/business_params` objects that most heavily impact performance.

<table>
<thead>
<tr>
<th>Group</th>
<th>Name</th>
<th>Description</th>
<th>Default value</th>
</tr>
</thead>
<tbody>
<tr>
<td>activity</td>
<td>prepaid_traffic_light_enable</td>
<td>Enables and disables light-weight authorization. See &quot;Using Lightweight Authorization&quot; in BRM Telco Integration.</td>
<td>Disabled</td>
</tr>
<tr>
<td>invoicing</td>
<td>sub_ar_items_included</td>
<td>Indicates whether subordinate account invoices include A/R items. See &quot;About Invoicing for Hierarchical Account Groups&quot; in BRM Designing and Generating Invoices.</td>
<td>A/R items are included</td>
</tr>
<tr>
<td>billing</td>
<td>cache_residency_distinction</td>
<td>Enables and disables object cache residency distinction. See &quot;About Convergent BRM Systems&quot;.</td>
<td>Disabled</td>
</tr>
<tr>
<td>billing</td>
<td>enable_ara</td>
<td>Specifies whether Revenue Assurance is enabled for out-of-cycle billing. See &quot;Configuring Bill Now, On-Demand Billing, and Auto-Triggered Billing to Collect Revenue Assurance Data&quot; in BRM Collecting Revenue Assurance Data.</td>
<td>Disabled</td>
</tr>
<tr>
<td>billing</td>
<td>general_ledger_reporting</td>
<td>Enables and disables general ledger reporting. If disabled, the journal is not generated. See &quot;Enabling and Disabling General Ledger Collection in BRM&quot; in BRM Collecting General Ledger Data.</td>
<td>Enabled</td>
</tr>
<tr>
<td>rating</td>
<td>enable_eras</td>
<td>Specifies how to enable ERAs:</td>
<td>All service and account profiles</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ 0 = No profiles</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ 1 = Accounts only</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ 2 = Services only</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ 3 = All</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Filtering the ERAs Considered During Rating and Discounting&quot;.</td>
<td></td>
</tr>
<tr>
<td>rating</td>
<td>enable_global_charge_sharing</td>
<td>Enables and disables global charge sharing. See &quot;About Global Charge Sharing Groups&quot; in BRM Managing Accounts Receivable.</td>
<td>Disabled</td>
</tr>
<tr>
<td>rating</td>
<td>enable_tailormade_cache</td>
<td>Specifies if tailor-made products need to be maintained in the rating cache. When enabled, tailor-made products are stored in cache. See &quot;Enabling and Disabling the Caching of Customized Products&quot;.</td>
<td>Disabled</td>
</tr>
</tbody>
</table>
To tune billing performance, you edit entries in the Performance Entries section of the configuration file that is used by all billing utilities (`BRM_Home/apps/pin_billd/pin.conf`). You can set values for the following:

- In case of certain BRM search operations that return large amounts of data, you can filter and find accounts in resource sharing groups. See "Filtering Search Results".
- The number of child threads (or processes) that the billing utilities use. See "Tuning the Number of Children for Billing Utilities".
- The number of accounts to cache in system memory before BRM processes the accounts. For example, if you specify `50000`, BRM finds and caches 50,000 accounts before processing them. See "Tuning the Account Cache Size for Billing Utilities (fetch_size)".
- How many accounts the child threads can find and move into the cache at one time. This ensures that the DM memory is not overloaded. See "Tuning the Batch Size for Billing Utilities (per_batch)".
- The number of accounts sent in batches to the credit card processing service. See "Tuning the Batch Size for the pin_collect Utility".
- The number of retries in case a deadlock occurs. See "Specifying the Number of Retries in Case of a Deadlock".
- How accounts cached in system memory are distributed to child threads. See "Rearranging Accounts to Improve Billing Performance".

---

**Table 14–4 (Cont.) BRM Features that Heavily Impact Performance**

<table>
<thead>
<tr>
<th>Group</th>
<th>Name</th>
<th>Description</th>
<th>Default value</th>
</tr>
</thead>
</table>
| multi-bal  | lock_concurrency      | Indicates the concurrency of object locking. Possible values are:  
  - Normal (0): Locks the account object.  
  - High (1): More concurrency of locking with greater granularity of which balance group to lock. This setting is best for performance.  
  See "Disabling Granular Object Locking" in BRM Developer’s Guide. | High          |
| rerate     | batch_rating_pipeline | Specifies whether the batch-rating pipeline is active during rerating. See "Specifying Whether the Batch Rating Pipeline is Enabled" in BRM Setting Up Pricing and Rating.                                                      | Enabled       |
| rerate     | line_management       | Enables and disables service transfer. See "Managing Customers’ Subscription-Level Services" in BRM Managing Customers.                                                                                                         | Disabled      |
In the configuration file, entries preceded by `-pin_billd` apply to all billing utilities. Entries preceded by a specific utility name override the general entry for that utility. In this example, the `children` and `fetch_size` entries are used by all billing utilities, but the `pin_bill_accts` and `pin_collect` utilities have different values for the `per_batch` entry:

- `pin_billd children 5`
- `pin_billd fetch_size 10000`
- `pin_billd per_batch 2500`
- `pin_bill_accts per_batch 1000`
- `pin_collect per_batch 200`

The `children`, `fetch_size`, and `per_batch` entries are standard configuration options for multithreaded applications. For information about these entries, see “Configuring Your Multithreaded Application” in *BRM Developer’s Guide*.

See the comments in the configuration file (`BRM_Home/apps/pin_billd/pin.conf`) for more information.

For billing performance issues, see "Tuning Memory for Billing". For billing performance issues not related to the configuration file, see "Additional Issues Related to Billing Performance".

**Filtering Search Results**

Some BRM operations, such as searching for group members or searching for items to include on an invoice, can return large amounts of data and cause the Data Manager to fail. In this case, use the following configuration file entries to use a step search to find accounts in resource sharing groups:

- **group_members_fetch**: Use this entry to search for members of the group sharing object when the parent group contains many members. The parameter is specified as `- fm_bill group_members_fetch n`, where `n` is the size of pool (memory in bytes) for the step search. The default value is 0.

- **group_children_fetch**: Use this entry to search for child accounts in groups when the parent group contains many members. The parameter is specified as `- fm_bill group_children_fetch n`, where `n` is the size of pool (memory in bytes) for the step search. The default value is 0.

- **item_fetch_size**: Use this entry when searching for items. The parameter is specified as `- fm_bill item_fetch_size n`, where `n` is the size of pool (memory in bytes) for the step search. The default value is 0.

To filter search results:

1. Open the Connection Manager (CM) configuration file (`BRM_home/sys/cm/pin.conf`) in a text editor.
2. Uncomment or enter the following lines, as needed:
   - `- fm_bill group_members_fetch n`
   - `- fm_bill group_children_fetch n`
   - `- fm_bill item_fetch_size n`

   where `n` is the size of pool (memory in bytes) for the step search.
3. Save and close the file.
Tuning Billing Performance Using Configuration File

Tuning the Number of Children for Billing Utilities
By default, a billing utility uses five child threads to process accounts. You can increase
the number in the children entry to get better billing performance when the database
server remains under-utilized even though you have a large number of accounts. If
you increase the number of children beyond the optimum, performance suffers from
context switching. This is often indicated by higher system time with no increase in
throughput.
Billing performance is best when the number of children is nearly equal to the number
of DM back ends and most back ends are dedicated to processing transactions. For
information on adjusting the number of DM back ends, see "Configuring DM Front
Ends and Back Ends".

To tune the number of children:
1. Open the billing utilities configuration file (BRM_Home/apps/pin_billd/pin.conf).
2. In the Performance Entries section, edit the children entry:
   - pin_billd children 5
3. Save and close the file.

Tuning the Account Cache Size for Billing Utilities (fetch_size)
If there is enough memory, make the value in the fetch_size entry equal to or greater
than the number of accounts to be billed. If memory is limited, you can use a fetch_
size entry of 10,000 and a per_batch entry of 2500.

| Tip: For best performance, use a fetch_size value that is a multiple of
      | the per_batch value. |

To tune the account cache size:
1. Open the billing utilities configuration file (BRM_Home/apps/pin_billd/pin.conf).
2. In the Performance Entries section, edit the fetch_size entry:
   - pin_billd fetch_size 10000
3. Save and close the file.

Tuning the Batch Size for Billing Utilities (per_batch)
When processing a batch of accounts before moving them into the cache, the billing
utility waits for all children to finish processing before getting another batch of
accounts. Therefore, a single account that requires a long processing time can reduce
performance because the idle child threads need to wait for one child thread to finish
processing the account. To minimize this loss in productivity, you can increase the
batch size. If you increase the batch size too much, you will run out of memory.

To tune the batch size:
1. Open the billing utilities configuration file (BRM_Home/apps/pin_billd/pin.conf).
2. In the Performance Entries section, edit the per_batch entry:
   - pin_billd per_batch 2500
3. Save and close the file.
Tuning the Batch Size for the pin_collect Utility

The pin_collect utility uses the per_batch entry differently than other billing utilities use it:

- For most billing utilities, the per_batch entry specifies the number of accounts that BRM finds and caches in system memory.
- For the pin_collect utility, the per_batch entry specifies the number of accounts sent in batches to the credit card processing service.

For best performance, the value in the per_batch entry for the pin_collect utility should be much smaller than the per_batch entry for the rest of the billing utilities. Use a value that is equal to the per_batch entry that applies to the pin_bill_accts entry divided by the value of the children entry. (The pin_bill_accts value can be set either by a generic -pin_billd entry in the billing utility configuration file or by a specific -pin_bill_accts entry.)

For example, if the per_batch entry for pin_bill_accts is 3000, and you use 10 child threads, the per_batch entry for the pin_collect utility should be 300.

To tune the batch size for the pin_collect utility:

1. Open the billing utility configuration file (BRM_Home/apps/pin_billd/pin.conf).
2. In the Performance Entries section, add a per_batch entry for pin_collect:
   - pin_collect per_batch 500
3. Save and close the file.

Specifying the Number of Retries in Case of a Deadlock

For Oracle, you need to specify the number of retries to be attempted in case a deadlock occurs during a billing run. For information on the deadlock error, see "Reference Guide to BRM Error Codes".

To specify the deadlock retry number:

1. Open the billing configuration file (BRM_Home/apps/pin_billd/pin.conf).
2. Uncomment the -pin_bill_accts deadlock_retry_count entry.
3. Change the deadlock_retry_count entry, if necessary. The default is 20.
   - pin_bill_accts deadlock_retry_count 20
4. Save the file.

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

Rearranging Accounts to Improve Billing Performance

Billing utilities fetch accounts and cache them to system memory in the same sequence in which they are stored in the BRM database.

---

**Note:** The number of accounts fetched from the database is determined by the fetch_size entry in the billing utilities configuration file.
Each account in memory is then distributed to individual child threads (or processes) for billing. This behavior may slow billing performance because of database contention.

You can sometimes improve billing performance by rearranging accounts in system memory prior to distributing the accounts to child threads for processing by using the `delta_step` entry in the billing utility configuration file.

When a value is specified for this parameter, the billing utilities rearrange accounts cached in system memory based on the parameter value specified. Generally, Oracle retrieves the accounts in the order in which they are found in the database, grouped according to their physical location on the disk.

For example, we might have 100 accounts to bill and 10 threads, as well as 10 database blocks (A, B, C, D, E, F, G, H, I, and J) that each contain 10 accounts. The database returns a list that looks like this: A1, A2, A3, A4, A5, A6, A7, A8, A9, A10, and so on for each block. When BRM starts its threads, each of the 10 threads gets the next available account to process, and the mapping might look like this:

```
Thread 1 - A1
Thread 2 - A2
Thread 3 - A3
Thread 4 - A4
Thread 5 - A5
Thread 6 - A6
Thread 7 - A7
Thread 8 - A8
Thread 9 - A9
Thread 10 - A10
```

When a thread finishes processing an account, it takes the next available account from the list, and processing continues until all accounts have been processed. As a result, all of the threads at any given time may be accessing accounts in the same database blocks and vying for the same resources.

You can change the order in which these accounts are processed using the `delta_step` parameter. For example, to rearrange accounts by selecting and placing every tenth account cached in system memory, and then distribute these accounts to threads for billing, set this entry to 10:

```
pin_billd delta_step 10
```

The thread mapping, instead of proceeding one account at a time, would look something like this:

```
Thread 1 - A1
Thread 2 - B1
Thread 3 - C1
Thread 4 - D1
Thread 5 - E1
Thread 6 - F1
Thread 7 - G1
Thread 8 - H1
Thread 9 - I1
Thread 10 - J1
```

The `delta_step` parameter makes it possible for each thread to be working on data from a different area of the database, reducing contention for the same resources and improving billing performance.
You can determine the optimal setting for the `delta_step` parameter by testing the billing processes and monitoring their performance. By default, this parameter is set to 0, which means that the accounts cached in system memory are not rearranged before distribution.

To rearrange accounts in system memory:

1. Open the billing utility configuration file (`BRM_Home/apps/pin_billd/pin.conf`).
2. In the Performance Entries section, edit the `delta_step` entry:
   ```
   pin_billd delta_step 10
   ```
3. Save and close the file.

### Improving Performance in Retrieving Purchased Offerings For a Bill Unit

You can improve billing performance while retrieving purchased products and discounts for a bill unit (`/billinfo` object) from the database, by specifying the batch size of the number of services to search at a time in the `MaxServicesToSearch` parameter in the `/config/business_params` object for subscription.

To specify the batch size of the number of services to search:

1. Create an editable XML file for the `subscription` parameter instance by using the following command:
   ```
   pin_bus_params -r -c "Subscription" bus_params_subscription.xml
   ```
   This command creates the XML file `bus_params_subscription.xml.out` in your working directory. If you don’t want this file in your working directory, specify the full path as part of the file name.
2. Open the `bus_params_subscription.xml.out` file in a text editor.
3. In the `BusParamsSubscription` section, specify the batch size of the number of services to search by changing the value of the `MaxServicesToSearch` tag as follows:
   ```
   <BusParamsSubscription>
   ...
   <MaxServicesToSearch>5</MaxServicesToSearch>
   </BusParamsSubscription>
   ```

   **Note:** The default value is 5. The minimum allowed value is 1. The maximum allowed value depends on the length of the POIDs involved in the search and the search criteria. Depending on the length of the POIDs involved in the search, if you increase the value of `MaxServicesToSearch` and the length of the select statement is more than 2048 characters, the database select statement could fail.

4. Save the file as `bus_params_subscription.xml`.
5. Go to the `BRM_Home/sys/data/config` directory and load the change into the `/config/business_params` object by using the following command:
   ```
   pin_bus_params bus_params_subscription.xml
   ```
6. Read the object with the `testnap` utility or the Object Browser to verify that all fields are correct.
   See "Reading an Object and Fields" in *BRM Developer’s Guide*.

7. Stop and restart the Connection Manager (CM).
   See "Starting and Stopping the BRM System".

8. For multiple databases, run the `pin_multidb` script with the `-R CONFIG` parameter.
   For more information on this script, see `pin_multidb` in *BRM System Administrator’s Guide*.

---

**Tuning Memory for Billing**

If you change the `children` and `per_batch` entries for billing utilities, you should also review the `dm_shmsize` configuration entry for the DM. You can estimate the amount of shared memory required by using the following formulas:

- **To run billing (pin_bill_accts):** The amount of shared memory required, in kilobytes, is equal to 0.45 multiplied by the number specified in the `per_batch` entry.

- **To run credit-card collection (pin_collect):** The amount of shared memory required, in kilobytes, is equal to the number specified in the `children` entry multiplied by the number specified in the `per_batch` entry.

---

**Additional Issues Related to Billing Performance**

- To reduce system load, you can split large billing runs into smaller billing runs.
  See "Reducing Billing Run Loads" in *BRM Configuring and Running Billing*.

- You can improve performance of the `pin_bill_accts` utility by excluding accounts that do not need to be billed. For more information, see "About Suspending Billing of Accounts and Bills" in *BRM Configuring and Running Billing*.

- When you run the `pin_collect` utility, you can improve performance by archiving the temporary transmission logs created by the DM for the credit card processing service. See "Maintaining Transmission Logs for Billing Transactions" in *BRM Configuring and Running Billing*.

- You can create billing-related indexes just before you run billing and then delete them when billing finishes. You can add these tasks to the billing scripts. See "Editing the Billing Scripts" in *BRM Configuring and Running Billing* and "Removing Unused Indexes".

---

**The Effect of the Number of Events on Billing**

The number of events has no effect on billing or invoicing except in the following cases:

- If you defer tax calculations, billing performance is slower.
If you create detailed invoices, performance is slower. A telephone usage invoice is an example of a detailed invoice; a fixed-fee cable subscription invoice is an example of a non-detailed invoice.

### Tuning Billing Performance Using Business Parameters

You can improve the billing performance by controlling whether BRM searches for closed offerings and overridden products during the billing process.

#### Configuring to Exclude Searches on Closed Offerings

By default, BRM retrieves active, inactive, and closed offerings during the billing process. However, most of the time, BRM does not use the data from the closed offerings.

You can configure BRM to retrieve only the active and inactive offerings by disabling the `CancelledOfferingsSearch` parameter in the `subscription` instance of the `/config/business_params` object.

You modify the `/config/business_params` object by using the `pin_bus_params` utility. For information on this utility, see `pin_bus_params` in the `BRM Developer’s Guide`.

To disable searched on closed offerings:

1. Use the following command to create an editable XML file from the `subscription` instance of the `/config/business_params` object:

   ```
   pin_bus_params -r BusParamsSubscription bus_params_subscription.xml
   ```

   This command creates the XML file named `bus_params_subscription.xml.out` in your working directory. To place this file in a different directory, specify the full path name for the file. For more information on this utility, see `pin_bus_params` in the `BRM Developer’s Guide`.

2. Locate the `CancelledOfferingsSearch` entry in the `bus_params_subscription.xml.out` file.

3. Set the value of `CancelledOfferingsSearch` to `disabled`, if necessary:

   ```xml
   <CancelledOfferingsSearch>disabled</CancelledOfferingsSearch>
   ```

4. Save this updated file as `bus_params_subscription.xml`.

5. Load the modified XML file into the appropriate `/config/business_params` object in the BRM database.

   ```
   pin_bus_params bus_params_subscription.xml
   ```

   You should execute this command from the `BRM_Home/sys/data/config` directory, which includes support files used by the utility. To execute it from a different directory, see the description for `pin_bus_params` in the `BRM System Administrator’s Guide`.

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

   For more information on reading objects by using the Object Browser, see `BRM Managing Customers`. For instructions on using the `testnap` utility, see `BRM Developer’s Guide`.

7. Stop and restart the Connection Manager. See "Starting and Stopping the BRM System".
8. For multiple databases, run the `pin_multidb` script with the `-R CONFIG` parameter. For more information on this script, see `pin_multidb` in BRM System Administrator’s Guide.

Configuring to Exclude Searches on Overridden Products

By default, BRM retrieves the overridden products during the billing process. If you are not using tailor-made products, you can configure BRM to ignore and not retrieve the overridden products by disabling the `TailormadeProductsSearch` parameter in the `subscription` instance of the `/config/business_params` object.

You modify the `/config/business_params` object by using the `pin_bus_params` utility. For information on this utility, see `pin_bus_params`.

To disable searched on closed offerings:

1. Use the following command to create an editable XML file from the `subscription` instance of the `/config/business_params` object:

   ```bash
   pin_bus_params -r BusParamsSubscription bus_params_subscription.xml
   ```

   This command creates the XML file named `bus_params_subscription.xml.out` in your working directory. To place this file in a different directory, specify the full path name for the file. For more information on this utility, see `pin_bus_params` in the BRM Developer’s Guide.

2. Locate the `TailormadeProductsSearch` entry in the `bus_params_subscription.xml.out` file.

3. Set the value of `TailormadeProductsSearch` to `disabled`, if necessary:

   ```xml
   <TailormadeProductsSearch>disabled</TailormadeProductsSearch>
   ```

4. Save this updated file as `bus_params_subscription.xml`.

5. Load the modified XML file into the appropriate `/config/business_params` object in the BRM database.

   ```bash
   pin_bus_params bus_params_subscription.xml
   ```

   You should execute this command from the `BRM_Home/sys/data/config` directory, which includes support files used by the utility. To execute it from a different directory, see the description for `pin_bus_params` in the BRM System Administrator’s Guide.

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

   For more information on reading objects by using the Object Browser, see BRM Managing Customers. For instructions on using the `testnap` utility, see BRM Developer’s Guide.

7. Stop and restart the Connection Manager. See "Starting and Stopping the BRM System".

8. For multiple databases, run the `pin_multidb` script with the `-R CONFIG` parameter. For more information on this script, see `pin_multidb` in BRM System Administrator’s Guide.
Tuning Invoicing Performance

Invoice utilities are multithreaded applications (MTAs) and use a similar set of configuration entries as the billing utilities, including children, fetch_size, and per_step. For information about these entries, see "Configuring Your Multithreaded Application" in BRM Developer’s Guide.

Not all the invoice utilities use the entries in the same way, so you can configure them individually. To specify an entry for a particular utility, replace the generic name -pin_mta with the name of the specific utility. For example, you might have these two entries for fetch_size:

- pin_mta fetch_size 30000
- pin_inv_accts fetch_size 50000

See the comments in the configuration file (BRM_Home/apps/pin_inv/pin.conf) for more information.

Setting the Number of Children for Invoice Utilities

Because the invoice utilities work faster than the billing utilities, the number of children for invoicing can be up to 50% more than for billing or credit card processing.

You need to tune the DM configuration file dm_shmsize entry to handle the number of children. A typical value for the dm_shmsize entry is the size of an invoice (in bytes) multiplied by the number specified in the children entry.

To set the number of children for invoice utilities:

1. Open the invoice utilities configuration file (BRM_Home/apps/pin_inv/pin.conf).
2. In the Performance Parameters section, edit the children entry:
   
   - pin_mta children 2500
3. Save and close the file.

Tuning the Account Cache Size for Invoice Utilities (fetch_size)

If enough memory is available, set the value of the fetch_size entry to the number of accounts that need to be invoiced. The fetch_size value should be a multiple of the number specified in the per_step entry.

To change the account cache size for invoice utilities:

1. Open the invoice utilities configuration file (BRM_Home/apps/pin_inv/pin.conf).
2. In the Performance Parameters section, edit the fetch_size entry:
   
   - pin_mta fetch_size 4000
3. Save and close the file.

Setting the Batch Size for Invoice Utilities (per_step)

The recommended value for the per_step entry is 100 times the number specified in the children entry. Too high a value for the per_step entry can overload the DM memory.

To set the batch size for invoice utilities:

1. Open the invoice utilities configuration file (BRM_Home/apps/pin_inv/pin.conf).
2. In the Performance Parameters section, edit the `per_step` entry:
   - `pin_mta fetch_size 1000`

3. Save and close the file.

**Optimizing Invoicing Performance**

To improve invoicing performance, you can set the `inv_perf_features` flag in the Connection Manager's configuration file (`pin.conf`) to enable or disable specific optimizations during PCM_OP_IN_MAKE_INVOICE opcode processing.

   - `fm_inv inv_perf_features 0x00000000`

This flag is a bitmask and each bit position represents a performance optimization that can be turned on or off. By default, it is set to `0x00000000` (no optimizations are enabled).

Table 14–5 lists the bit values and their significance:

**Table 14–5  invoicing Performance Bit Values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00000001</td>
<td>Selects all event types. By default, only events that are configured in the /config/invoice_events object are selected.</td>
</tr>
<tr>
<td>0x00000002</td>
<td>Selects only the following event types:</td>
</tr>
<tr>
<td></td>
<td>• /event/billing/adjustment/account</td>
</tr>
<tr>
<td></td>
<td>• /event/billing/adjustment/event</td>
</tr>
<tr>
<td></td>
<td>• /event/billing/adjustment/item</td>
</tr>
<tr>
<td></td>
<td>• /event/billing/adjustment/tax_event</td>
</tr>
<tr>
<td></td>
<td>• /event/billing/cycle/tax</td>
</tr>
<tr>
<td></td>
<td>• /event/billing/payment/cash</td>
</tr>
<tr>
<td></td>
<td>• /event/billing/payment/cc</td>
</tr>
<tr>
<td></td>
<td>• /event/billing/payment/check</td>
</tr>
<tr>
<td></td>
<td>• /event/billing/payment/dd</td>
</tr>
<tr>
<td></td>
<td>• /event/billing/payment/payorder</td>
</tr>
<tr>
<td></td>
<td>• /event/billing/payment/postalorder</td>
</tr>
<tr>
<td></td>
<td>• /event/billing/payment/wtransfer</td>
</tr>
<tr>
<td></td>
<td>• /event/billing/product/fee/cancel</td>
</tr>
<tr>
<td></td>
<td>• /event/billing/product/fee/cycle</td>
</tr>
<tr>
<td></td>
<td>• /event/billing/product/fee/cycle/cycle_arrear</td>
</tr>
<tr>
<td></td>
<td>• /event/billing/product/fee/cycle/cycle_forward_bimonthly</td>
</tr>
<tr>
<td></td>
<td>• /event/billing/product/fee/cycle/cycle_forward_monthly</td>
</tr>
<tr>
<td></td>
<td>• /event/billing/product/fee/cycle/cycle_forward_quarterly</td>
</tr>
<tr>
<td></td>
<td>• /event/billing/product/fee/cycle/cycle_forward_semiannual</td>
</tr>
<tr>
<td></td>
<td>• /event/billing/product/fee/purchase</td>
</tr>
<tr>
<td></td>
<td>• /event/billing/refund/cash</td>
</tr>
<tr>
<td></td>
<td>• /event/billing/refund/cc</td>
</tr>
</tbody>
</table>
To enable multiple optimizations, you can OR the bits. For example, to select hard coded list of event types and to not write invoices to the database, set the flag to 0x00000006.

### Improving Data Processing Performance

A significant amount of time can be consumed in parsing SQL statements that read or write in the database. Adjusting an account, rating an event, and many other BRM activities require several steps to read and write data.

SQL statement-handle caching increases the speed at which statements are parsed.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
</table>
| 0x00000003 | - /event/billing/refund/check  
- /event/billing/refund/dd  
- /event/billing/refund/payorder  
- /event/billing/refund/postalorder  
- /event/billing/refund/wtransfer  
- /event/billing/reversal/cc  
- /event/billing/reversal/check  
- /event/billing/reversal/dd  
- /event/billing/reversal/payorder  
- /event/billing/reversal/postalorder  
- /event/billing/reversal/wtransfer  
- /event/billing/settlement/item  
- /event/billing/writeoff/account  
- /event/billing/writeoff/bill  
- /event/billing/writeoff/item  
- /event/billing/writeoff/tax_account  
- /event/billing/writeoff/tax_bill  
- /event/billing/writeoff/tax_item  
- /event/session/dialup |
| 0x00000004 | Creates the invoices but does not write them to the database. The bill object is updated with the invoice information. |
| 0x00000008 | Invoicing passes the input flist by reference. By default, invoicing creates a copy of the input flist. For large invoice input flists, setting this flag saves memory. |
| 0x00000010 | Events with no balance impacts are retained. If this flag is not set, events with no balance impacts are dropped from the final invoice. |
| 0x00000400 | Specifies to keep balance impacts for sponsored events in the member accounts. |

To enable multiple optimizations, you can OR the bits. For example, to select hard coded list of event types and to not write invoices to the database, set the flag to 0x00000006.
Managing Database Usage

How Statement-Handle Caching Works

An application sends an opcode through the CM to a DM, which maps each PCM operation to one or more dynamic SQL statements. For each such statement, the relational database management system (RDBMS)—such as Oracle—parses the statement, executes it, and fetches the results.

Oracle maintains a cache of the most frequent SQL queries. It uses soft parsing to shortcut its process of deciphering these statements, thus saving time in retrieving the requested data from the actual database. It then sends the data back through the DM to the application.

BRM generates and repeats a finite set of SQL statement forms. If the caching of statement handles is not enabled, the DM always parses the statement before each execution. With caching enabled, BRM maintains its most recently used statement handles within the DM, freeing the RDBMS from spending its time on soft parsing.

How to Use the Statement-Handle Cache

The stmt_cache_entries entry in the Oracle DM configuration file (BRM_Home/sys/dm_oracle/pin.conf) controls the statement-handle cache. The entry can be one of these two values:

- A value of 0 disables the cache.
- The default value of 1 means that the DM maintains 32 entries in each statement-handle cache for each back-end thread or process.

See the configuration file for more information.

The statement-handle caching performance feature requires a large value for the open_cursors setting in the initSID.ora database configuration file. See "Configuring Oracle Databases" in BRM Installation Guide.

Note: If your Oracle database and DM both reside on computers with extraordinarily large memory resources, you might be able to cache more statement handles. Consult with Oracle for advice before attempting to cache more statement handles. It can be dangerous to exceed 32 entries because two Oracle parameters need to be increased along with stmt_cache_entries to prevent system failure.

Managing Database Usage

Performance of your BRM system is affected by the number of events in the database. You can limit which types of events are recorded in the database, which saves space and improves performance.

It is essential that all ratable events (events with a balance impact) be recorded in the database. You do this by including them in the pin_event_map file (BRM_Home/sys/data/pricing/example) when you set up your price list. This is enforced by Pricing Center and the Price List Facilities Module (FM) opcodes.

However, many non-ratable events may not need to be recorded. For example, event objects that are recorded during account registration, deal purchase, and product purchase require no further updating.

BRM provides a utility and file for excluding events from being recorded in the database. The load_pin_event_record_map utility loads the pin_event_record_map
Removing Unused Indexes

file (BRM_Home/sys/data/config/pin_event_record_map), in which you specify the event types to exclude.

---

**Note:**

- By default, if an event type is not listed in the pin_event_record_map file, it is recorded.
- Event notification can still be performed based on excluded events because it is triggered even by events that are configured not to be recorded. See “Using Event Notification” in BRM Developer’s Guide.

---

Ratable events that are mapped in the pin_event_map file should not be added to the pin_event_record_map file. If you specify a ratable event type, the event record map is ignored when the ratable event occurs.

---

**Caution:** Excluding events from being recorded can cause applications that use the event data to return incorrect results. Make sure the events you exclude are not being used by any other application before you load the event record file.

---

To exclude events from being recorded:

1. Open the BRM_Home/sys/data/config/pin_event_record_map file.
2. List the events you want to exclude and set their record flag value to 0. For example, to not record folds that have no balance impacts, enter

   /event/billing/cycle/fold:   0

---

**Note:** The file includes the option to enable recording of listed events. You can use this option under special circumstances to record events that are normally not recorded.

---

3. Save and close the file.
4. Use the load_pin_event_record_map utility to load the file.
5. Verify that the file was loaded by using Object Browser or the robj command in the testmap utility to display the /config/event_record_map object. (See “Reading an Object and Writing Its Contents to a File” in BRM Developer’s Guide.)

---

Rebuilding Indexes

Indexes can become large and unbalanced, which reduces performance. To increase performance, rebuild the most heavily used indexes regularly. You can quickly rebuild indexes at any time; for example, you might want to rebuild some indexes before running billing. See “Rebuilding Indexes” in BRM Installation Guide.

---

Removing Unused Indexes

By default, BRM installation creates indexes for all features. However, if you do not use some features, you can delete their associated indexes.
See your database documentation for information about finding unused indexes. For example, on an Oracle database, turn on the Oracle tracing facility while BRM is running. This produces an output trace file, which you use as input to the Oracle TKPROF utility. The TKPROF utility creates a file that lists the access paths for each SQL command. These access paths include indexes.

**Tip:** You can also use the Oracle tracing facility to find missing indexes.

---

**BRM Account and Rating Performance Considerations**

Certain aspects of basic BRM functionality can affect performance; other aspects have no effect:

- The number of accounts does not affect performance.
- There is no performance difference when using different payment methods, such as invoice or credit card.
- BRM client applications, such as Customer Center, have little impact on system performance.
- Cycle events are rated faster than usage events.
- Performance decreases when accounts own a large number of products.
- Performance may decrease when the database contains a large number of ratable usage metrics (RUMs). In addition to computing the RUM for the specific event, BRM computes the RUMs for all base types of that event. For example, if you configured a RUM for the `/a/b/c/d` event, BRM also computes RUMs configured for the `/a/b/c` and `/a/b` events. You can increase performance by removing unused RUMs from your database.

---

**Tuning Multithreaded Workloads**

In most cases, increasing the number of threads that an application or process can use increases performance. However, too many threads can result in too much context switching between threads, which can decrease performance.

To determine the optimum number of threads, increase the number of threads and watch the CPU utilization. If adding threads increases system time, adding threads is not helping performance.

---

**CM and DM RAM and Swap Guidelines**

Each CM system should have at least 128 MB of RAM. Too much more than 256 MB might not add additional performance unless there are a large number of connections to the CM.

DM RAM can be lower than 512 MB for smaller installations. For larger installations with eight or more DM CPUs on a single SMP box, 512 MB to 1 GB is recommended. Heavy usage of certain business policies and large searches can greatly increase CM memory requirements.

A typical requirement for swap on a CM/DM system is two to three times the amount of RAM, but this depends upon the operating system and the number of processes running on the CM/DM system.
Hardware Guidelines

- Use large disk controller RAM size.
- Maximize the processor cache size on the database and DM servers. For best performance, use caches that are at least 1 MB.
- In most cases, performance is best when there are approximately one and a half times as many CPUs on the CM/DM systems as on the database server system.
- The total number of CM CPUs (regardless of the number of CMs) is approximately 25% - 33% of the total number of CPUs on the database server.
- CM and DM systems need less RAM than the database system.
- CM and DM systems have very low disk requirements. Disks are needed only for the operating system, BRM software, and swap space. Two 9-GB disks on each CM or DM system is usually enough, as long as these systems are not being used for non-BRM workloads.

Two disks are recommended so that access to temp space, the operating system, swap, and BRM code are not bottlenecked on a single disk.

Improving Network Performance

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

The network bandwidth needs are relatively simple. BRM has an OLTP footprint (as opposed to Decision Support). This means that transactions are normally small and network traffic will consist of smaller packets. The only requirement for network connectivity is TCP/IP. The real constraint is to have enough bandwidth between the DM systems and the database server; otherwise, the network may become a bottleneck even under moderate loads.

- Use 100BaseT or FDDI between each DM system and the database server. 10 Mbit Ethernets have too little bandwidth to handle heavy DM-to-database-server traffic. To minimize any collisions, the DM-to-database connections should use a separate physical connection (or a switch) and each of these should be a separate network.

- For switch-connected systems, connect all systems in the BRM configuration by using a single switch. For the largest configurations, use gigabit connections.

- When the test systems have multiple network cards, verify that the operating system network routing tables are configured to avoid bottlenecks. By default (depending on the system), output from the database server system may go through one LAN card, even though you have several configured, and even though input comes in through the multiple cards from different DMs. Examine and fix the routing tables as necessary. Make sure that each DM-to-database-server connection is explicit so all traffic between the two machines goes through the single dedicated physical path. The most common environment where you might find this problem is when there are multiple paths between the same sets of systems.

- For best performance, make sure all the systems in the testing environment are connected to the same hub with no intermediate hops.

Troubleshooting Poor Performance

When troubleshooting poor performance, first consider the following:
Troubleshooting Poor Performance

- Under-configured hardware.
- Inefficient table layout.
- Database bottlenecks.
- Inefficient custom application code.
- Repeated runtime errors resulting from configuration problems.

In addition, you can look for different problems depending on whether CPU utilization is high or low.

Low Performance with High CPU Utilization

If performance is low and CPU utilization is high, or if there are performance spikes, there is probably a configuration or indexing issue. Check the following:

- Hardware limitations.
- Table/volume layout.
- Spin count is too high.
- Lack of proper indexes. This can show up as very high CPU utilization with no other apparent problems except for a high number of processes. Find which columns are being accessed in the operation being performed and make sure that they are properly indexed.
- Not enough database buffers.
- Swapping.
- Kernel parameters too low.

Low Performance with Low CPU Utilization

If performance is low and CPU utilization is low, check for a bottleneck between different system tiers; for example, between the DM and the database.

- Use the database monitoring tools to analyze the performance of the database system.
- Use SQL tracing and timing to check for inefficient application code.
- Check for an under-configured BRM system, which could be one of the following:
  - CM Proxy with a low number of children.
  - RADIUS Manager with a low number of threads.
  - DMs with a low number of back ends.
  - System logging level is too high.

Monitor the DM system utilization and Oracle system utilization and tune the number of DM back ends accordingly. A good starting point for DM back-end numbers is eight times the number of processors.

For more information, see "Improving Data Manager and Queue Manager Performance".
Quick Troubleshooting Steps

- Run quick timing tests by using the `testnap` utility with `op_timing` turned on to ping each CM and DM (with the PCM_OP_TEST_LOOPBACK opcode). If the operations are relatively slow, it indicates a problem in the basic configuration.
- Run the system with a log level of DEBUG on the CM and DM and analyze log files.
- Check for network collisions and usage data.
- Check if you have logging (debugging) turned on in the CM. Logging is good for troubleshooting, but it should not be turned on in a production environment because it reduces performance.
- Performance parameters in `pin.conf` files should be large enough to handle the load. The most likely problems are in the DM entries.
- Check if you have enough DM back ends to handle your transaction load.
- Try putting tables and indexes on different disks.
- Check the size of redo and rollback logs and database configuration parameters.
- Send a few `kill -USR1` commands to the DMs and CMs that seem to be having problems. This causes them to dump their state to the BRM error log files. Snapshots should be up to 20 minutes apart. These log files may contain information that indicates the nature of the problem.
- Turn on SQL tracing and analyze query plans. Look for full table scans. Make sure that indexes are on the appropriate columns for the query being run. Especially verify for any customizations.
- Turn on the `timed_statistics` parameter. Look for unusually long execution times for SQL commands.
- Monitor hardware activity:
  - On HP-UX IA64, AIX, Linux, and Solaris systems, use `vmstat, netstat, and sar`.
  - HP-UX IA64 has an additional tool called `Glance`.
  - Drill down to the storage device level by using `sar` with the `-d` parameter. This should help you find the source of the problem.

**Note:** If the file systems are configured from logical volumes that are comprised of physical disks, different file systems could be sharing the same underlying disk. It is important to unravel who owns what in order to isolate potential contention (waiting on I/O).

- Problems such as intermittent daemon failures can be indicated by core files. Try the following command to locate them:
  ```
  % find BRM_Home -name core -exec file {} 
  %
  ```
  If there are no core files, try turning on maximal debugging. You do not want to do this for very long, especially on a production system, because the log files fill up rapidly.
  ```
  % pin_ctl stop cm
  % setenv CMAP_DEBUG to 0x1331f3
  % setenv CM_DEBUG to 0x0001
  % setenv cm_loglevel to 3
  ```
About Benchmarking

To determine the best possible performance for your system, you need to identify the desired transaction capacity at each tier and ensure that the system handles several times that capacity. The maximum capacity threshold or the transaction capacity threshold can be determined by running benchmark scenarios.

The primary goal is to achieve full utilization of the database system. This is best accomplished by measuring system performance as the following operations are carried out:

- Increase the load to get maximum throughput.
- Increase the number of DM back ends to get maximum RDBMS utilization for a given workload.
- Increase database utilization for a given number of DM back ends.
- Slowly reduce the load to keep the same performance but with faster response time.
- Multiple iterations of the above steps.

The general process for benchmarking is:

1. Create a workload on the system. The best choice to do this is by running your own program. The second best choice is to use the ITM-C workloads.

2. Measure the results. Most programs need a ramp-up period of 200 seconds to reach a steady-state condition during which actual measurements should take place. Running the program for 10 to 20 minutes should produce the same results as running the program for hours, with the exception of the amount of disk space used. If you run the system for a long time, indexes might become imbalanced and need to be rebuilt, especially before billing.

Use monitoring tools for the systems on which BRM is running to determine system load and identify performance issues. Be sure to turn monitoring tools off for your final test runs. When you start the system, turn on level 3 debugging in all BRM processes and make sure that there are no error messages while running the benchmark programs. When there are no more errors, turn off logging.

3. Monitor the hardware, operating system, and BRM utilization.

BRM Performance Diagnosis Checklist

When troubleshooting performance, use this checklist to help you look for problems. For more information, see:

- Monitoring and Maintaining Your BRM System
- Resolving Problems in Your BRM System

You can also use this checklist to gather information that Support needs when diagnosing trouble tickets. If you submit a performance issue to technical support, you should also include the following:
■ All applicable error log files; for example, log files (or portions of log files) for the CM, DM, and client applications.

■ Operating system settings such as maximum shared memory segment size and number of processes. Provide a full list:
  – Solaris: /etc/system
  – HP-UX IA64: /stand/system
  – Linux: /etc/sysctl.conf
  – AIX: Enter the following command to get a list of all parameters:
    ```bash
    lsattr -E -l sys0
    ```

■ Administrative tools for managing systems:
  – HP-UX IA64: sam
  – Solaris: Admintool
  – Linux: Webmin and Easilix
  – AIX: smit

■ The pin.conf files for CMs, DMs, and clients such as RADIUS Manager.

■ For Oracle, the init.ora file.

Describe the Problem

■ What part of the system is experiencing the problem?

■ What operation or application is running; for example, billing or credit card processing?

■ What is the actual and expected performance?

■ What appears to be the problem?

■ What are the error messages?

Describe the Configuration

Hardware Configuration
For each system in the configuration:

■ What is the manufacturer, model, number, and types of CPUs, and amount of RAM?

■ What is the swap size?

For the database server system:

■ What is the RAID level?

■ How many disks, and what is their size?

■ How are logical volumes configured?

Operating System Configuration

■ What is the operating system version?
What are the operating system settings for maximum shared memory segment size, number of processes, and so forth?

Which patches have been applied?

**BRM Configuration**

- Which release of BRM are you using?
- Which systems do the following components run on? Which systems have multiple components, and which components run on multiple systems? How are the pin.conf files configured?
  - CMMP
  - CM Proxy
  - CM
  - DM
  - Optional managers such as RADIUS Manager
  - BRM client applications
  - Custom applications
  - Billing utilities
- Which BRM operations are slow?
- Which PCM_OPs are those slow operations associated with? (This can be found by using log level 3.)
- What is the estimated number of accounts in the database?
- What is the average number of products per account?
- What is the largest quantity of products owned by one account?
- What percentage of accounts use which payment method; for example, credit card or invoice?
- What is the estimated number of events in database?

**Network Configuration**

- How are the systems connected; for example, 10BaseT or 100BaseT?
- Are separate networks used for each DM database connection?

**Database Server Configuration**

- What are the index and data file sizes?
- What are the database hot spots?
- What is the disk layout?
- What is the assignment of tablespaces to logical devices?
- Are disk volumes used?
- Are redo logs on their own disk?

**Oracle Configuration**

- What is the Oracle version?
- How is the init.ora file configured?
  The following init.ora Oracle parameters are particularly important.
  - db_block_buffers
  - shared_pool_size
  - use_aysnc_io
  - db_block_size
  - max rollback_segments
  - processes
  - dml_locks
  - log_buffer
  Compare how your parameters are configured to those in the example BRM performance configurations.
  - Does the SGA roughly equal half the physical RAM?
  - What are the sizes and number of rollbacks?
  - Is check-pointing or archiving enabled?
  - Index and table fragmentation?
  - Number of extents; next extent size?
  - Run the query select index_name from user_indexes to view indexes. Check the indexes vs. columns in the WHERE clause.
  - Which optimizer option is being used (CHOOSE or RULE)?

**Describe the Activity**

- Are there any messages in any error logs (CM, DM, application)?
- Are there any operating system or database system error messages?
- Are there any bad blocks?
- Are you using any nonstandard resources, custom code (especially in the CM), or debugging aids such as writing log records to files that might result in contention or bottlenecks?
- Is there enough free swap space?
- What is the CPU utilization on servers used for BRM processes?
- Database system:
  - What are I/Os per disk per second; size of disk queues, disk service time, and percent of time waiting for I/O?
  - What is the CPU utilization on the database system?
This document describes tools and techniques you can use to optimize Oracle Communications Billing and Revenue Management (BRM) Pipeline Manager performance.

Before reading this chapter, you should be familiar with the following topics:

- Configuring Pipeline Manager
- "About Pipeline Rating" in BRM Configuring Pipeline Rating and Discounting

**Pipeline Manager Optimization Overview**

When you optimize Pipeline Manager performance, your objective is to increase the percentage of CPU time spent on user processes and to decrease the percentage of time spent idle or on system processes.

Complete performance tuning requires much testing. Due to the complexity of most Pipeline Manager configurations, optimization is a highly iterative process. You cannot configure options formulaically, but you must test many configurations and then implement the optimal configuration. This document describes optimization methods to guide your testing for a given set of hardware resources.

Software optimization techniques can include modifying the following:

- The number and type of function modules.
- The design of custom iScripts and iRules.
- The number of system threads used by a pipeline.
- The number of call data record (CDR) files configured for a transaction.
- The number of pipelines configured for the system.

**Note:** Available hardware resources can constrain the usefulness of some optimization techniques. For example, if your system has only a few CPUs, you probably will not see performance gains by using multithreaded mode.

**Key Metrics for Measuring Performance**

When evaluating performance improvement, the primary metrics to monitor are:

- The ratio of CPU time spent on system processes to CPU time spent on user processes. This ratio should be about 1 to 2 or lower.
- The percentage of idle CPU time. This percentage should be 20 percent or less.
- The results of performance tests using sample CDR files.

### About Measuring Pipeline Manager Performance

You use the Pipeline Manager instrumentation feature as the primary tool for measuring Pipeline Manager performance. See "Measuring System Latencies with Instrumentation" for more information. When instrumentation is enabled, information about how much time in microseconds each function module uses to process a certain number of files is written to the pipeline log file (`pipeline.log`). You then use this information when you apply some optimization techniques.

**Note:** For more information on the pipeline log, see “LOG” in BRM Configuring Pipeline Rating and Discounting.

Other Pipeline Manager performance monitoring tools are:

- Monitor event data record (EDR) throughput. See "Monitoring Pipeline Manager EDR Throughput".
- Monitor recent log files. See "Getting Recent Pipeline Log File Entries".
- Monitor memory usage. See “Memory Monitor” in BRM Configuring Pipeline Rating and Discounting.

### Information Requirements

Before you optimize Pipeline Manager, be familiar with your existing system configuration, such as:

- Total system memory.
- Other (non-pipeline) processes running on the Pipeline Manager system that will share system resources.
- The number and types of pipelines required for your business logic or planned load balancing.
- The expected load for each pipeline.
- Whether your business logic is more CPU intensive or I/O intensive. (For example, if you use the FCT_Discount module, your business logic is likely to be more CPU intensive.)

### Testing Requirements

Before you optimize Pipeline Manager, you should have a set of error-free sample CDRs that resemble those used in your production system.

### Optimizing Pipeline Manager

To optimize Pipeline Manager, consider the following actions:

(Oracle Solaris, Linux, HP-UX IA64, and AIX) Be sure that OS-specific system configurations were put in place during installation. See the following topics in BRM Installation Guide:

- Solaris
Creating a User and Configuring Environment Variables

(Solaris) Setting Maximum Open Files on Solaris

(Solaris) Configuring Memory Allocation and Block Transfer Mode on Solaris Systems

Linux

Creating a User and Configuring Environment Variables

(Linux) Setting Maximum Open Files on Linux

HP-UX IA64

Creating a User and Configuring Environment Variables

(HP-UX IA64) Setting Maximum Open Files on HP-UX IA64

---

**Important:** For HP-UX IA64, you must set the `_M_ARENA_OPTS` and `_M_CACHE_OPTS` environment variables to achieve acceptable system performance.

---

AIX

Creating a User and Configuring Environment Variables

(AIX) Setting Maximum Open Files on AIX

- Configure pipelines to run in either single-threaded or multithreaded mode. See "Configuring Single-Threaded or Multithreaded Operation". It is especially important to maximize the performance of the DAT_AccountBatch and DAT_BalanceBatch modules. See:
  - Configuring the DAT_AccountBatch Module Database Connections and Threads
  - Configuring Threads for DAT_BalanceBatch Connections
- Configure function pools within pipelines. See "Optimizing a Pipeline by Using Function Pools".
- If you have CDR files smaller than a few thousand records, consider grouping multiple CDR files into one transaction. See "Combining Multiple CDR Files into One Transaction".
- Configure multithreading in the Output Controller. See "Increasing Pipeline Manager Throughput When an EDR Is Associated with Multiple Output Streams" for more information.
- Add additional pipelines. See "Configuring Multiple Pipelines".
- Verify that any custom iScripts and iRules are efficiently designed. See "Optimizing Function Modules".
- Configure event and service mapping to only supply the Pipeline Rating Engine with the services being rated. See "Mapping Events and Services" in BRM Setting Up Pricing and Rating.
- Configure the DAT_USC_Map module to improve startup performance. See "Configuring the DAT_USC_Map Module for Startup Performance".
**Troubleshooting Pipeline Performance**

Use this checklist to troubleshoot drops in performance.

- If you installed a patch, find out if the patch changed operating system functions such as threading or memory management, or made any changes to Pipeline Manager framework modules.
- Check recent customizations, such as iScripts. Look for customizations that might impact database access or hash usage.
- Use database monitoring tools to monitor the Pipeline Manager database to see if there is a lot of activity. If so, check which queries are used and which indexes are used. This might point to the data involved, which might point to the module processing that data.
- Use a monitoring command such as `iostat` to check I/O activity.
- Use a memory monitoring command such as `prstat`, `vmstat`, or `sar` to check if the Pipeline Manager memory usage has changed. If Pipeline Manager uses an unexpected amount of memory, check for duplicate keys related to buffers and call assembly.
- Check for large numbers of files in the following directories:
  - in
  - err
  - done
  - dupl
  - assembl
  - rej
  Delete old files that are no longer needed.
- Look for bottlenecks in the pipeline by using the `prstat` command and the thread ID in the `process.log` file to identify slow threads. Check for:
  - icx (involuntary context switch)
  - vcx (voluntary context switch)
  - scl (system call)
  - slp (sleep)
- Check the `pipeline.log` file for records of a large amount of rollbacks.

**Optimizing Function Modules**

Slow function modules can be very detrimental to overall Pipeline Manager performance. To optimize individual function modules:

1. Identify the high latency modules by using instrumentation. See "Measuring System Latencies with Instrumentation".
2. Check if the high latency modules can be optimized. For example, you might discover that the business logic used in high latency iScripts or iRules can be redesigned to improve performance.
Configuring Single-Threaded or Multithreaded Operation

You configure pipelines to run in single-threaded or multithreaded mode by using the `MultiThreaded` registry entry in the registry file.

- **Single-threaded mode**: Use this mode if you are using a system with just a few CPUs and limited RAM.
  
  In a single-threaded environment, pipelines use a single thread to run all modules and only one CPU is used for each pipeline.
  
  If the `MultiThreaded` registry entry is not included in the registry file, pipelines will by default run in multithreaded mode.

  **Note**: Business logic can prevent the setup of multiple pipelines.

- **Multithreaded mode**: Use this mode if your system has many CPUs.
  
  In a multithreaded environment, pipelines use three or more threads to process each transaction. By default, one thread is used for the input module and one for the output module. An additional thread is used for each function pool that you configure to process function modules.

  For information on optimizing pipelines when using multithreaded mode, see:
  - Assigning Multiple Threads to Process Function Modules
  - Optimizing a Pipeline by Using Function Pools

  For information about the `MultiThreaded` registry entry, see "Pipeline Controller" in BRM Configuring Pipeline Rating and Discounting.

To configure single-threaded or multithreaded operation:

1. Open the registry file in a text editor.
2. Set the input controller `MultiThreaded` registry entry to the appropriate value:
   - `True` to configure the pipeline for multithreaded processing.
   - `False` to configure the pipeline for single-threaded processing.

   Pipelines
   
   ```
   {  
     PipelineName
     {  
       MultiThreaded = value
       . . .
     }
   }
   ```

3. Restart the pipeline. See "Starting and Stopping Individual Pipelines".

**Note**: For more information on all registry entries pertaining to individual pipelines, see "Pipeline Controller" in BRM Configuring Pipeline Rating and Discounting.

Reducing Startup Times with Parallel Loading

You can reduce your startup times by configuring Pipeline Manager to:

- Load all pipelines in parallel.
Configuring Single-Threaded or Multithreaded Operation

- Load data modules in parallel.
- Load function modules in parallel.

By default, Pipeline Manager loads pipelines, data modules, and function modules sequentially.

To enable parallel loading, use the Parallel Load Manager module:

1. Open the registry file in a text editor.
2. Configure the `ifw.ParallelLoadManager` section of the registry file:
   - Set the `Active` registry entry to `True`.
   - Set the `NumberOfThreads` registry entry to the number of threads you want Pipeline Manager to use for loading your pipelines, data modules, and function modules.

   For example:
   ```
   ifw
   {
     ...
     ParallelLoadManager
     {
       Active = True
       NumberOfThreads = 4
     }
     ...
   }
   ```

3. Restart the pipeline. See "Starting and Stopping Individual Pipelines".

Assigning Multiple Threads to Process Function Modules

If a pipeline is configured for multithreaded processing and you have idle CPU resources, you might be able to increase performance by grouping function modules into two or more function pools. The pipeline runs each function pool in a separate thread.

---

**Important**: Adding too many function pools to a pipeline can decrease performance because the buffers between the threads consume system CPU overhead and RAM. (Typically, two to six function pools is optimal.)

---

**Tip**: If you are using a high-latency module such as FCT_AccountBatch or FCT_Discount and have sufficient hardware resources, assign the module to its own function pool and test for performance improvement.

To create a separate thread for an individual function module or a group of function modules, you use the `FunctionPool` registry entry.

---

**Important**: Before you perform this procedure, read "Optimizing a Pipeline by Using Function Pools".
1. Submit some sample CDRs to the pipeline with instrumentation enabled. See "Measuring System Latencies with Instrumentation".

2. Locate the instrumentation results in the pipeline.log file.

3. Open the registry file in a text editor.

4. Using the instrumentation data, reduce the processing time required by the slowest function pool by:
   - (Optional) Adding an additional function pool to the Functions section of the registry file.
   - Shifting one or more modules from a function pool to an adjacent function pool.

   The objective is to make the processing times of all function pools as similar as possible.

5. Save the registry file.

6. Restart the pipeline. See "Starting and Stopping Individual Pipelines".

7. Measure pipeline performance with the sample CDRs by measuring transaction start times and end times in the pipeline.log file.

8. Go to Step 3 and repeat testing until optimal results are achieved.

---

**Configuring the DAT_AccountBatch Module Database Connections and Threads**

To improve performance, you can configure multiple DAT_AccountBatch connections to the BRM database. Configure the following registry entries:

- Use the Threads registry entry to specify the number of threads. Set this value to at least the number of CPUs in the system. Increasing the number of threads increases performance, up to a point. Specifying too many threads decreases performance.
  
  The default is 4.

- Use the Connections registry entry to specify the number of connections to the database. This value must be at least one more than the number of threads.
  
  The default is 5.

- Use the LoadPercentage registry entry to specify the percentage of account POIDs to store locally when determining the account blocks for which each thread is responsible.
  
  Values must be greater than 0.000000 and less than or equal to 100.0.
  
  The default is 10.

---

**Setting the Hash Map Size for Threads**

You can use the following DAT_AccountBatch registry entries to set the temporary hash map size built for each thread. Each entry controls the hash map size for a different type of data; for example, accounts, logins, and services.

In general, larger maps perform better but consume more memory. Smaller maps save memory but can slow down Pipeline Manager startup. Very low numbers can dramatically slow down Pipeline Manager startup.

The default system-calculated value uses this formula:

$$((\text{number of accounts}/\text{number of threads}) \times 2).$$
The registry entries are:

- **ThreadAccountHashMapSize**: Used for account data.

  __Important__: Changing the default system-calculated values for this entry is not recommended. Replacing this entry with one larger than the default wastes memory. Replacing this entry with one smaller than the default slows the Pipeline Manager startup.

- **ThreadGroupSharingChargesHashMapSize**: Used for charge sharing group data. The system-calculated default value might not be appropriate.

  If your accounts average fewer than two or more than four GroupSharingCharges per account, use the following formula as a guideline to calculate an entry:

  \[
  \left(\frac{\text{number of accounts} \times \text{average number of GroupSharingCharges per account}}{\text{number of threads}}\right) \times 75\%.
  \]

- **ThreadGroupSharingDiscountsHashMapSize**: Used for discount sharing group data. The system-calculated default value might not be appropriate.

  If your accounts average fewer than two or more than four GroupSharingDiscounts per account, use the following formula as a guideline to calculate an entry:

  \[
  \left(\frac{\text{number of accounts} \times \text{average number of GroupSharingDiscounts per account}}{\text{number of threads}}\right) \times 75\%.
  \]

- **ThreadGroupSharingProfilesHashMapSizes**: Used for profile sharing group data. The system-calculated default value might not be appropriate.

  If your accounts average fewer than two or more than four profile sharing groups per account, use the following formula as a guideline to calculate an entry:

  \[
  \left(\frac{\text{number of accounts} \times \text{average number of GroupSharingProfiles per account}}{\text{number of threads}}\right) \times 75\%.
  \]

- **ThreadLoginHashMapSize**: Used for login data. The system-calculated default value is appropriate for most implementations.

  If your accounts average more than four logins per account, use the following formula as a guideline to calculate an entry:

  \[
  \left(\frac{\text{number of accounts} \times \text{average number of logins per account}}{\text{number of threads}}\right) \times 75\%.
  \]

- **ThreadServiceHashMapSize**: Used for service data. The system-calculated default value is appropriate for most implementations.

  If your accounts average more than four services per account, use the following formula as a guideline to calculate an entry:

  \[
  \left(\frac{\text{number of accounts} \times \text{average number of services per account}}{\text{number of threads}}\right) \times 75\%.
  \]

### Locking Objects During DAT_AccountBatch Processing

You can set the number of pre-allocated mutex objects that are used to lock individual objects during processing to prevent multiple threads from contending for access to the same object. You can use different settings for account, login, and service objects by setting the following DAT_AccountBatch registry entries:
Improving Pipeline Manager Startup Performance

- AccountLocks
- LoginLocks
- ServiceLocks

The default value for these entries should be appropriate in most cases. If you use a larger value, less allocation is needed for additional mutex objects during processing, but more memory is used.

The default for all entries is 10.

Configuring Threads for DAT_BalanceBatch Connections

Use the following DAT_BalanceBatch registry entry to configure connections to the BRM database:

- **Threads**: Specifies the number of threads for loading the balance data from the BRM database. The number of threads must be smaller than or equal to the number of connections.
  
  The default is 4.

- **ThreadHashMapSize**: Specifies the size of the hash map in each thread used for loading balance data from the BRM database.
  
  The default is 1024.

Improving Pipeline Manager Startup Performance

For information about improving Pipeline Manager startup performance, see:

- Improving DAT_BalanceBatch Loading Performance
- Improving DAT_AccountBatch and DAT_BalanceBatch Load Balancing

Improving DAT_BalanceBatch Loading Performance

The DAT_BalanceBatch module uses the non-currency resource validity to select the non-currency sub-balances to load from the BRM database into pipeline memory. If the non-currency resource validity is not configured, at Pipeline Manager startup, DAT_BalanceBatch selects the sub-balances that were valid for 366 days by default. When the BRM database contains a large number of non-currency sub-balances, loading them leads to increased Pipeline Manager startup times.

To improve Pipeline Manager startup performance, you can set the non-currency resource validity to specify the sub-balances to load. See "Specifying Which Non-Currency Sub-Balances to Load on Startup" in BRM Setting Up Pricing and Rating for more information.

Improving DAT_AccountBatch and DAT_BalanceBatch Load Balancing

The DAT_AccountBatch and DAT_BalanceBatch modules use multithreaded framework to load account and balance data from the BRM database into Pipeline Manager memory. The modules group the accounts and balances into batches or jobs. Multiple worker threads run in parallel to process the jobs. When a thread completes processing, it is assigned another job from the jobs pool, which improves load balancing between the threads and increases Pipeline Manager startup performance.

By default, the number of jobs per thread is 3, which is appropriate in most installations to achieve load balancing. However, if thread loading times vary greatly,
you can use the **PerThreadJobsCount** entry in the DAT_AccountBatch registry and the **BalancesPerThreadJobsCount** entry in the DAT_BalanceBatch registry to adjust the number of jobs per thread.

---

**Important:** Setting the number of jobs per thread to a large number can outweigh the performance gain because of the system overhead associated with creating too many jobs. (Typically, three to eight jobs per thread is optimal). If you want to adjust the number of accounts or balances per job, you can do this by increasing or decreasing the number of threads. However, when the number of accounts or balances is too small, the data modules use one thread to optimize performance.

---

**Breaking Up Large Nested Subsections in Registry Files**

Pipeline Manager can encounter parser stack overflow errors when a pipeline registry section contains a large number of nested subsections.

You can break up large nested subsections and prevent parser stack overflow errors by using anonymous blocks in your registry file. An anonymous block consists of a nested subsection with braces `{}` and no subsection name, as shown below:

```plaintext
#------------------------------------------------------------------------
# Input section
#------------------------------------------------------------------------
Input
{
  UnitsPerTransaction = 1
 InputModule
  {
    # <-- Beginning of Anonymous Block
    ModuleName = INP_GenericStream
    Module
    {
      Grammar = ./formatDesc/Formats/Solution42/SOL42_V670_REL_InGrammar.dsc
      DefaultOutput = TELOutput
    }
    InputStream
    {
      ModuleName = EXT_InFileManager
      Module
      {
        InputPath = ./data/incollect/reprice/in
        InputPrefix = test_
        InputSuffix = .edr
        ...
      }
    } # end of InputStream
  } # end of InputModule
  } # --> End of Anonymous Block
} # end of InputDataPool

You can place anonymous blocks in any location and at any hierarchy level of the registry file. For the best effect, divide large sections by placing an anonymous block around a group of smaller subsections. This breaks up the section without affecting the hierarchy of the subsections enclosed within the anonymous block.
Optimizing a Pipeline by Using Function Pools

In general, the performance of a multithreaded pipeline varies directly with its slowest thread. The objective of optimizing a multithreaded pipeline is to group the function modules into function pools so that the slowest function pool is as fast as possible. In this environment, faster threads wait a minimum amount of time for data to be delivered or processed by slower threads.

**Important:** Adding too many function pools to a pipeline can decrease performance because the buffers between the threads consume system CPU overhead and RAM. (Typically, two to six function pools is optimal.)

You use instrumentation results to guide function pool configuration. Instrumentation results indicate how many microseconds are required by each module to process a given number of requests. You use this information to add function pools or regroup the modules in existing function pools.

**Tip:** You cannot improve performance by adding function pools or shifting function modules to adjacent function pools if your slowest function pool:

- Has one function module in it. (or)
- Is faster than the input or output module.

You might be able to improve performance by reducing the number of function pools as long as the slowest function pool is faster than the output module. (Any performance gain comes from the reduced number of buffers. Fewer buffers require less system process overhead.)

**Note:** For information on the buffers between the function pools, see "Configuring Buffers".

```plaintext
input {...}

Functions
{
  PreProcessing
  {
    FunctionPool
    {
      module_1 {}
      module_2 {}
      module_3 {}
    }
  }
```
The instrumentation output in the `pipeline.log` file reveals the following latencies for each module for processing a fixed set of test transactions:

**Note:** For simplicity, the sample latencies have been rounded to the nearest 5,000,000 microseconds.

```
15.03.2004 13:25:07 testserver ifw IFW NORMAL 00516 -
(ifw.Pipelines.ALL_RATE.Functions.PreProcessing)
Plugin processing time statistics:

15.03.2004 13:25:07 testserver ifw IFW NORMAL 00516 -
(ifw.Pipelines.ALL_RATE.Functions.Rating)
Plugin processing time statistics:

15.03.2004 13:25:07 testserver ifw IFW NORMAL 00516 -
(ifw.Pipelines.ALL_RATE.Functions.PostRating)
Plugin processing time statistics:
```

This output is summarized in Table 15–1:
The total latency in this configuration is 185,000,000; this represents the microseconds used by the slowest function pool.

Figure 15–1 shows that about a third of the CPU cycles used by the function pool threads are idle:

In this example, the pipeline can be optimized if module_6 is assigned to its own function pool, as in this revised sample:

```
input {...}

Functions
{
  PreProcessing
  {
    FunctionPool
    {
      module_1 {}
      module_2 {}
      module_3 {}
    }
  }
  Buffer1 {...}
}

Rating
{
  FunctionPool
  {
    module_4 {}
    module_5 {}
    module_6 {}
    module_7 {}
    module_8 {}
  }
  PostRating
  {
    module_7 {}
    module_8 {}
  }
}
```

### Table 15–1 Example 1 Module Latencies Summary

<table>
<thead>
<tr>
<th>Module</th>
<th>Module Latency (Microseconds)</th>
<th>Function Pool</th>
<th>Function Pool Latency (Microseconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>module_1</td>
<td>40,000,000</td>
<td>PreProcessing</td>
<td>100,000,000</td>
</tr>
<tr>
<td>module_2</td>
<td>15,000,000</td>
<td>PreProcessing</td>
<td>100,000,000</td>
</tr>
<tr>
<td>module_3</td>
<td>45,000,000</td>
<td>PreProcessing</td>
<td>100,000,000</td>
</tr>
<tr>
<td>module_4</td>
<td>65,000,000</td>
<td>Rating</td>
<td>185,000,000</td>
</tr>
<tr>
<td>module_5</td>
<td>30,000,000</td>
<td>Rating</td>
<td>185,000,000</td>
</tr>
<tr>
<td>module_6</td>
<td>90,000,000</td>
<td>Rating</td>
<td>185,000,000</td>
</tr>
<tr>
<td>module_7</td>
<td>35,000,000</td>
<td>PostRating</td>
<td>85,000,000</td>
</tr>
<tr>
<td>module_8</td>
<td>50,000,000</td>
<td>PostRating</td>
<td>85,000,000</td>
</tr>
</tbody>
</table>
module_4 {}
module_5 {}

Buffer2 {...}

Discounting
{
    functionpool
    {
        module_6 {}
    }
}

Buffer3 {...}

PostRating
{
    FunctionPool
    {
        module_7 {}
        module_8 {}
    }
}

output {...}

The latency table now appears as shown in Table 15–2:

<table>
<thead>
<tr>
<th>Module</th>
<th>Module Latency (Microseconds)</th>
<th>Function Pool</th>
<th>Function Pool Latency (Microseconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>module_1</td>
<td>40,000,000</td>
<td>PreProcessing</td>
<td>100,000,000</td>
</tr>
<tr>
<td>module_2</td>
<td>15,000,000</td>
<td>PreProcessing</td>
<td>100,000,000</td>
</tr>
<tr>
<td>module_3</td>
<td>45,000,000</td>
<td>PreProcessing</td>
<td>100,000,000</td>
</tr>
<tr>
<td>module_4</td>
<td>65,000,000</td>
<td>Rating</td>
<td>95,000,000</td>
</tr>
<tr>
<td>module_5</td>
<td>30,000,000</td>
<td>Rating</td>
<td>95,000,000</td>
</tr>
<tr>
<td>module_6</td>
<td>90,000,000</td>
<td>Discounting</td>
<td>90,000,000</td>
</tr>
<tr>
<td>module_7</td>
<td>35,000,000</td>
<td>PostRating</td>
<td>85,000,000</td>
</tr>
<tr>
<td>module_8</td>
<td>50,000,000</td>
<td>PostRating</td>
<td>85,000,000</td>
</tr>
</tbody>
</table>

Total function module latency in the new configuration is 100,000,000 microseconds, equivalent to the latency of the PreProcessing function pool. Less than eight percent of function pool CPU cycles are now idle as shown by the gray cycles in Figure 15–2:
Figure 15–2 Unused CPU Cycles Example 2

![Diagram](https://via.placeholder.com/150)

Shifting Modules Between Function Pools

Adding an additional function pool can decrease performance in some situations (see "Adding Additional Function Pools" for more information). This can occur if the system overhead for the additional buffer more than offsets the performance gains from a faster highest-latency function pool. When this occurs, you might be able to improve performance by keeping the number of function pools constant and shifting modules to adjoining function pools.

In the sample above, if adding an additional function pool decreased performance, you could return to using three function pools and then move module 4 to the end of the PreProcessing function pool as shown in Table 15–3:

Table 15–3 Example 3 Modules Latencies Summary

<table>
<thead>
<tr>
<th>Module</th>
<th>Module Latency (Microseconds)</th>
<th>Function Pool</th>
<th>Function Pool Latency (Microseconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>module_1</td>
<td>40,000,000</td>
<td>PreProcessing</td>
<td>165,000,000</td>
</tr>
<tr>
<td>module_2</td>
<td>15,000,000</td>
<td>PreProcessing</td>
<td>165,000,000</td>
</tr>
<tr>
<td>module_3</td>
<td>45,000,000</td>
<td>PreProcessing</td>
<td>165,000,000</td>
</tr>
<tr>
<td>module_4</td>
<td>65,000,000</td>
<td>PreProcessing</td>
<td>165,000,000</td>
</tr>
<tr>
<td>module_5</td>
<td>30,000,000</td>
<td>Rating</td>
<td>120,000,000</td>
</tr>
<tr>
<td>module_6</td>
<td>90,000,000</td>
<td>Rating</td>
<td>120,000,000</td>
</tr>
<tr>
<td>module_7</td>
<td>35,000,000</td>
<td>PostRating</td>
<td>85,000,000</td>
</tr>
<tr>
<td>module_8</td>
<td>50,000,000</td>
<td>PostRating</td>
<td>85,000,000</td>
</tr>
</tbody>
</table>

Total function module latency in the new configuration is 165,000,000 microseconds. This is equivalent to the latency of the PreProcessing function pool. While performance gains might be more modest than in the first scenario where a new function pool was added, the performance gain is more certain because no additional buffer overhead was added.

Configuring Buffers

In a multithreaded pipeline, each pair of consecutive threads communicates through a buffer. Because each function pool is assigned a thread, you must configure a buffer between consecutive function pools.
Combining Multiple CDR Files into One Transaction

You configure the buffers between function pool sections in the pipeline registry file. Normally, each buffer can be configured as follows:

Buffer1
{
  Size = 100
}

**Important:** On Solaris systems, you should configure block transfer mode. See "Block Transfer Mode" in BRM Installation Guide.

## Combining Multiple CDR Files into One Transaction

Pipeline Manager is generally more efficient when it processes large CDR files. If a pipeline receives and processes small CDR files, you can improve processing performance by combining multiple CDR input files into one pipeline transaction. You use the **UnitsPerTransaction** registry entry in the input controller to implement this functionality. See "Input Controller“ in BRM Configuring Pipeline Rating and Discounting.

The **UnitsPerTransaction** entry specifies the number of CDR input files that make up a transaction. By default, each CDR file forms its own transaction.

**Note:** The optimal transaction size depends on your system configuration and pricing model. In general, most system configurations perform best when the total number of CDRs, which is the average number of CDRs per input file multiplied by the number of input files in the transaction, is greater than 10,000.

If the **UnitsPerTransaction** value is greater than 1, you can use the **SequenceGeneration** registry entry in the output controller to specify whether the pipeline generates one output file per CDR input file or one output file for the entire transaction (see "Output Controller“ in BRM Configuring Pipeline Rating and Discounting). Pipeline Manager performance is generally faster when one output file is generated for the entire (multi-CDR) transaction.

To combine multiple CDR files into one transaction:

1. In the **Input** section of the registry file, set the **UnitsPerTransaction** entry to the number of CDR input files that make up one transaction. For example, set **UnitsPerTransaction** to **100** to combine 100 CDR input files into one transaction.

   **Note:** The default **UnitsPerTransaction** value is **1**.

   ```plaintext
   Input
   {
     ...
     UnitsPerTransaction = 100
     ...
   }
   ```

2. (Optional) In the **Output** section of the registry file, set the **SequenceGeneration** entry to **Transaction**. This configures the pipeline to generate one output file for the entire transaction.
Increasing Pipeline Manager Throughput When an EDR Is Associated with Multiple Output Streams

You can enhance Pipeline Manager throughput by configuring multithreading in the Output Controller. This allows Pipeline Manager to write multiple EDRs in parallel when the EDRs are associated with multiple output streams.

Enabling multithreading may cause an increase in the overall memory usage of the Output Controller. However, the memory usage becomes constant after processing EDRs for some time.

To configure multithreading in the Output Controller:

1. Open the registry file (for example, Pipeline_Home/conf/wireless.reg) in a text editor.

2. In the **MultiThreading** section, do the following:
   - Set the **Active** registry entry to **True**.
   - Set the **NumberOfThreads** registry entry to the number of threads you want the Output Controller to create for Pipeline Manager to write multiple EDRs in parallel.
   - Set the **BatchSize** registry entry to the appropriate value:
     - 0 indicates that the Output Controller does not run in batch mode.
     - A value greater than 0 indicates that the Output Controller operates in batch mode with the batch size equal to the specified value.

For example:

```plaintext
Output
{
  ...
  SequenceGeneration = Transaction
  ...
}

3. Stop and restart the pipeline. See "Starting and Stopping Individual Pipelines".

---

**Note:** The default **SequenceGeneration** value is **Units**, which configures the pipeline to generate one output file per CDR input file.

---

**Important:** Enable multithreading in the Output Controller only if the EDRs are associated with multiple output streams.
3. Save and close the file.
4. Restart the pipeline. See "Starting and Stopping Individual Pipelines".

For information about the MultiThreading registry entry, see "Output Controller" in BRM Configuring Pipeline Rating and Discounting.

**Configuring Multiple Pipelines**

If you have high transaction throughput requirements and additional system resources, you might improve system performance by running multiple pipelines that perform the same function.

In general, consider running multiple pipelines if:

- Your system has a relatively large number of CPUs.
- The order of the input streams is not important.

---

**Note:** When you use the FCT_CallAssembling or FCT_DuplicateCheck module, you have to process the EDRs for the same account in the same pipeline. See "Using Duplicate Check with Multiple Pipelines" and "Assembling Calls with Multiple Pipelines" in BRM Configuring Pipeline Rating and Discounting.

---

**Tip:** If you configure multiple pipelines and your system is running at near full capacity on a limited number of CPUs, test running the pipelines in single-threaded mode. This configuration reduces the buffer memory allocation requirement and thread-handling overhead. To enable single-threaded operation, set the MultiThreaded entry to False. See "Assigning Multiple Threads to Process Function Modules".

### Customizing Flists Sent to a Real-Time Pipeline

You can configure the fields included in flists sent to a real-time pipeline by using the load_pin_rtp_trim_flist utility (see BRM Developer’s Guide). This utility is useful for:

- Improving system efficiency by removing (trimming) fields that the pipeline does not use.
- Supporting custom iScripts and iRules in the pipeline by adding fields to default flists that are not included in the flists by default.

To optimize the set of fields sent to a real-time pipeline:

1. Determine which fields are required by the real-time pipeline.
2. Create an XML file that describes the fields to be sent to the real-time pipeline based on one or more event types.
3. Load the XML file using the load_pin_rtp_trim_flist utility.

### Configuration Object Dot Notation

The load_pin_rtp_trim_flist utility creates a configuration object (/config/rtp/trim_flist). This object is used to create the trimmed flists.
The configuration object uses dot notation. For example, the PIN_FLD_STATUS_FLAGS field at the end of this portion of a sample flist:

0 PIN_FLD_INHERITED_INFO SUBSTRUCT [0] allocated 32, used 32
1 PIN_FLD_POID POID [0] 0.0.0.1 /account 10243 13
1 PIN_FLD_MOD_T TSTAMP [0] (1063218065) Wed Sep 10 11:21:05 2003
1 PIN_FLD_ACCOUNT_NO STR [0] *0.0.0.1-10243*
1 PIN_FLD_CURRENT_INT [0] 840
1 PIN_FLD_LAST_BILL_T TSTAMP [0] (1063217469) Wed Sep 10 11:11:09 2003
1 PIN_FLD_BAL_GRP_OBJ POID [0] 0.0.0.1 /balance_group 8323 4
1 PIN_FLD_SERVICE_INFO SUBSTRUCT [0] allocated 32, used 32
2 PIN_FLD_STATUS_ENUM [0] 10100
2 PIN_FLD_STATUS_FLAGS INT [0] 0

is represented as:

PIN_FLD_INHERITED_INFO.PIN_FLD_SERVICE_INFO.PIN_FLD_STATUS_FLAGS

in the configuration object.

About the field_list.xml File

The field_list.xml file specifies the fields from the /account and /service objects that are included in the flist that is sent to Pipeline Manager. You can define conditions in <EventMap> sections in the XML file that indicate which fields should be included in the flist depending on the event type.

The following example shows the XML file structure with session and provisioning event filters:

```xml
<EventMapList>

<!--* The following event map specifies fields sent * when the event type is exactly /event/session -->

<EventMap>
  <Event>
    <Type>/event/session</Type>
    <Flags>0</Flags>
  </Event>
  <RequiredField>
    <!-- List of fields sent put here. -->
  </RequiredField>
</EventMap>

<!--* The following event map specifies fields sent * when the event type starts with /event/session/ -->

<EventMap>
  <Event>
    <Type>/event/session/</Type>
    <Flags>1</Flags>
  </Event>
  <RequiredField>
    <!-- List of fields sent put here. -->
  </RequiredField>
</EventMap>
```

Customizing Flists Sent to a Real-Time Pipeline

The `Flags` tag in the XML file specifies event matching criteria.

- A `Flags` value of 0 specifies that an exact match is required.
- A `Flags` value of 1 specifies that the event type must start with the string specified in the `Type` tag. The value 1 is also used when indicating `Type` value asterisk (*). This value matches all event types.

**Important:** Search order is important. The fields included with the flist are the fields specified in the first event map section of the XML file where the event type matches the string in the `Type` field.

You can use the sample XML fields list (BRM_Home/sys/data/config/pin_config_rtp_trim_flist.xml) as a base for your custom XML file.

For a detailed example using session event filters, see "Usage Example".
Mapping Events to Flists

Because one flist can be used by more than one event, you can specify the relationship between an event and the flist.

For example, the following section is of an event map XML file:

```
<EventMap>
  <Event>
    <Type>/event/session</Type>
    <Flags>0</Flags>
  </Event>
  <Event>
    <Type>/event/session/</Type>
    <Flags>1</Flags>
  </Event>
</EventMap>
```

is mapped to an flist as follows:

0 PIN_FLD_EVENT_MAP ARRAY [0] allocated 20, used 8
1 PIN_FLD_EVENTS ARRAY [0] allocated 20, used 8
2 PIN_FLD_EVENT_TYPE STR [0] "/event/session"
2 PIN_FLD_FLAGS INT [0] 0
1 PIN_FLD_EVENTS ARRAY [1] allocated 20, used 8
2 PIN_FLD_EVENT_TYPE STR [0] "/event/session/"
2 PIN_FLD_FLAGS INT [0] 1

Usage Example

An unmodified flist might look like the sample shown in "Sample Unmodified Flist". However, in this example, Pipeline Manager only requires subsets of fields listed in "Sample Fields Required by Pipeline Manager" depending on the event type.

In this example, to implement the trimmed flist:

1. Create the XML file shown in "sample.xml File" to modify the default list of fields ("Sample Unmodified Flist") included in the flist.
2. Load the XML file using the utility:

   `load_pin_rtp_trim_flist -f sample.xml [-v] [-d]`

Sample Unmodified Flist

The following is the default (untrimmed) list of fields sent to Pipeline Manager.

```
0 PIN_FLD_POID POID [0] 0.0.0.1 /event/session -1 0
0 PIN_FLD_EVENT SUBSTRUCT [0] allocated 25, used 25
1 PIN_FLD_POID POID [0] 0.0.0.1 /event/session -1 0
1 PIN_FLD_NAME STR [0] "Activity Session Log"
1 PIN_FLD_USERID POID [0] 0.0.0.1 /service/pcm_client 1 0
1 PIN_FLD_ACCOUNT_OBJ POID [0] 0.0.0.1 /account 10243 0
1 PIN_FLD_PROGRAM_NAME STR [0] "testnap"
1 PIN_FLD_START_T TSTAMP [0] (1065785673) Fri Oct 10 04:34:33 2003
1 PIN_FLD_END_T TSTAMP [0] (1065785683) Fri Oct 10 04:34:43 2003
1 PIN_FLD_SERVICE_OBJ POID [0] 0.0.0.1 /service/ip 11907 1
1 PIN_FLD_SYS_DESCR STR [0] "Session: generic"
1 PIN_FLD_RUM_NAME STR [0] "Duration"
1 PIN_FLD_UNIT ENUM [0] 1
1 PIN_FLD_TOD_MODE ENUM [0] 2
1 PIN_FLD_NET_QUANTITY DECIMAL [0] 60.000000000000000
1 PIN_FLD_MIN_QUANTITY DECIMAL [0] 60.000000000000000
1 PIN_FLD_INCR_QUANTITY DECIMAL [0] 60.000000000000000
1 PIN_FLD_MIN_UNIT ENUM [0] 2
```
<table>
<thead>
<tr>
<th>PIN_FLDIDX</th>
<th>NAME</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>PIN_FLD_INCR_UNIT</td>
<td>ENUM [0] 2</td>
</tr>
<tr>
<td>1</td>
<td>PIN_FLD_ROUNDING_MODE</td>
<td>ENUM [0] 1</td>
</tr>
<tr>
<td>1</td>
<td>PIN_FLD_TIMEZONE_MODE</td>
<td>ENUM [0] 1</td>
</tr>
<tr>
<td>1</td>
<td>PIN_FLD_TIMEZONE_ID</td>
<td>STR [0] &quot;GMT-08:00&quot;</td>
</tr>
<tr>
<td></td>
<td>PIN_FLD_TIMEZONE_ADJ_START_T</td>
<td>TSTAMP [0] (1065760473) Thu Oct 09 21:34:33 2003</td>
</tr>
<tr>
<td>1</td>
<td>PIN_FLD_TIMEZONE_ADJ_END_T</td>
<td>TSTAMP [0] (1065760483) Thu Oct 09 21:34:43 2003</td>
</tr>
<tr>
<td>1</td>
<td>PIN_FLD_TOTAL</td>
<td>ARRAY [840] allocated 20, used 1</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_AMOUNT</td>
<td>DECIMAL [0] 0.0166667</td>
</tr>
<tr>
<td>1</td>
<td>PIN_FLD_BAL_IMPACTS</td>
<td>ARRAY [840] allocated 20, used 17</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_ACCOUNT_OBJ</td>
<td>POID [0] 0.0.0.1 /account 10243 13</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_AMOUNT</td>
<td>DECIMAL [0] 0.0166667</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_RESOURCE_ID</td>
<td>INT [0] 840</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_PRODUCT_OBJ</td>
<td>POID [0] 0.0.0.1 /product 10030 0</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_RATE_OBJ</td>
<td>POID [0] 0.0.0.1 /rate 9390 1</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_DISCOUNT</td>
<td>DECIMAL [0] 0</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_AMOUNT_DEFERR</td>
<td>DECIMAL [0] 0</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_GL_ID</td>
<td>INT [0] 104</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_IMPACT_TYPE</td>
<td>ENUM [0] 1</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_QUANTITY</td>
<td>DECIMAL [0] 60.00000000</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_RATE_TAG</td>
<td>STR [0] &quot;$1 per hour&quot;</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_TAX_CODE</td>
<td>STR [0] &quot;&quot;</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_IMPACTCATEGORY</td>
<td>STR [0] &quot;default&quot;</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_PACKAGE_ID</td>
<td>INT [0] &quot;20030910&quot;</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_LINEAGE</td>
<td>STR [0] &quot;&quot;</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_PERCENT</td>
<td>DECIMAL [0] 1</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_BAL_GRP_OBJ</td>
<td>POID [0] 0.0.0.1 /balance_group 8323 4</td>
</tr>
<tr>
<td>1</td>
<td>PIN_FLD_UNRATED_QUANTITY</td>
<td>DECIMAL [0] 0</td>
</tr>
<tr>
<td>0</td>
<td>PIN_FLD_DISCOUNTS</td>
<td>ARRAY [0] allocated 20, used 8</td>
</tr>
<tr>
<td>1</td>
<td>PIN_FLD_ACCOUNT_OBJ</td>
<td>POID [0] 0.0.0.1 /account 10243 0</td>
</tr>
<tr>
<td>1</td>
<td>PIN_FLD_OWNER_OBJ</td>
<td>POID [0] 0.0.0.1 /service/ip 11907 1</td>
</tr>
<tr>
<td>1</td>
<td>PIN_FLD_BAL_GRP_OBJ</td>
<td>POID [0] 0.0.0.1 /balance_group 8323 4</td>
</tr>
<tr>
<td>1</td>
<td>PIN_FLD_DISCOUNTLIST</td>
<td>ARRAY [0] allocated 20, used 19</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_CREATED_T</td>
<td>TSTAMP [0] (1063218065) Wed Sep 10 11:21:05 2003</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_CYCLE_END_T</td>
<td>TSTAMP [0] (1052871608) Wed Sep 10 11:21:05 2003</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_CYCLE_START_T</td>
<td>TSTAMP [0] (1052871608) Wed Sep 10 11:21:05 2003</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_DEAL_OBJ</td>
<td>POID [0] 0.0.0.0 0 0</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_DESCR</td>
<td>STR [0] &quot;&quot;</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_DISCOUNT_OBJ</td>
<td>POID [0] 0.0.0.1 /discount 8273 0</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_LAST_MODIFIED_T</td>
<td>TSTAMP [0] (1063218065) Wed Sep 10 11:21:05 2003</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_PACKAGE_ID</td>
<td>INT [0] &quot;12222&quot;</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_PLAN_OBJ</td>
<td>POID [0] 0.0.0.0 0 0</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_PURCHASE_END_T</td>
<td>TSTAMP [0] (1052871608) Wed Sep 10 11:21:05 2003</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_PURCHASE_START_T</td>
<td>TSTAMP [0] (1052871608) Wed Sep 10 11:21:05 2003</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_QUANTITY</td>
<td>DECIMAL [0] 1</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_STATUS_FLAGS</td>
<td>INT [0] 1</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_USAGE_END_T</td>
<td>TSTAMP [0] (1052871608) Wed Sep 10 11:21:05 2003</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_USAGE_START_T</td>
<td>TSTAMP [0] (1052871608) Wed Sep 10 11:21:05 2003</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_FLAGS</td>
<td>INT [0] 1</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_TYPE</td>
<td>ENUM [0] 602</td>
</tr>
<tr>
<td>1</td>
<td>PIN_FLD_DISCOUNT_LIST</td>
<td>ARRAY [1] allocated 20, used 19</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_CREATED_T</td>
<td>TSTAMP [0] (1063218065) Wed Sep 10 11:21:05 2003</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_CYCLE_END_T</td>
<td>TSTAMP [0] (1071385462) Sat Dec 13 23:04:22 2003</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_CYCLE_START_T</td>
<td>TSTAMP [0] (1052895862) Tue May 13 17:20:08 2003</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_DEAL_OBJ</td>
<td>POID [0] 0.0.0.0 0 0</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_DESCR</td>
<td>STR [0] &quot;&quot;</td>
</tr>
<tr>
<td>2</td>
<td>PIN_FLD_DISCOUNT_OBJ</td>
<td>POID [0] 0.0.0.1 /discount 11345 0</td>
</tr>
</tbody>
</table>
Customizing Flits Sent to a Real-Time Pipeline

Optimizing Pipeline Manager Performance

15-23
Customizing Flits Sent to a Real-Time Pipeline

| PIN_FLD_OWN_MIN | DECIMAL [0] | 0 |
| PIN_FLD_PERMITTED | STR [0] | "" |
| PIN_FLD_PRIORITY | DECIMAL [0] | 14 |
| PIN_FLD_PURCHASE_MAX | DECIMAL [0] | 0 |
| PIN_FLD_PURCHASE_MIN | DECIMAL [0] | 0 |
| PIN_FLD_TYPE | ENUM [0] | 603 |
| PIN_FLD_USAGE_MAP | ARRAY [0] | allocated 20, used 4 |
| PIN_FLD_DISCOUNT_MODEL | STR [0] | "DMStandard" |
| PIN_FLD_EVENT_TYPE | STR [0] | "/event/session" |
| PIN_FLD_FLAGS | INT [0] | 1 |
| PIN_FLD_SNOWBALL_FLAG | INT [0] | 0 |
| PIN_FLD_DISCOUNT_OBJ | POID [0] | 0.0.0.1 /discount 9755 1 |
| PIN_FLD_SERVICE_OBJ | POID [0] | NULL poid pointer |
| PIN_FLD_PACKAGE_ID | INT [0] | "" |
| PIN_FLD_PURCHASE_END_T | TSTAMP [0] | (1069334036) Thu Nov 20 05:13:56 2003 |
| PIN_FLD_USAGE_END_T | TSTAMP [0] | (1069334036) Thu Nov 20 05:13:56 2003 |
| PIN_FLD_STATUS | ENUM [0] | 1 |
| PIN_FLD_FLAGS | INT [0] | 1 |
| PIN_FLD_DISCOUNT_LIST | ARRAY [4] | allocated 28, used 28 |
| PIN_FLD_POID | POID [0] | 0.0.0.1 /discount 11291 1 |
| PIN_FLD_CREATED_T | TSTAMP [0] | (1064334029) Tue Sep 23 09:20:29 2003 |
| PIN_FLD_READ_ACCESS | STR [0] | "R" |
| PIN_FLD_WRITE_ACCESS | STR [0] | "W" |
| PIN_FLD_ACCOUNT_OBJ | POID [0] | 0.0.0.1 /account 1 1 |
| PIN_FLD_DESCR | STR [0] | "" |
| PIN_FLD_END_T | TSTAMP [0] | (1069334029) Thu Nov 20 05:13:49 2003 |
| PIN_FLD_MODE | ENUM [0] | 801 |
| PIN_FLD_NAME | STR [0] | "Sys discount 2" |
| PIN_FLD_OWN_MAX | DECIMAL [0] | 0 |
| PIN_FLD_OWN_MIN | DECIMAL [0] | 0 |
| PIN_FLD_PERMITTED | STR [0] | "" |
| PIN_FLD_PRIORITY | DECIMAL [0] | 200 |
| PIN_FLD_PURCHASE_MAX | DECIMAL [0] | 0 |
| PIN_FLD_PURCHASE_MIN | DECIMAL [0] | 0 |
| PIN_FLD_TYPE | ENUM [0] | 603 |
| PIN_FLD_USAGE_MAP | ARRAY [0] | allocated 20, used 4 |
| PIN_FLD_DISCOUNT_MODEL | STR [0] | "DMStandard" |
| PIN_FLD_EVENT_TYPE | STR [0] | "/event/session" |
| PIN_FLD_FLAGS | INT [0] | 1 |
| PIN_FLD_SNOWBALL_FLAG | INT [0] | 0 |
| PIN_FLD_DISCOUNT_OBJ | POID [0] | 0.0.0.1 /discount 11291 1 |
| PIN_FLD_SERVICE_OBJ | POID [0] | NULL poid pointer |
| PIN_FLD_PACKAGE_ID | STR [0] | "" |
| PIN_FLD_PURCHASE_END_T | TSTAMP [0] | (1069334036) Thu Nov 20 05:13:56 2003 |
| PIN_FLD_USAGE_END_T | TSTAMP [0] | (1069334036) Thu Nov 20 05:13:56 2003 |
| PIN_FLD_STATUS | ENUM [0] | 1 |
| PIN_FLD_FLAGS | INT [0] | 1 |
| PIN_FLD_BAL_INFO | ARRAY [0] | allocated 20, used 3 |
| PIN_FLD_BAL_GRP_OBJ | POID [0] | 0.0.0.1 /balance_group 8323 4 |
| PIN_FLD_BALANCES | ARRAY [840] | allocated 11, used 6 |
| PIN_FLD_NEXT_BAL | DECIMAL [0] | 0 |
Customizing Fls\text{t}s Sent to a Real-Time Pipeline

Optimizing Pipeline Manager Performance

\begin{verbatim}
2  PIN_FLD_RESERVED_AMOUNT DECIMAL [0] 0
2  PIN_FLD_CURRENT_BAL DECIMAL [0] 19.590836
2  PIN_FLD_CREDIT_LIMIT DECIMAL [0] 100
2  PIN_FLD_CREDIT_FLOOR DECIMAL [0] 0
2  PIN_FLD_CREDIT_THRESHOLDS INT [0] 0
1  PIN_FLD_BALANCES ARRAY [1000001] allocated 7, used 6
2  PIN_FLD_NEXT_BAL DECIMAL [0] 0
2  PIN_FLD_RESERVED_AMOUNT DECIMAL [0] 0
2  PIN_FLD_CURRENT_BAL DECIMAL [0] 0
2  PIN_FLD_CREDIT_LIMIT DECIMAL [0] 100
2  PIN_FLD_CREDIT_FLOOR DECIMAL [0] 0
2  PIN_FLD_CREDIT_THRESHOLDS INT [0] 0
1  PIN_FLD_INHERITED_INFO SUBSTRUCT [0] allocated 32, used 32
1  PIN_FLD_POID POID [0] 0.0.0.1 /account 10243 13
1  PIN_FLD_MOD_T TSTAMP [0] (1063218065) Wed Sep 10 11:21:05 2003
1  PIN_FLD_ACCOUNT_NO STR [0] "0.0.0.1-10243"
1  PIN_FLD_STATUS ENUM [0] 10100
1  PIN_FLD_STATUS_FLAGS INT [0] 0
1  PIN_FLD_TIMEZONE_ID STR [0] ""
1  PIN_FLD_HOLIDAY_ENUM [0] 0
1  PIN_FLD_INHERITED_INFO_SUBSTRUCT [0] allocated 51, used 26
2  PIN_FLD_POID POID [0] 0.0.0.1 /service/ip 11907 5
2  PIN_FLD_CREATED_T TSTAMP [0] (1063217471) Wed Sep 10 11:11:11 2003
2  PIN_FLD_MOD_T TSTAMP [0] (1063217473) Wed Sep 10 11:11:13 2003
2  PIN_FLD_READ_ACCESS STR [0] "L"
2  PIN_FLD_WRITE_ACCESS STR [0] "L"
2  PIN_FLD_AAC_ACCESS STR [0] ""
2  PIN_FLD_AAC_PACKAGE STR [0] ""
2  PIN_FLD_AAC_PROMO_CODE STR [0] ""
2  PIN_FLD_AAC_SERIAL_NUM STR [0] ""
2  PIN_FLD_AAC_SOURCE STR [0] ""
2  PIN_FLD_AAC_VENDOR STR [0] ""
2  PIN_FLD_ACCOUNT_OBJ POID [0] 0.0.0.1 /account 10243
2  PIN_FLD_CLOSE_WHEN_T TSTAMP [0] (0) <null>
2  PIN_FLD_EFFECTIVE_T TSTAMP [0] (1063217469) Wed Sep 10 11:11:09 2003
2  PIN_FLD_ITEM_POID_LIST STR [0] "0.0.0.1|/item/misc 8835 0"
2  PIN_FLD-next_ITEM_POID_LIST STR [0] ""
2  PIN_FLD_ACTG_TYPE ENUM [0] 2
2  PIN_FLD_LAST_STATUS_T TSTAMP [0] (1063217469) Wed Sep 10 11:11:09 2003
2  PIN_FLD_GL_SEGMENT STR [0] "."
2  PIN_FLD_PAY_TYPE ENUM [0] 0
2  PIN_FLD_AR_BILLINFO_OBJ POID [0] 0.0.0.1 /billinfo 8451 0
2  PIN_FLD_BILL_OBJ POID [0] 0.0.0.1 /bill 10499 0
2  PIN_FLD_PENDING_RECV DECIMAL [0] 0
2  PIN_FLD_BAL_GRP_OBJ POID [0] 0.0.0.1 /balance_group 8323 4
2  PIN_FLD_SERVICE_INFO SUBSTRUCT [0] allocated 51, used 26
2  PIN_FLD_POID POID [0] 0.0.0.1 /service/ip 11907 5
2  PIN_FLD_CREATED_T TSTAMP [0] (1063217471) Wed Sep 10 11:11:11 2003
2  PIN_FLD_MOD_T TSTAMP [0] (1063217473) Wed Sep 10 11:11:13 2003
2  PIN_FLD_READ_ACCESS STR [0] "L"
2  PIN_FLD_WRITE_ACCESS STR [0] "L"
2  PIN_FLD_AAC_ACCESS STR [0] ""
2  PIN_FLD_AAC_PACKAGE STR [0] ""
2  PIN_FLD_AAC_PROMO_CODE STR [0] ""
2  PIN_FLD_AAC_SERIAL_NUM STR [0] ""
2  PIN_FLD_AAC_SOURCE STR [0] ""
2  PIN_FLD_AAC_VENDOR STR [0] ""
2  PIN_FLD_ACCOUNT_OBJ POID [0] 0.0.0.1 /account 10243 0
2  PIN_FLD_CLOSE_WHEN_T TSTAMP [0] (0) <null>
2  PIN_FLD_EFFECTIVE_T TSTAMP [0] (1063217469) Wed Sep 10 11:11:09 2003
2  PIN_FLD_ITEM_POID_LIST STR [0] "0.0.0.1|/item/cycle_forward 11651 0"
\end{verbatim}
Sample Fields Required by Pipeline Manager

The following are sample fields, in flist format, required by Pipeline Manager when the event type is /event/session:

Important: You cannot trim the default fields for the PIN_FLD_INHERITED_INFO substruct listed in "Sample Unmodified Flist". However, you can specify additional /account and /service fields. In the text below, the /account field PIN_FLD_RESIDENCE_FLAG is specified at the end of the list. It is added to the default PIN_FLD_INHERITED_INFO fields sent to Pipeline Manager.

```plaintext
0 PIN_FLD_POID POID [0] 0.0.0.1 /event/session -1 0
0 PIN_FLD_EVENT SUBSTRUCT [0] allocated 25, used 25
1 PIN_FLD_POID POID [0] 0.0.0.1 /event/session -1 0
1 PIN_FLD_START_T TSTAMP [0] (1065785673) Fri Oct 10 04:34:33 2003
1 PIN_FLD_END_T TSTAMP [0] (1065785683) Fri Oct 10 04:34:43 2003
1 PIN_FLD_BAL_IMPACTS ARRAY [0] allocated 20, used 17 and other array elements
  2 PIN_FLD_AMOUNT DECIMAL [0] 0.0166667
  2 PIN_FLD_AMOUNT_DEFERRED DECIMAL [0] 0
  2 PIN_FLDRESOURCE_ID INT [0] 840
  2 PIN_FLD_GL_ID INT [0] 104
  2 PIN_FLD_IMPACT_TYPE ENUM [0] 1
  2 PIN_FLD_QUANTITY DECIMAL [0] 60.00000000
  2 PIN_FLD_RATE_TAG STR [0] "$1 per hour"
  2 PIN_FLD_TAX_CODE STR [0] **
0 PIN_FLD_DISCOUNTS ARRAY [0] allocated 20, used 8 and other array elements
  1 PIN_FLD_ACCOUNT_OBJ POID [0] 0.0.0.1 /account 10243 0
  1 PIN_FLD_OWNER_OBJ POID [0] 0.0.0.1 /service/ip 11907 1
  1 PIN_FLD_BAL_GRP_OBJ POID [0] 0.0.0.1 /balance_group 8323 4
  1 PIN_FLD_DISCOUNT_LIST ARRAY [0] allocated 20, used 19 and other array elements
    2 PIN_FLD_DISCOUNT_OBJ POID [0] 0.0.0.1 /discount 8273 0
    2 PIN_FLD_PACKAGE_ID INT [0] "12222"
    2 PIN_FLD_PURCHASE_END_T TSTAMP [0] (0) <null>
    2 PIN_FLD_PURCHASE_START_T TSTAMP [0] (1052871608) Tue May 13 17:20:08 2003
    2 PIN_FLD_QUANTITY DECIMAL [0] 1
    2 PIN_FLD_STATUS ENUM [0] 1
    2 PIN_FLD_USAGE_END_T TSTAMP [0] (0) <null>
    2 PIN_FLD_USAGE_START_T TSTAMP [0] (1052871608) Tue May 13 17:20:08 2003
    2 PIN_FLD_FLAGS INT [0] 1
```
A different set of fields is required when the event type is /event/session/ (including the final forward slash), and another set of fields is sent for any other type of event. To implement the trimmed flist in the example, create the following XML file (sample.xml). When this XML file is loaded with load_pin_rtp_trim_flist, the flist sent to Pipeline Manager is constructed as follows:

- If the event type is exactly /event/session, the PIN_FLD_RESIDENCE_FLAG field is included with the trimmed flist as shown in the flist sample above, otherwise,
- If the event type starts with /event/session/ (including the last forward slash), the PIN_FLD_RESIDENCE_FLAG field is not included with the trimmed flist, otherwise,
- If the event type is any other value (which matches the section specified by Type value * with Flags value 1), then neither the PIN_FLD_RESIDENCE_FLAG field nor the PIN_FLD_BAL_IMPACTS array is included with the trimmed flist.

---

Important: You cannot trim the default fields for the PIN_FLD_INHERITED_INFO substruct listed in "Sample Unmodified Flist." However, you can specify additional /account and /service fields. In the text below, the /account field PIN_FLD_RESIDENCE_FLAG is specified at the end of the list. It is added to the default PIN_FLD_INHERITED_INFO fields sent to Pipeline Manager.

---

**sample.xml File**

```xml
<?xml version='1.0' encoding='UTF-8' ?>
<!--
================================================================
Copyright (c) 2004 Portal Software, Inc. All rights reserved.
This material is the confidential property of Portal Software, Inc.
or its Subsidiaries or licensors and may be used, reproduced, stored
or transmitted only in accordance with a valid Portal license or
sublicense agreement.
================================================================
-->

<RTPTrimFlistConfiguration xmlns="http://www.portal.com/InfranetXMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.portal.com/InfranetXMLSchema pin_config_rtp_trim_flist.xsd">
  <EventMapList>
    <EventMap>
      <!-- Section which specifies fields sent when the event type is exactly /event/session -->
      <Event>
```
<Type>/event/session</Type>
<Flags>0</Flags>
</Event>
<RequiredField>
  <Name>PIN_FLD_POID</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_EVENT</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_POID</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_START_T</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_END_T</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_BAL_IMPACTS</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_AMOUNT</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_AMOUNT_DEFERRED</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_RESOURCE_ID</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_GL_ID</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_IMPACT_TYPE</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_QUANTITY</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_RATE_TAG</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_TAX_CODE</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_DISCOUNTS</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_DISCOUNT_OBJ</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_BAL_GRP_OBJ</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_DISCOUNT_LIST</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_DISCOUNT_OBJ</Name>
</RequiredField>
<EventMap>

<!-- Section which specifies fields sent when the event type starts with /event/session/ -->

</EventMap>

<!-- Section which specifies fields sent when the event type starts with /event/session/ -->

</EventMap>

<Event>
  <Type>/event/session/</Type>
  <Flags>1</Flags>
  <RequiredField>
    <Name>PIN_FLD_POID</Name>
  </RequiredField>
</Event>
<RequiredField>
  <Name>PIN_FLD_EVENT</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_POID</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_START_T</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_END_T</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_BAL_IMPACTS</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_AMOUNT</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_RESOURCE_ID</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_GL_ID</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_IMPACT_TYPE</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_QUANTITY</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_RATE_TAG</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_TAX_CODE</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_DISCOUNTS</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_ACCOUNT_OBJ</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_OWNER_OBJ</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_BAL_GRP_OBJ</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_DISCOUNT_LIST</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_DISCOUNT_OBJ</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_PACKAGE_ID</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_PURCHASE_END_T</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_PURCHASE_START_T</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_QUANTITY</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_STATUS</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_USAGE_END_T</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_USAGE_START_T</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_FLAGS</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_TYPE</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_BAL_INFO</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_BAL_GRP_OBJ</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_BALANCES</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_CURRENT_BAL</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_POID</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_EVENT</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_DISCOUNTS</Name>
</RequiredField>

<!--* Section which specifies fields sent when the event type is
* any other value.-->
<RequiredField>
  <Name>PIN_FLD_ACCOUNT_OBJ</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_OWNER_OBJ</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_BAL_GRP_OBJ</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_DISCOUNT_LIST</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_PACKAGE_ID</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_PURCHASE_END_T</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_PURCHASE_START_T</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_QUANTITY</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_STATUS</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_USAGE_END_T</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_USAGE_START_T</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_FLAGS</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_TYPE</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_BAL_INFO</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_BAL_GRP_OBJ</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_BALANCES</Name>
</RequiredField>
<RequiredField>
  <Name>PIN_FLD_CURRENT_BAL</Name>
</RequiredField>
</RTPTrimFlistConfiguration>
Measuring System Latencies with Instrumentation

You use the Pipeline Manager instrumentation feature to determine how much processing time each Pipeline Manager component (function modules, iScripts, and iRules) is consuming in microseconds (millionths of a second). This information allows you to:

- Determine system benchmarks.
- Identify performance bottlenecks at the function module level.
- Add or reconfigure function pools to optimize CPU utilization.

Instrumentation collects statistics for the following components:

- The input module.
- Each function module.
- The output module.

After each transaction, the statistics for each pipeline tested are written to the `pipeline.log` file.

Using Instrumentation to Collect Module Performance Statistics

To enable instrumentation:

1. Start the pipeline.
2. Send a signal to the pipeline to toggle instrumentation on and off. Use the following commands to toggle the instrumentation state:

   **Solaris and Linux:**
   
   ```sh
   kill -s USR1 ifw_process_pid
   ```

   **HP-UX IA64:**
   
   ```sh
   kill -USR1 ifw_process_pid
   ```

   **AIX:**
   
   ```sh
   kill -s USR1 ifw_process_pid
   ```

   At the end of each transaction, the statistics are logged to the `pipeline.log` file and the statistics counters are reset.

**Note:** By default, Pipeline Manager instrumentation is disabled on startup. When Pipeline Manager is running, you can toggle between the disabled and enabled modes.

**Important:** Pipeline Manager begins gathering statistics immediately after receiving the signal. To assure accurate measurements, be sure that Pipeline Manager is not processing transactions when the signal is sent.

4. Check the pipeline log files for processing time statistics. See "Viewing Instrumentation Testing Results".
5. When testing is complete, stop the instrumentation process by sending another signal. See step 2.

**Viewing Instrumentation Testing Results**

Each log file record consists of the fully qualified module name and the accumulated processing time spent in the module.

---

**Note:** Pipeline processing time statistics are not cumulative. The output module writes data to a file whereas a function module processes EDRs in a different thread.

---

**Sample Log File**

The following sample log file shows instrumentation data:

```
',
```

```
',
```

```
',
```
Optimizing the DAT_USC_Map Module

The DAT_USC_Map module uses the Pipeline Manager framework component (FSM) to compile data mapping rules, which are stored in the database as regular expressions. The FSM compiles the data mapping structures only during Pipeline Manager startup because the rules can contain many comparisons of mapping patterns; this impacts startup performance. You can optimize the DAT_USC_Map module to enable Pipeline Manager to serialize the data structures and restore them from the serialized format.

About Precompiling USC Mapping Rules

Not all USC mapping data is stored in a compiled format; for example, rules used to define zone models. When the DAT_USC_Map module loads event data, it reorganizes it according to zone models to enable faster searching of the data structures during run time. This increases load time and memory requirements. To reduce the impact, you can configure Pipeline Manager to serialize the data structures the first time they are loaded and then reuse the serialized version during subsequent startup operations.

When Pipeline Manager begins processing data for a given zone model, it checks to see if a precompiled data file exists for that zone model. If so, it prepares the complex data structure by using the serialized format rather than by recompiling the structure from the USC map data.

If you enable the precompiling functionality, the following data is serialized:

- USC group
- Usage class and usage type
- Service code and service class
- Wholesale zone and retail zone

**Note:** Data that is not in the precompiled format is read from the database or file system, depending on your DAT_USC_Map module configuration. See "DAT_USC_Map" in BRM Configuring Pipeline Rating and Discounting.

For more information, see "Precompiling Usage Scenario Mapping Data".
About Filtering Mapping Rules

You use USC groups to assemble the rules that define which services and service configurations are available to the pipeline; they contain the rules for mapping the service EDR attributes to each usage class.

You can configure your system to filter mapping rules based on USC groups so only the rules in the USC groups you specify are compiled and loaded into the DAT_USC_Map module. All other rules are ignored. This is more efficient than having one zone model that uses a large number of rules.

Note: This is necessary only when your USC mapping rules are stored in the database; if they are read from a file, the data is already organized according to USC groups.

Generally you define USC Groups to contain the mapping rules for a specific type of EDR processing. For example, say you rate telephony services and process EDRs by using three USC groups (GSM, SMS, and GPRS), each of which contains mapping rules to determine domestic standard charges, domestic roaming charges, and international charges.

To increase performance, you can define the mapping rules for each set of charges in a separate zone model. Then, when an EDR is processed, based on the USC group specified, only the rules used in those zone models are compiled and loaded. This increases startup performance.

For more information, see "Filtering the Mapping Data to Compile and Load".

For information on USC groups, see "About Usage Scenario Mapping" in BRM Setting Up Pricing and Rating.

Configuring the DAT_USC_Map Module for Startup Performance

You improve startup performance of the DAT_USC_Map module by:

- Increasing the number of threads used to load mapping data.
- Precompiling usage scenario mapping data.
- Filtering the mapping data to compile and load.

You define these configurations in the Pipeline Manager registry file. For more information, see "About Configuring Pipeline Manager".

Increasing the Number of Threads Used to Load Mapping Data

The DAT_USC_Map module loads mapping rules for each zone model in a USC group by using a separate thread; therefore, it is only necessary to increase the number of threads when your USC groups contain multiple zone models.

To use multiple threads, set the NumberOfThreads entry to the desired number of threads. This enables Pipeline Manager to compile data in parallel and to restore it from the precompiled data files.

For example:

```
NumberOfThreads = 4
```

The default is 1.
Precompiling Usage Scenario Mapping Data

To enable precompiling of USC mapping data, set the `PreCompiledDataDir` entry. This entry both enables the precompile functionality and defines the location of the compiled data files. By default, compiled data files are saved in the `./compiled_usc_data` directory.

Pipeline Manager saves them with the following naming convention:

`USCzoneModelID.pc`

For example, `GSM.pc`, `GSM_DOMESTIC.pc`, and `GSM_ROAMING.pc`.

If this entry is set, compiled files are created the next time the pipeline starts. For each subsequent run, the data files are validated against the data structures in the database and, if necessary, recompiled and resaved to the file system.

**Note:** You can use this entry as a semaphore.

Filtering the Mapping Data to Compile and Load

If the source for your USC mapping rules is the database rather than a file, you can filter which rules are compiled and loaded into the DAT_USC_Map module when a pipeline starts by setting the `UscGroups` entry to one or more USC groups. For example:

```
UscGroups {GSM GSM_ROAMING}
```

**Important:** You can specify only one USC group per each pipeline running in your system. If you use multiple USC groups, you must configure Pipeline Manager to run multiple pipeline instances. To do this, configure the Pipeline Manager registry file so the FCT_USC_Map in each pipeline instance refers to the appropriate DAT_USC_Map module reference and `UscGroups` entry. For more information, see "About Configuring Pipeline Manager".

By default, all mapping rules are loaded into the pipeline. For more information, see "About Filtering Mapping Rules".

**Note:** You can use this entry as a semaphore.

Using Semaphores

You can use the new `NumberOfThreads`, `PreCompiledDataDir`, and `UscGroups` registry entries as semaphore to configure and control Pipeline Manager during pipeline startup. These semaphores perform the same tasks that the `Reload` semaphore performs, as specified in the startup registry or last-processed semaphore:

1. Load mapping data from the source (`Database` or `File`).
2. Create the USC zone model (from data in the `PreCompiledDataDir` or `USCMAPFile`).

**Note:** You can use this entry as a semaphore.
3. Compile or precompile each USC zone model.

When you change the values of these semaphores after startup, they are not updated automatically in your system; you must use the `Reload` semaphore to update them during run time.

For example:

- To use multiple threads to load data, edit the `NumberOfThreads` semaphore and then call the `Reload` semaphore. Each thread processes a different zone model when loading the USC data.
- To reload USC data using a different set of files in the `PreCompiledDataDir` directory, edit the `PreCompiledDataDir` semaphore and then call the `Reload` semaphore.
- To filter a different set of mapping rules, edit the `UscGroups` semaphore and then call the `Reload` semaphore.

For more information on the `DAT_USC_Map` semaphore entries, see "Semaphore File Entries" in BRM Configuring Pipeline Rating and Discounting.

Other Pipeline Manager Monitoring Tools

This section describes additional Pipeline Manager performance monitoring tools.

Viewing the Pipeline Log

You can see the results of tests for each pipeline in that pipeline’s `pipeline.log` file.

---

**Tip:** Open each log file in a terminal windows and run `tail -f` on the logs.

---

After each batch stream is processed, the pipeline writes the following information to the `pipeline.log` files:

- The number of processed EDRs.
- The number of errors that occurred during EDR processing.
- The number of EDRs processed per second for a stream.
- If instrumentation is on, the instrumentation results. See "Viewing Instrumentation Testing Results".

Tuning Tips

- Let the system process a few files before you measure performance. This assures that any additional memory needed (for example, for the buffers) has been allocated.
- Use the system monitor tool to monitor system utilization.

Configuring Buffer Size Polling

Use the `QueueRequestTimeout` Controller entry in the registry to specify the interval in seconds that each queue’s fill status is written to the log. For example:

```bash
ifw
{
    Active = TRUE
```
OS-Specific Pipeline Manager Monitoring Tools

Optimizing Pipeline Manager Performance

15-39

ProcessLoopTimeout = 10
QueueRequestTimeout = 10  # Optional, 0 disables
...

The default is 0 (no polling).
Buffer fill status information can indicate which function pool is the slowest. Over time, buffers in front of the slowest function pool fill up, and those that occur later in the stream are empty.

Note: Instrumentation is the recommended tool for identifying the slowest function pool. See "Measuring System Latencies with Instrumentation".

OS-Specific Pipeline Manager Monitoring Tools

This section describes OS-specific tools that you can use to monitor and maintain your Pipeline Manager system.

Solaris Monitoring Tools

This section describes Solaris monitoring tools.

Displaying Thread Details

To display details of the threads within a process, use the `prstat` utility:

```
prstat -Lmv -p
```

Example output:

```
prstat -Lmv -p 22376

PID USERNAME USR SYS TRP TFL DFL LCK SLP LAT VCX ICX SCL SIG PROCESS/LWPID
22376 integ 86 13 0.0 0.0 0.0 0.0 0.0 0.9 12 3K .1M 0 ifw/4
22376 integ 61 34 0.0 0.0 0.0 0.5 2.8 2.0 298 1K 64K 0 ifw/16
22376 integ 52 0.8 0.0 0.0 0.0 52 0.0 1.8 158 1K 7K 0 ifw/117
22376 integ 43 3.6 0.0 0.0 0.0 42 0.0 4.9 56 11K 11K 0 ifw/117
22376 integ 22 0.1 0.0 0.0 0.0 75 0.0 2.6 393 125 463 0 ifw/115
22376 integ 21 0.1 0.0 0.0 0.0 77 0.0 2.6 89 357 412 0 ifw/115
...

Total: 1 processes, 48 lwps, load averages: 4.18, 1.94, 2.38
```

In the pipeline `process.log` file, you can see when a thread is created, its name, and corresponding OS number:

```
09.12.2003 19:54:38 igscoll1 ifw IFW NORMAL 00000 -
...
09.12.2003 20:01:31 igscoll1 ifw IFW NORMAL 00000 -
(ifw.Pipelines.GSM.Input) Thread instance ID '16'; and Name 'ifw.Pipelines.GSM.Input'.
...
09.12.2003 21:38:40 igscoll1 ifw IFW NORMAL 00000 -
(ifw.Pipelines.GSM.Functions.PreProcessing2) Thread instance ID '135'; and Name 'ifw.Pipelines.GSM.Functions.PreProcessing2'.
...
```
Dumping the Call Stack of an Active Process
To dump the call stack of an active process, use the `vmstat` and `mpstat pstack` utilities.

Identifying Thread Functions
To identify what each thread is used for, such as the pipeline framework, input, output, or function modules, use the `mpstat` utility:

```
mpstat pstack
```

---

**Tip:** To determine the correlation between pipeline thread names and thread numbers, see the `pipeline.log` file.

---

**HP-UX IA64 Monitoring Tools**
Tools useful for monitoring Pipeline Manager on HP-UX IA64 systems include:
- `glance`
- `sar -AM`
- `top`
- `vmstat`
- `iostat`

For more information on these and other tools, see your HP documentation.

**Linux Monitoring Tools**
Tools useful for monitoring Pipeline Manager on Linux systems include:
- `vmstat`
- `sar`
- `top`
- `pmap`
- `gnome-system-monitor`
- `sysstat` package (`iostat, mpstat, sadc, and sar`)

**AIX Monitoring Tools**
Tools useful for monitoring Pipeline Manager on AIX systems include:
- `sar`
- `vmstat`
- `iostat`
- `filemon`
- `topas`
- `trace`
- `svmon`
- `netpmon`
This document describes cache residency distinction, which enables you to optimize your Oracle Communications Billing and Revenue Management (BRM) system for rating prepaid and postpaid accounts in real time and batch.

It assumes you have installed the components necessary for rating prepaid and postpaid accounts and have set up real-time and pipeline batch rating. For more information, see "Putting Together Your BRM System" in BRM Installation Guide.

Before reading this document, you should be familiar with the following:

- Basic BRM concepts. See "Introducing BRM" in BRM Concepts.
- BRM system architecture. See "BRM System Architecture" in BRM Concepts.
- Pricing and rating configuration. See "About Creating a Price List" in BRM Configuring Pipeline Rating and Discounting.
- Basic database administration concepts. See your database documentation.

**About Convergent BRM Systems**

A BRM system that handles prepaid and postpaid subscribers and rates events using real-time and batch processing is referred to as a convergent system.

To allow both types of subscriber accounts and provide both types of rating, you need to set up a BRM system with real-time and batch rating for prepaid and postpaid services. Then you need to ensure the following behaviors:

- Only the data relevant to Pipeline Manager is stored in its cache to improve performance and reduce memory requirements.
- The data stored in or updated by Pipeline Manager is synchronized with the database to maintain data integrity.

**About Cache Residency**

To perform rating, BRM caches subscriber data, such as data from /account, /billinfo, and /service objects as shown in Figure 16–1. Objects associated with prepaid subscribers are stored in the BRM database. Objects associated with postpaid subscribers are stored in Pipeline Manager memory. Objects associated with convergent subscribers are stored in both.

You can configure your BRM system so only the required objects are loaded into the Pipeline Manager memory after the object has been created or changed in the
database. This reduces the load time during initialization and data synchronization operations and minimizes memory size.

*Figure 16–1  BRM Cache Residency*

After events are processed and objects are recorded in the database, the Pipeline Manager memory is updated with the data from the database to maintain data integrity. Likewise, after Pipeline Manager processes events and stores them in the Pipeline Manager memory, the BRM database is updated whenever you run RE Loader.

**How Cache Residency is Determined**

BRM objects have a RESIDENCY_TYPE attribute that identifies how the reside in the BRM system. This value is defined in the data dictionary. For example, all BRM objects have a RESIDENCY_TYPE attribute value of 0, which indicates that the objects are owned by the BRM database.

BRM uses the residency type value to determine which objects to retrieve during requestion operations. Objects that are updated during batch rating are cached in Pipeline Manager. The database is updated with all data changed during both real-time and batch processing.

You can configure which objects are eligible for Pipeline Manager memory when they are updated by associating them with prepaid, postpaid, or convergent business profiles and setting up validation rules.

The following objects are valid for cache residency distinction:

- /account
- /balance_group
- /billinfo
- /group
- /group/sharing
- /group/sharing/charges
About Convergent BRM Systems

- /group/sharing/discounts
- /ordered_balgrp
- /profile
- /profile/acct_extrating
- /profile/serv_extrating
- /purchased_discount
- /purchased_product
- /service
- /uniqueness

Note: You cannot configure the /account object or the /uniqueness object with a specific cache type. Account object instances receive their cache type values from the associated /billinfo object, and uniqueness object instances receive their cache type values from the associated services.

For information on how to configure non-reference objects for cache residency distinction, see "Configuring Nonreference Objects for Cache Residency Distinction".

About Setting Up Business Profiles for Cache Residency Distinction

You can use the following business profiles listed in Table 16–1, together with the CacheResidency NameValue entry in an object’s validation template, to designate which object instances are loaded when they are created or changed:

<table>
<thead>
<tr>
<th>Business profile</th>
<th>CacheResidency</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Convergent</td>
<td>DEFAULT (0)</td>
<td>Object instances are loaded into batch pipeline memory. These objects are used for rating events in both real time and batch and contain both prepaid and postpaid subscriber information.</td>
</tr>
<tr>
<td>Prepaid</td>
<td>REALTIME (1)</td>
<td>Object instances reside in the BRM database only. These objects are used for rating events in real time and contain prepaid subscriber information.</td>
</tr>
<tr>
<td>Postpaid</td>
<td>BATCH (2)</td>
<td>Object instances are loaded into batch pipeline memory. These objects are used for rating events in batches and contain postpaid subscriber information.</td>
</tr>
<tr>
<td>Nonusage</td>
<td>DBONLY (3)</td>
<td>Object instances reside in the BRM database. These objects do not generate any usage and hence do not need to be loaded into Pipeline Manager memory.</td>
</tr>
</tbody>
</table>

When an object is created or modified, based on the business profile to which it is associated, BRM reads the validation template to determine the cache residency value and sets the object’s PIN_FLD_OBJECT_CACHE_TYPE value accordingly. The object instance is updated in the memory cache.

Important: The validation templates for a business profile must have compatible cache type values for all objects related to the /billinfo object.
About Convergent BRM Systems

Note: The account object does not have a validation template. Account object instances receive their cache type values from their associated /billinfo object. All objects outside the /billinfo context and associated with the account receive their cache type values from the account. For example, an account-level product has the same PIN_FLD_OBJECT_CACHE_TYPE value as its owner account.

If you do not have cache residency distinction enabled, or do not have a business profile assigned to a reference object, BRM assigns it a Convergent business profile. This sets the cache type to DEFAULT and ensures data is loaded into Pipeline Memory memory. You can change the default setting by modifying the /config/business_params object. See "Changing the Default Business Profile".

For information on setting up business profiles and validation templates, see "Managing Business Profiles" in BRM Managing Customers.

How BRM Caches Objects in a Convergent Rating System

Before loading data into Pipeline Manager memory, BRM determines the PIN_FLD_OBJECT_CACHE_TYPE value of each object and validates that any related objects have valid cache types. For example, if a /service/GSM object has a cache type value of REALTIME, any /purchase_product objects associated with that service must have a value of REALTIME or DEFAULT.

During initialization:
- Objects with a BATCH cache value are loaded into Pipeline Manager memory. These include objects used for rating postpaid events.
- Objects with a DEFAULT cache value are loaded into the database and Pipeline Manager memory. These include objects used for rating both prepaid and postpaid events.

During pipeline rating, the following occurs:

1. The FCT_BillingRecord module retrieves the ObjectCacheType value of a balance group from the DAT_BalanceBatch module (see BRM Configuring Pipeline Rating and Discounting) and publishes it in the EDR. The value can be BATCH (postpaid events) or DEFAULT (convergent events).
2. Rated Event (RE) Loader loads objects with BATCH and DEFAULT cache types into the database.

About Changing the Cache Type of an Object

You can change the cache type of an object by associating it with a business profile whose validation template specifies a different cache type value.

Table 16-2 shows what happens when you change an object’s cache type value:
### Table 16-2 Cache Type Change Ramifications

<table>
<thead>
<tr>
<th>Old cache type</th>
<th>New cache type</th>
<th>How BRM handles the change</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prepaid</td>
<td>Postpaid</td>
<td>All subsequent object instances are loaded only into the DAT_AccountBatch and DAT_BalanceBatch modules. The data is loaded into DAT_BalanceBatch immediately; however, the data is loaded into DAT_AccountBatch on the next business event processing for that account or service.</td>
</tr>
<tr>
<td>Prepaid</td>
<td>Convergent</td>
<td>All subsequent objects are loaded into the DAT_AccountBatch and DAT_BalanceBatch modules. The data is loaded into DAT_BalanceBatch immediately; however, the data is loaded into DAT_AccountBatch on the next business event processing for that account or service.</td>
</tr>
<tr>
<td>Prepaid</td>
<td>Nonusage</td>
<td>No change occur.</td>
</tr>
<tr>
<td>Postpaid</td>
<td>Prepaid</td>
<td>The object instances previously loaded into the DAT_AccountBatch and DAT_BalanceBatch modules remain in the memory until the next Pipeline Manager initialization.</td>
</tr>
<tr>
<td>Postpaid</td>
<td>Convergent</td>
<td>The object instances previously loaded into the DAT_AccountBatch and DAT_BalanceBatch modules are reloaded.</td>
</tr>
<tr>
<td>Postpaid</td>
<td>Nonusage</td>
<td>The object instances previously loaded into the DAT_AccountBatch and DAT_BalanceBatch modules are reloaded, and all subsequent objects are not loaded in Pipeline Manager memory from the next initialization of Pipeline Manager.</td>
</tr>
<tr>
<td>Convergent</td>
<td>Prepaid</td>
<td>The object instances previously loaded into the DAT_AccountBatch and DAT_BalanceBatch modules remain in the memory until the next Pipeline Manager initialization. The data will not be loaded from the next initialization of Pipeline Manager.</td>
</tr>
<tr>
<td>Convergent</td>
<td>Postpaid</td>
<td>All subsequent object instances are reloaded into the DAT_AccountBatch and DAT_BalanceBatch modules.</td>
</tr>
<tr>
<td>Convergent</td>
<td>Nonusage</td>
<td>The object instances previously loaded into the DAT_AccountBatch and DAT_BalanceBatch modules are reloaded. The data will not be loaded from the next initialization of Pipeline Manager.</td>
</tr>
<tr>
<td>Nonusage</td>
<td>Prepaid</td>
<td>No changes occur.</td>
</tr>
<tr>
<td>Nonusage</td>
<td>Postpaid</td>
<td>The object instances are loaded into the DAT_AccountBatch and DAT_BalanceBatch modules. The data is loaded into DAT_BalanceBatch immediately; however, the data is loaded into DAT_AccountBatch on the next business event processing for that account or service.</td>
</tr>
<tr>
<td>Nonusage</td>
<td>Convergent</td>
<td>The object instances are loaded into the DAT_AccountBatch and DAT_BalanceBatch module. The data is loaded into DAT_BalanceBatch immediately; however, the data is loaded into DAT_AccountBatch on the next business event processing for that account or service.</td>
</tr>
</tbody>
</table>

**Note:** Any cache residency value change is not reflected immediately in the DAT_AccountBatch module of Pipeline Manager. The data is loaded only during the subsequent business event processing for that account or service. If you change the cache residency value to a value that is not configured for batch loading, Pipeline Manager does not unload any data. However, the data is not loaded during subsequent Pipeline Manager initialization.
You change an object’s cache type by calling the PCM_OP_CUST_CHANGE_BUSINESS_PROFILE opcode. For more information, see "Changing a Bill Unit’s Business Profile" in BRM Managing Customers.

To configure BRM to handle all objects as postpaid, do not call PCM_OP_CUST_CHANGE_BUSINESS_PROFILE for each object. Instead, do the following:

- Disable the cache residency distinction functionality. See "Enabling Cache Residency Distinction", but in step 3, set the CacheResidencyDistinction value to disabled.
- In the wireless.reg registry file, set the ISC_ObjectCacheTypeOutputSplitter iScript’s Active entry to False. (See BRM Configuring Pipeline Rating and Discounting.)

### Configuring BRM for Cache Residency Distinction

The procedures in this section describe how to configure BRM for cache residency distinction.

- Enable cache residency distinction functionality. See "Enabling Cache Residency Distinction".
- Assign objects a cache residency value. See "Assigning Objects a Cache Residency Value".
- Set up cache residency validation rules. See "Setting Up Cache Residency Validation Rules".
- Set the default cache type for the system. See "Changing the Default Business Profile".
- Override the default business profile. See "Overriding the Default Business Profile".

### Enabling Cache Residency Distinction

By default, cache residency distinction is disabled. You enable it by modifying a field in the billing instance of the /config/business_params object.

You modify the /config/business_params object by using the pin_bus_params utility. See BRM Developer’s Guide.

1. Use the following command to create an editable XML file from the billing instance of the /config/business_params object:

   ```bash
   pin_bus_params -r BusParamsBilling bus_params_billing.xml
   ```

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

2. Search the XML file for following line:

   ```xml
   <CacheResidencyDistinction>disabled</CacheResidencyDistinction>
   ```

3. Change disabled to enabled.
Configuring BRM for Cache Residency Distinction

4. Save and close the file.

5. Use the following command to load the change into the \config\business_params object:

   ```
   pin_bus_params bus_params_billing.xml
   ```

   You should execute this command from the BRM_Home\sys\data\config directory, which includes support files used by the utility. To execute it from a different directory, see `pin_bus_params` in BRM Developer’s Guide.

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

   For more information, see the following topics in BRM Developer’s Guide:
   - Using `testnap`
   - Reading Objects by Using Object Browser

7. Stop and restart the Connection Manager (CM). For more information, see "Starting and Stopping the BRM System".

8. For multiple databases, run the `pin_multidb` script with the `-R CONFIG` parameter. For more information on this script, see "pin_multidb".

Assigning Objects a Cache Residency Value

To assign a cache residency value to an object, associate it with a prepaid, postpaid, or convergent business profile and define cache validation rules.

For information on setting up business profiles, see "Setting Up Business Profiles and Validation Templates" in BRM Managing Customers.

For information on cache residency validation rules, see "Setting Up Cache Residency Validation Rules".

Setting Up Cache Residency Validation Rules

You define the PIN_FLD_OBJECT_CACHE_TYPE field value by associating an object with a prepaid, postpaid, or convergent business profile and setting the CacheResidency value in an object’s validation template as shown in Table 16–3.

<table>
<thead>
<tr>
<th>Business profile</th>
<th>CacheResidency key</th>
<th>CACHE_TYPE field value</th>
<th>Memory cache</th>
</tr>
</thead>
<tbody>
<tr>
<td>Convergent</td>
<td>DEFAULT</td>
<td>0</td>
<td>BRM database, DAT_AccountBatch, DAT_BalanceBatch</td>
</tr>
<tr>
<td>Prepaid</td>
<td>REALTIME</td>
<td>1</td>
<td>BRM database only</td>
</tr>
<tr>
<td>Postpaid</td>
<td>BATCH</td>
<td>2</td>
<td>DAT_AccountBatch, DAT_BalanceBatch</td>
</tr>
</tbody>
</table>

Caution: BRM uses the XML in this file to overwrite the existing billing instance of the \config\business_params object. Changing any parameter in this file affects the associated aspects of the BRM billing configuration.
When you set up a business profile, each validation template you associate with it must have a compatible cache residency value for all objects related to the /billinfo object. For example, if you set up a prepaid business profile with validation templates for the /billinfo, /balance_group, and /group/sharing/charges objects, the cache residency value in each object’s validation template must be set to REALTIME, DEFAULT, DEFAULT_INACTIVE, or REALTIME_INACTIVE.

For specific instructions on how to set up business profiles and validation templates for objects, see “Setting Up Business Profiles and Validation Templates” in BRM Managing Customers.

### Changing the Default Business Profile

By default, all reference object instances get assigned an object cache type value of DEFAULT (Convergent business profile) if they have not been associated with a business profile.

To assign a REALTIME or BATCH object cache type to object instances when they are made, change the default business profile for BRM. This value is defined in the billing instance of the /config/business_params object.
Modify this object by using the pin_bus_params utility. See pin_bus_params in BRM Developer’s Guide.

1. Use the following command to create an editable XML file from the billing instance of the /config/business_params object:

   pin_bus_params -r BusParamsBilling bus_params_billing.xml

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

2. Search the XML file for following line:

   <DefaultBusinessProfile>Convergent</DefaultBusinessProfile>

3. Change Convergent to one of the following:
   - **Prepaid**: Sets the default cache type of all new objects to REALTIME (1).
   - **Postpaid**: Sets the default cache type of all new objects to BATCH (2).

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

4. Save and close the file.

5. Use the following command to load the change into the /config/business_params object:

   pin_bus_params bus_params_billing.xml

   You should execute this command from the BRM_Home/sys/data/config directory, which includes support files used by the utility. To execute it from a different directory, see pin_bus_params in BRM Developer’s Guide.

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

   For more information, see the following topics in BRM Developer’s Guide:
   - Using testnap
   - Reading Objects by Using Object Browser

7. Stop and restart the Connection Manager (CM). For more information, see “Starting and Stopping the BRM System”.

---

**Note:** Before you change the default business profile, define and load the business files into the database. See "Setting up Business Profiles and Validation Templates" in BRM Managing Customers.

**Important:** If the business profile you set as the default business profile has not been loaded into the database or is not valid, the Convergent business profile will be used.

---

**Important:** If the business profile you set as the default business profile has not been loaded into the database or is not valid, the Convergent business profile will be used.

---

**Note:** Before you change the default business profile, define and load the business files into the database. See "Setting up Business Profiles and Validation Templates" in BRM Managing Customers.
8. For multiple databases, run the **pin_multidb** script with the `-R CONFIG` parameter. For more information on this script, see "pin_multidb".

**Overriding the Default Business Profile**

To override the default business profile that is assigned to an object during account creation, pass the business profile POID in the input flist of the PCM_OP_CUST_COMMIT_CUSTOMER opcode.

**About Selective Account Loading**

Pipeline Manager loads the subscriber data based on the service types configured for batch rating. If the service type is the same for both the prepaid and the postpaid subscribers, Pipeline Manager loads the prepaid subscriber data also.

You can configure your BRM system to load subscriber data selectively in Pipeline Manager based on the business profiles assigned to the accounts. For example, if you use selective account loading, you can load only data for postpaid services instead of postpaid and prepaid data, even though the service type is the same. You can configure any cache residency type data to be loaded into Pipeline Manager memory.

Selective account loading in Pipeline Manager provides:

- Reduced load time during initialization because less data is retrieved from the database.
- Improved memory usage because only selective subscriber information is stored in memory.

When rating the CDRs, Pipeline Manager treats the data as valid only if the cache residency value of the data at the time of the event matches with the values configured for loading data into Pipeline Manager.

**Configuring Pipeline Manager for Selective Account Loading**

You can configure Pipeline Manager to load selective accounts during initialization by enabling selective account loading functionality. See "Enabling Selective Account Loading".

---

**Note:** For selective account loading functionality, you must enable the cache residency distinction parameter. See "Enabling Cache Residency Distinction".

---

**Enabling Selective Account Loading**

By default, selective account loading functionality is disabled. You can enable this functionality by loading and configuring an optional business parameter, `CacheResidenciesForBatchPipeline`, in the `BRM_Home/sys/data/config/bus_params_selective_loading.xml` file.

To load and configure the values in the `CacheResidenciesForBatchPipeline` business parameter:

1. Search the `bus_params_selective_loading.xml` file for following line:

```
<CacheResidenciesForBatchPipeline>0,1</CacheResidenciesForBatchPipeline>
```
2. Change 0,1 to any cache residency values of accounts you want to load, separated by comma. For example, to load convergent, prepaid, and postpaid accounts into Pipeline Manager, change 0,1 to 0,1,2.

3. Save and close the file.

4. Use the following command to load the change into the /config/business_params object:

   ```
   pin_bus_params bus_params_selective_loading.xml
   ```

   You should execute this command from the BRM_Home/sys/data/config directory, which includes support files used by the utility. To execute it from a different directory, see `pin_bus_params` in BRM Developer’s Guide.

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

   For more information, see the following topics in BRM Developer’s Guide:
   - Using testnap
   - Reading Objects by Using Object Browser

6. Stop and restart the CM. For more information, see "Starting and Stopping the BRM System".

7. Stop and restart Pipeline Manager.

8. For multiple databases, run the `pin_multidb` script with the -R CONFIG parameter. For more information on this script, see "pin_multidb".

The following is a sample `bus_params_selective_loading.xml` file:

   ```xml
   <BusParamConfigurationClass>
     <BusParamsSelectiveLoading>
       <CacheResidenciesForBatchPipeline>0,1,2</CacheResidenciesForBatchPipeline>
     </BusParamsSelectiveLoading>
   </BusParamConfigurationClass>
   ```

   Here, 0,1,2 specifies the cache residency types DEFAULT, REALTIME, and POSTPAID. After the `CacheResidenciesForBatchPipeline` business parameter is loaded, Pipeline Manager loads all accounts with Convergent, Prepaid, and Postpaid business profiles.

### Configuring Pipeline Manager to Process Prepaid CDRs

By default, Pipeline Manager rates only one event type per service; for example, delayed session event for GSM telephony service.

If the selective account loading functionality is enabled, you can load prepaid subscribers in Pipeline Manager. However, Pipeline Manager rejects any prepaid CDRs of the prepaid subscribers if the delayed event type configured for batch rating is not present in any of the products owned by the service or account. This is because of the difference in the prepaid and postpaid event types. For example, real-time session event for prepaid events and delayed session event for postpaid events.

To allow the CustomerSearch module to accept the prepaid CDRs, you can use the FCT_Account module `DisableRatingProductCheck` registry entry to configure how product rating is checked:

- If you enable this entry, FCT_Account does not reject any prepaid CDRs of the prepaid subscribers if the configured event for batch rating is not present in any of
the products owned by the service or account. Pipeline Manager does not rate CDRs, but the DAT_AccountBatch plug-in provides the subscriber information. You can use this subscriber information for any customized processing. For example, if you want to pass rated roaming prepaid CDRs through Pipeline Manager, you can customize the action on the CDRs based on the subscriber information.

- If you disable this entry, the FCT_Account rejects any prepaid CDRs of the prepaid subscribers if the configured event for batch rating is not present in any of the products owned by the service or account. By default, DisableRatingProductCheck is set to False.

**Customizing Cache Residency Distinction**

You can customize cache residency by performing any of the following tasks:

- Configure non-reference storable classes or custom storable classes for cache residency distinction. See "Configuring Nonreference Objects for Cache Residency Distinction".

- Create, change, or delete business profiles. See "Setting Up Business Profiles and Validation Templates" in BRM Managing Customers.

- Create new validation templates (subclasses of the \config\template base class). See "Creating Custom Fields and Storable Classes" in BRM Developer's Guide.

- Customize the ISC_ObjectCacheTypeOutputSplitter iScript. See "ISC_ObjectCacheTypeOutputSplitter" in BRM Configuring Pipeline Rating and Discounting.

**Configuring Nonreference Objects for Cache Residency Distinction**

For specific instructions on how to create and modify storable classes, see "Creating Custom Fields and Storable Classes" in BRM Developer's Guide.

To configure non-reference objects:

1. Use Storable Class Editor to define a new class or modify an existing BRM class, and add the RESIDENCY_TYPE attribute and PIN_FLD_OBJECT_CACHE_TYPE field to the class. See Storable Class Editor Help.

2. Create a validation template for the new storable class. See "Setting Up Business Profiles and Validation Templates" in BRM Managing Customers.

3. Associate the new validation template with a business profile.

4. Add or modify iScript validation rules to determine the PIN_FLD_OBJECT_CACHE_TYPE value for related object instances.

**Important:** This is necessary to validate the cache residency value. Be certain to set the CacheResidency key value to Prepaid, Postpaid, or Convergent.

**Note:** BRM does not supply iScript rules for business profile validation. Instead, you must create your own rules. For information about creating iScript validation rules, see "Creating iScripts and iRules" in BRM Developer’s Guide.
5. Configure the iScript in the validation templates to return a list of object POIDs and the cache residency value for each one.

6. Customize the appropriate opcode to call the validation template. Perform a WRITE_FLDS operation on the object instances to set their PIN_FLD_OBJECT_CACHE_TYPE value.
Part IV describes how to troubleshoot an Oracle Communications Business and Revenue Management (BRM) system. It contains the following chapters:

- Resolving Problems in Your BRM System
- Reference Guide to BRM Error Codes
- Pipeline Manager Error Messages
Resolving Problems in Your BRM System

This document provides guidelines to help you troubleshoot problems with your Oracle Communications Billing and Revenue Management (BRM) system. You can find information about interpreting error messages, diagnosing common problems, and contacting Oracle Technical Support.

Before you read this document, you should be familiar with how BRM works. See "BRM System Architecture" in BRM Concepts.

For information on problems related to poor performance, see "Improving BRM Performance".

General Checklist for Resolving Problems with BRM

When any problems occur, it is best to do some troubleshooting before you contact Oracle Technical Support:

- You know your installation better than Oracle Technical Support does. You know if anything in the system has been changed, so you are more likely to know where to look first.

- Troubleshooting skills are important. Relying on Technical Support to research and solve all of your problems prevents you from being in full control of your system.

If you have a problem with your BRM system, ask yourself these questions first, because Oracle Technical Support will ask them of you:

- What exactly is the problem? Can you isolate it? For example, if an account causes Customer Center to crash on one computer, does it give the same result on another computer? Or if users cannot authenticate, is it all services or just one service? If it’s an IP problem, does it affect all users or just those on a specific POP or a specific type of terminal server?

Oracle Technical Support needs a clear and concise description of the problem, including when it began to occur.

- What do the log files say?

This is the first thing that Oracle Technical Support asks for. Check the error log for the BRM component you’re having problems with. If you’re having problems connecting to BRM, start with the log for the Connection Manager.

See "Using Error Logs to Troubleshoot BRM".

- Have you read the documentation?

Look through the list of common problems and their solutions in "Diagnosing Some Common Problems with BRM".
Using Error Logs to Troubleshoot BRM

BRM error log files provide detailed information about system problems. If you’re having a problem with BRM, look in the log files.

If you use HP OpenView, you can monitor a set of BRM and Pipeline Manager errors. You can also add errors to monitor. See "Monitoring Error Messages in Log Files by Using HP OpenView".

Log files include errors that need to be managed, as well as errors that do not need immediate attention (for example, invalid logins). To manage log files, you should make a list of the important errors for your system, as opposed to errors that do not need immediate attention. The errors monitored by HP OpenView are those that indicate a problem that needs to be fixed. See "Error Messages Monitored by HP OpenView".

Finding Error Log Files

BRM records system activity in a set of log files, one for each component or application. If a component of your BRM system has a problem, you can find the error message in the log file for that component. For information on locating a log file, see "Using Logs to Monitor Components".

Understanding Error-Message Syntax

BRM error messages use this syntax:

\[ \text{severity} \ \text{date\_&\_time} \ \text{host\_name} \ \text{program}:\text{pid} \ \text{file}:\text{line} \ \text{correlation\_id} \ \text{message} \]

where:

\textit{severity} is the severity of the error message. It can be one of three values:
Using Error Logs to Troubleshoot BRM

Resolving Problems in Your BRM System

■ **E:** Error. An error indicates that a component of your BRM system is not operating correctly. This is the most severe type of problem.

■ **W:** Warning. Warnings indicate that the system cannot perform a task because the database contains incorrect or inconsistent data. The system continues to operate, but you should investigate and resolve problems with the data immediately.

■ **D:** Debug. Debugging messages, which indicate problems with an application, are typically used by application developers to diagnose errors in custom applications. You see these messages only if you set error reporting to the highest level. See "Setting the Reporting Level for Logging Messages".

`date & time` is the date and time the message was logged.

`host_name` is the name of the computer generating the message. If several machines are sharing a log file using Network File System (NFS), or if all log files are stored in a central location, use this information to pinpoint the machine with the problem.

`program` is the name of the program (or process) generating the log message. This information helps resolve problems in billing utilities, for example, because all billing utilities use the same log file.

`pid` is the process ID for the process generating the log message. This information helps resolve problems in components, such as Connection Managers and Data Managers, that might have many processes running in parallel with the same name (`program`).

`file` is the name of the source file where the error was detected. Technical Support uses this information when diagnosing system problems.

`line` is the line number in the source file where the error was detected. Technical Support uses this information when diagnosing system problems.

`correlation_id` is the identifier for all error messages related to a single error occurrence. This information can be used to sort error messages and to identify the set of error messages generated from a single error occurrence.

`message` is a detailed description of the error condition. Part of the message often includes the error type, location, and code, which you can use to interpret the error. See "Reference Guide to BRM Error Codes".

Resolving Clusters of Error Messages

An error often produces a cluster of error messages in the log file. The Facilities Modules (FMs), especially, tend to generate cascading messages. To resolve the error, isolate the group of messages, as defined by their common `correlation ID`, `date/time`, and `process ID`, and look at the first one in the series. The error location for that message generally indicates the source of the problem. Then find the last message text in the first error, to identify the operation that was associated with the error. Always consider whether an error could have been caused by something happening in a downstream process.

Interpreting Error Messages

The following examples show the typical process for evaluating and interpreting error messages to resolve problems with BRM.

Example 1: Failure of a Client Application
A BRM client application fails and displays an error message.

- Look in the application’s log file. The file shows the following error message:
The message shows that:

- At 014:50:05 the system returned an error.
- The host name is *db2.corp*.
- The file name is *sample_app.c*.
- The line of code is 173.
- There was a problem creating an account.
- The error was first found in the Data Manager.

Check the Data Manager log file (*dm_oracle.pinlog*) for an error message that occurred at the same time and has the same correlation ID, in this case Fri Sep 12 14:50:5 and 1:CT1255:Account_Manager:1948:1684:63:1063403309:14:

```
E Fri Sep 12 14:50:05 2003 db2.corp:12602 sample_app.c:173
op_cust_create_acct error [location=pin_errloc_dm class= errno= field num=
recid=<0> reserved=<0>]
```

The error message shows an Oracle error, with the Oracle code 1653.

- Consult the Oracle documentation. Code 1653 indicates that there is a problem with growing an extent. Because the error message reported that BRM was unable to extend one of the tables, you can infer that the problem is that there is no more room in the database and you must increase its size, as explained in the Oracle documentation.

**Example 2: Problem with Customer Center**

When you try to add a new account in Customer Center, you see the following message:

A problem occurred while attempting to create an account. The error is likely related to some invalid field, missing required information, or duplicate information. Please check fields carefully, and try again.

- Check the *cm.pinlog* file, which shows the following error message:

```
qnty (200.000000) > max (1.000000)...  
E Mon Apr 18 12:41:17 2003 turnip cm:16798 fm_bill_purchase.c:196
op_bill_purchase error [location=<PIN_ERRLOC_FM:5> class=<PIN_ERRCLASS_APPLICATION:4> errno=<PIN_ERR_BAD_VALUE:46> field num=<PIN_FLD_PURCHASE_MAX:4,237> recid=<0> reserved=<0>]
```

The debugging message shows that the account creation routine is trying to purchase a product. The error indicates that the purchase quantity (200) is more than the allowed...
quantity of the product (1). Because the account creation processes use deals to purchase products, this error probably means that the deal has been defined incorrectly. Look in the price list to check the maximum allowable purchase amount for the product.

Example 3: Getting More Information from Error Numbers
You cannot start the Data Manager.

- Check the dm_oracle.pinlog file, which shows the following error message:

```
E THU Sep 11 00:30:49 2003 kauai dm:29349 dm_main.c(1.74):1723
1:CT1255:dm:28492:1:0:1063265316:0
DM master dm_die:"bad bind(2)", errno 125
```

This error message indicates that the Data Manager cannot initiate itself. Usually, errno followed by a number means that a system message is associated with this error. You can check the error file: /usr/include/sys/errno.h. In this case, error 125 is listed as “EADDRINUSE: Address already in use”. In other words, the DM process is trying to use a port that is already in use by another process.

Example 4: Getting More Information about Oracle Errors
An error in the application log file indicates the error location is the Data Manager.

- Check the dm_oracle.pinlog file, which shows the following error message:

```
E WED Aug 18 01:40:07 2003 kauai dm:402.354 dm_subr.c(1.80):481
1:CT1255:dm:28509:1:0:1061195411:7
ORACLE error: do_sql_insert: obndrv: code 1036, op 28, peo 0
=ORA-01036: illegal variable name/number
was binding ':poid_DB' buf 0x195b180, bufl 5, ftype 5
```

The message shows an Oracle error, number 1036, which you can investigate in the Oracle documentation by using the oerr command.

```
% oerr ora 1036
01036, 00000, "illegal variable name/number"
// *Cause: Unable to find bind context on user side
// *Action: Make sure that the variable being bound is in the sql statement.
```

The obndrv function is looking for the variable :poid_DB in the SQL statement, but the error says that it’s not there. For information on how to gather the SQL statements generated by the Oracle or Data Manager, see "Increasing the Level of Reporting for a DM".

**Diagnosing Some Common Problems with BRM**

This section lists some common problems with BRM and shows you how to diagnose the error messages and resolve the problems.

**Problems Starting BRM Components**

**Problem: Bad Bind, Error 13**

One of the log files (log or pinlog) for the Data Manager or Connection Manager has a reference to “bad bind” and “errno 13”.

**Cause**
The port number specified in the configuration file (dm_port or cm_ptr entry) is incorrect.

Another possibility is that the port number is below 1023, and the CM, CMMP, or DM was not started as root. System processes that use port numbers below 1023 must be started as root. If you use a port number greater than 1024, you do not have to start the process as root.

Solution

Edit the configuration file for the component to specify an unassigned port number above 1023, such as 1950.

Problem: Bad Bind, Error 125

The log file for the Data Manager or Connection Manager has a reference to “bad bind” and “errno 125”.

Cause

Duplicate port number. Some other process is already using the port.

Solution

Edit the configuration file for the component to specify an unassigned port number above 1023, such as 1950.

Problem: Cannot Connect to Oracle Database

When you look at the processes running, you see the master Oracle DM and front ends running, but no back end running.

Causes

- The database name configuration entry (sm_database) for the DM points to the wrong database name. (The error message shows which database name the DM is trying to connect to.)

- The Oracle password configuration entry (sm_pw) is missing.

- The Oracle tnsnames file is missing or incorrect.

- The oracle_sid or oracle_Home environment variable is set incorrectly.

- The Oracle DM is spawning too many back-end processes simultaneously for the IPC or BEQ protocol to handle.

Solutions

- Enter the correct database name and Oracle user name and password for the BRM database in the configuration file for the Oracle DM and restart the Data Manager.

- Create a valid Oracle tnsnames file and check the environment variables.

- If you are using the IPC or BEQ protocol, configure the Oracle DM to wait a specified amount of time before spawning or respawning a new back-end process. To do this, add the following entry to the Oracle DM configuration file (BRM_Home/sys/dm_oracle/pin.conf):

  - `dm dm_restart_delay DelayTime`

  **Note:** Adding a delay increases the Oracle DM startup time.
where \( \text{DelayTime} \) is the amount of time, in microseconds, the Oracle DM should wait before spawning a new back-end process. Set \( \text{DelayTime} \) to the smallest possible time that fixes your connection problems. As a guideline, start with 1000000 microseconds (1 second) and then decrease the time until you find the optimal setting for your system.

**Problem: ORA-01502: Index 'PINPAP.I_EVENT_ITEM_OBJ__ID' or Partition of Such Index Is in Unusable State**

While loading the CDRs using the direct path load option, an error stating that the index is in an unusable state occurs.

**Cause**

While IREL processes the CDRs using the direct path loading option, it updates the indexes. However, as the index is being updated, another application, for example, \texttt{pin_monitor_balance} would also access the same index partition.

**Solution**

Configure the \texttt{dm_sql_retry} entry in the \texttt{pin.conf} file. This is specified as an integer value that indicates the number of times an SQL statement is to be retried if this error occurs.

\begin{note}
This is not a mandatory parameter to be set in the \texttt{pin.conf} file. The default behavior is to not try running the SQL statement if the error occurs.
\end{note}

**Problems Stopping BRM Components**

**Problem: No Permission to Stop the Component**

You run the stop script, but the script fails. You find a reference to “permission denied” in the log file for the component.

**Cause**

You do not have permission to stop the BRM system.

**Solution**

Log in as \texttt{root} or as the user who started the BRM system.

**Problem: No pid File**

You run the stop script, but the script fails. You find a reference to “no pid file” in the log file for the component.

**Cause**

BRM cannot find the \texttt{.pid} file.

**Solution**

Identify the process ID for the component you want to stop, and then stop the process manually. See "Starting and Stopping the BRM System".

**Problems Connecting to BRM**
**Problem: Cannot Connect to the Database**

When you try to start a client application, you get an error message advising you of “problems connecting to the database.”

**Cause**

The CM might not be set to handle the number of current client sessions.

**Solution**

Set the `cm_max_connects` entry in the configuration file for the CM to a number larger than the number of client sessions you anticipate. Then restart the CM. See “Starting and Stopping the BRM System”.

**Problem: Cannot Connect to the CM**

An application cannot connect to BRM, and the log file for the application (which might be `default.pinlog` in the current directory) shows the error “PIN_ERR_NAP_CONNECT_FAILED(27).”

**Causes**

- The configuration file (`pin.conf`) for the application might be pointing to the wrong CM.
- The CM is not running.
- The CM is not set to handle this many connections.
- No TCP sockets are available on the client or CM machine, perhaps because you used many sockets recently and the sockets haven’t been released from their 2-minute wait period after the connections were closed.

**Solutions**

- Open the configuration file for the application and check the entries that specify the CM.
- Check for CM processes. See "Checking the Number and ID of a BRM Process”.
- Set the `cm_max_connects` entry in the configuration file for the CM to a number larger than the number of application sessions you anticipate. Then restart the CM.
- Wait a few minutes to see if the sockets are freed up.

**On Solaris:** To see how many sockets are available:

```
netstat -n -f inet -p tcp | wc -l
```

**On HP-UX IA64:** To see how many sockets are available:

```
netstat -n -f inet | grep `^tcp | wc -l
```

**On Linux:** To see how many sockets are available:

```
netstat -n -A inet -t | wc -l
```

**On AIX:** To see how many sockets are available:

```
netstat -n -f inet | grep tcp
```

If the resulting number is close to 65535, there are too many socket connections for a single IP address on this machine.
Problem: CM Cannot Connect to a DM
You might find a message similar to the following:

DMfe #3: dropped connect from 111.122.123.1:45826, too full
W Thu Aug 06 13:58:05 2001 portalhost dm:17446 dm_front.c(1.47):1498

Cause
There are not enough connections allowed for the DM.

Solution
- Use the dm_max_per_fe parameter in the DM configuration file to increase the number of CM connections allowed.
- Install and configure an additional DM.

Problems with Deadlocking

Problem: BRM “Hangs” or Oracle Deadlocks
Your BRM system stops responding, or Oracle reports deadlocking messages.

Cause
The Data Manager might have too few back ends for the type of BRM activity.

Solution
Configure the Data Manager with more back ends. For example, provide at least two Data Manager back ends for each customer service representative. For more guidelines on setting the number of back ends, see "Improving Data Manager and Queue Manager Performance".

Problem: dm_oracle Cannot Connect to the Oracle Database
The Oracle Data Manager (dm_oracle) waits indefinitely for a response from the Oracle database.

Cause
If there is a problem with dm_oracle can hang when it attempts to connect to the database.

Solution
Set the database_request_timeout_duration parameter in the dm_oracle configuration file (BRM_Home/sys/dm_oracle/pin.conf):
- dm database_request_timeout_duration milliseconds

where milliseconds is the number of milliseconds the Data Manager (DM) will wait for a response. For example:
- dm database_request_timeout_duration 10000

If the database doesn’t respond during the wait period and you are using Oracle RAC, the DM times out and then makes one attempt to connect to another Oracle database. If this pin.conf parameter isn’t specified or is set to 0, connection attempt doesn’t time out.
Problems with Memory Management

**Problem: Out of Memory**
The Data Manager will not start, and the error log file for the DM refers to “bad shmget” or “bad shmat” and “errno 12.” Or, when the system is running, the Connection Manager or an application shows the error “PIN_ERR_NO_MEM” in its log file.

**Causes**
The DM or another queuing-based daemon did not have enough shared memory to complete the operation. This is caused by one or more of the following conditions:

- Other processes are using all of the shared memory.
- There are too many CM processes.
- There are memory leaks in the CM or its FMs.
- **On Solaris:** The shared memory segment allocated by one of the DM processes has not been cleaned up properly, leaving a sizeable chunk of memory allocated but unused. This condition, rare in normal operation, can be caused by the following activities:
  - Repeated starting and stopping of the system.
  - Stopping the DMs manually, especially by using `kill -9`.
- **On Solaris:** The shared memory configuration for the system is less than the shared memory set for BRM.

**Solutions**
To check for memory leaks, use `ps` with the `vsz` flag at two or more intervals to see changes in shared memory.

**On Solaris:**

```
ps -eo pid,vsz,f,s,osz,pmem,comm | egrep 'cmldml [application]'
```

**On HP-UX IA64:**

```
ps -eo pid,vsz,flags,state,sz,pmem,comm | egrep 'cmldml [application]'
```

**On Linux:**

```
ps -eo pid,vsz,f,s,sz,pmem,comm | egrep 'cmldml [application]'
```

**On AIX:**

```
ps -eo pid,vsz,dpgsz,THREAD,comm | egrep 'cmldml [application]'
```
For a CM, vsz should grow only until an operation has passed through the CM and then stay constant. For example, if vsz is growing during billing or RADIUS Manager operations, there is a memory leak.

To check for and clean up unused memory on Solaris:

1. Stop all Data Manager processes. See "Starting and Stopping the BRM System".
2. Confirm that there are no DM processes running. See "Checking the Number and ID of a BRM Process".
3. Run `df -k` to check swap space usage. Confirm that the available space is very low.
4. Run `ipcs -ma` to show the shared memory segments that have been allocated but not used recently. A shared memory segment is probably abandoned when you see the following conditions:
   - Number of attaches (NATTCH) is 0
   - KEY is 0 (and not using a special dm_shmkey)
   - Creator process ID (CPID) is gone
   - Last detach time (DTIME) has a value
5. Run `ipcrm -m` segment_id on each of the unused segments to free up the space.
6. Run `df -k` again to confirm that the available swap space has been cleared.
7. Stop and restart the DM processes. See "Starting and Stopping the BRM System".

To increase the system shared memory on Solaris, open the `/etc/system` file and set the `shminfo_shmmax` configuration parameter to a value greater than the value of `dm_shmsize` in the DM configuration file (`pin.conf`). Stop and restart the computer.

Example `/etc/system` file for a 64 MB system:

```
set shmsys:shminfo_shmmax=37748736
set shmsys:shminfo_shmmni=1
set shmsys:shminfo_shmmni=100
set shmsys:shminfo_shmsseg=10
set semsys:seminfo_semmns=200
set semsys:seminfo_semmni=70
```

In this example, the shared memory segment has been set to 36 MB (1048576 times 36).

To check for and clean up unused memory on Linux:

1. Stop all Data Manager processes. See "Starting and Stopping the BRM System".
2. Confirm that there are no DM processes running. See "Checking the Number and ID of a BRM Process".
3. Run `df -k` to check swap space usage. Confirm that the available space is very low.
4. Run `ipcs -ma` to show the shared memory segments that have been allocated but not used recently.
5. Run `ipcs -mac` to show the shared memory segments that have been allocated along with the corresponding user information.
6. Run `ipcs -mat` to show the shared memory segments that have been allocated detach timing information.
Diagnosing Some Common Problems with BRM

7. Run `ipcrm -m segment_id` on each of the unused segments to free up the space.
8. Run `df -k` again to confirm that the available swap space has been cleared.
9. Stop and restart the DM processes. See "Starting and Stopping the BRM System".

To increase the system shared memory on Linux, open the `/etc/sysctl.conf` file and set the `shminfo_shmmax` configuration parameter to a value greater than the value of `dm_shmsize` in the DM configuration file (`pin.conf`). Stop and restart the computer.

To check for and clean up unused memory on AIX:
1. Stop all Data Manager processes. See "Starting and Stopping the BRM System".
2. Confirm that there are no DM processes running. See "Checking the Number and ID of a BRM Process".
3. Run `df -k` to check swap space usage. Confirm that the available space is very low.
4. Run `ipcs -ma` to show the shared memory segments that have been allocated but not used recently.
5. Run `ipcs -mac` to show the shared memory segments that have been allocated along with the corresponding user information.
6. Run `ipcs -mat` to show the shared memory segments that have been allocated detach timing information.

7. Run `ipcrm -m segment_id` on each of the unused segments to free up the space.
8. Run `df -k` again to confirm that the available swap space has been cleared.
9. Stop and restart the DM processes. See "Starting and Stopping the BRM System".

**Problem: Java Out of Memory Error**

When using GUI applications such as Pricing Center, Suspense Manager, and Customer Center or batch applications such as Invoice formatter, you may sometimes receive “java.lang.OutOfMemoryError: Java heap space” error messages.

**Cause**
The Java application does not have enough memory to complete the operation.

Solution

Increase the maximum heap size used by the Java Virtual Machine (JVM). The exact amount varies greatly with your needs and system resources.

The heap size is controlled by the `-Xmx` size entry in the Java application startup script. By default, the `-Xmx` size entry is not present in the startup line. To increase the maximum heap size, add this entry and a number (in megabytes) to the application startup line. The following example adds a 1024 MB maximum heap size to the class:

```
java -Xmx1024m class
```

---

**Note:** Increasing the heap size may degrade the performance of other processes if insufficient resources are available. You must adjust the heap size based on your application needs and within your system’s limits.

---

**Problem: Memory Problems with the Oracle DM**

The error log file for the Data Manager (DM) for your Oracle database refers to “No memory for...”, such as “No memory for list in pini_flist_grow.” You suspect memory problems, but your system has sufficient memory for the environment.

**Cause**

The DM is not configured to use sufficient shared memory.

**Solution**

1. Open the Data Manager configuration file (`BRM_Home/sys/dm_oracle/pin.conf`).
2. Increase the size of the `dm_bigsize` and `dm_shmsize` parameters. Follow the guidelines in the configuration file for editing these entries.
3. Save the configuration file.
4. Stop and restart the DM.

---

**Problems Running Billing**

**Problem: Billing Daemons Are Running, but Nothing Happens**

Even though the billing processes are running, BRM isn’t producing billing data.

**Cause**

There are too few back ends for the Data Manager (DM). Because billing daemons run in parallel, you must have at least one DM back end for each billing program thread, plus one back end for the master thread searches.

**Solution**

Edit the `dm_n_be` entry in the Data Manager configuration file (`pin.conf`) to add more back ends to the DM, and then stop and restart the DM. See "Configuring DM Front Ends and Back Ends".
**Problem: High CPU Usage for the Number of Accounts Processed**

Running the billing scripts puts an inordinately heavy load on the computer, and processing the accounts takes a long time.

**Cause**

An index is missing or unbalanced; or, in Oracle, an index is in the CHOOSE Optimizer mode and statistics are out of date.

**Solution**

Rebuild the BRM indexes before you run the billing scripts. See "Rebuilding Indexes" in *BRM Installation Guide*.

**Problems Creating Accounts**

**Problem: fm_delivery_mail_sendmsgs Error Reported in the CM Log File**

**Cause**

BRM is trying to send a welcome email message, but the Email Data Manager (*dm_email*) is not running.

**Solution**

Start the Email Data Manager or disable the welcome email message.

- To start the Email Data Manager, see "Sending Email to Customers Automatically" in *BRM Managing Customers*.
- To disable the welcome message, see "Disabling the Welcome Message" in *BRM Managing Customers*.

**Getting Help with BRM Problems**

If you cannot resolve your problems with BRM, contact Oracle Technical Support.

**Before You Contact Technical Support**

Problems can often be fixed simply by shutting down BRM and restarting the computer that the BRM system runs on. To shut down and restart BRM, see "Starting and Stopping the BRM System".

If that doesn’t solve the problem, the first troubleshooting step is to look at the error log for the application or process that reported the problem. See "Using Error Logs to Troubleshoot BRM". Be sure to observe "General Checklist for Resolving Problems with BRM" before reporting the problem to Oracle Technical Support.

**Reporting Problems**

If "General Checklist for Resolving Problems with BRM" doesn’t help you to resolve the problem, write down the pertinent information:

- A clear and concise description of the problem, including when it began to occur.
- Relevant portions of the relevant log files.
- Relevant configuration files (*pin.conf* or *properties*).
- Recent changes in your system, even if you do not think they are relevant.
- List of all BRM components, ServicePaks, FeaturePaks, and patches installed on your system.

---

**Note:**

- You can collect BRM information by using the `pinrev` script. For information, see "Checking BRM Component Version Numbers".

- You can collect pipeline information by using the `piperev` script. For information, see "Checking Pipeline Component Version Numbers".

---

When you are ready, report the problem to Oracle Technical Support.
Reference Guide to BRM Error Codes

This document lists the Oracle Communications Billing and Revenue Management (BRM) error locations, classes, and codes.

For information about troubleshooting BRM, including examples of error messages that use these error codes, see "Resolving Problems in Your BRM System".

Interpreting BRM Error Codes

When a BRM process has a problem, its log file displays an error message that often includes:

- Error location; for example,
  
  location=<PIN_ERRLOC_FM:5>

- Error class; for example,
  
  class=<PIN_ERRCLASS_APPLICATION:4>

- Error code; for example,
  
  errno=<PIN_ERR_BAD_VALUE:46>

The tables below list the error locations, classes, and codes and give the meaning of each. Use this information to help find what caused the problem and what you can do to solve it.

BRM Error Locations

Table 18–1 lists the BRM error locations.

<table>
<thead>
<tr>
<th>Error location</th>
<th>No.</th>
<th>Source of the error</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIN_ERRLOC_PCM</td>
<td>1</td>
<td>General problem connecting to BRM. Common causes include illegal parameters.</td>
</tr>
<tr>
<td>PIN_ERRLOC_PCP</td>
<td>2</td>
<td>Internal error. The Portal Communications Protocol (PCP) library provides communication support between the modules of BRM. Common causes include network connection failures. This value indicates a system problem that requires immediate attention.</td>
</tr>
<tr>
<td>PIN_ERRLOC_CM</td>
<td>3</td>
<td>Connection Manager. Common causes include an unknown opcode or an input flist missing the required Portal object ID (POID) field.</td>
</tr>
</tbody>
</table>

Reference Guide to BRM Error Codes  18-1
Table 18–1  (Cont.) BRM Error Locations

<table>
<thead>
<tr>
<th>Error location</th>
<th>No.</th>
<th>Source of the error</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIN_ERRLOC_DM</td>
<td>4</td>
<td>Data Manager. Common causes include an input flist that does not meet the required specification or a problem communicating with the underlying data storage system.</td>
</tr>
<tr>
<td>PIN_ERRLOC_FM</td>
<td>5</td>
<td>Facilities Module. Common causes include an input flist that does not conform to the required specification.</td>
</tr>
<tr>
<td>PIN_ERRLOC_FLIST</td>
<td>6</td>
<td>An flist manipulation routine local to the application. Common causes include an illegal parameter and low system memory.</td>
</tr>
<tr>
<td>PIN_ERRLOC_POID</td>
<td>7</td>
<td>POID manipulation routine local to the application. Common causes include an illegal parameter and low system memory.</td>
</tr>
<tr>
<td>PIN_ERRLOC_APP</td>
<td>8</td>
<td>An error occurred within an application.</td>
</tr>
<tr>
<td>PIN_ERRLOC_QM</td>
<td>9</td>
<td>An error occurred within the Queue Manager (qmflist).</td>
</tr>
<tr>
<td>PIN_ERRLOC_PCMCPP</td>
<td>10</td>
<td>An error occurred within the PCM C++ wrapper.</td>
</tr>
<tr>
<td>PIN_ERRLOC_LDAP</td>
<td>11</td>
<td>An error occurred within the LDAP library.</td>
</tr>
<tr>
<td>PIN_ERRLOC_NMGR</td>
<td>12</td>
<td>An error occurred within Node Manager.</td>
</tr>
<tr>
<td>PIN_ERRLOC_INFMGR</td>
<td>13</td>
<td>An error occurred within System Manager.</td>
</tr>
<tr>
<td>PIN_ERRLOC_UTILS</td>
<td>14</td>
<td>An error occurred within a BRM utility.</td>
</tr>
<tr>
<td>PIN_ERRLOC_JS</td>
<td>15</td>
<td>An error occurred within the Java Server Framework.</td>
</tr>
<tr>
<td>PIN_ERRLOC_JSAPP</td>
<td>16</td>
<td>An error occurred within the Java Server Application or opcode handler.</td>
</tr>
<tr>
<td>PIN_ERRLOC_PDO</td>
<td>17</td>
<td>An error occurred within the BRM data objects.</td>
</tr>
<tr>
<td>PIN_ERRLOC_RTP</td>
<td>19</td>
<td>An error occurred within a real-time pipeline.</td>
</tr>
<tr>
<td>PIN_ERRLOC_ADTP</td>
<td>21</td>
<td>An error occurred within the ADT lib (iScript).</td>
</tr>
</tbody>
</table>

BRM Error Classes

Table 18–2 lists the BRM error classes.
Table 18–2 BRM Error Classes

<table>
<thead>
<tr>
<th>Error class</th>
<th>No.</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIN_ERRCLASS_SYSTEM_DETERMINATE</td>
<td>1</td>
<td>The error was caused by a system failure during the operation. Retrying the operation is unlikely to succeed, and the system failure should be investigated immediately. The error was detected before any data was committed to the database; no data has changed. After the error is fixed, retry the operation.</td>
</tr>
<tr>
<td>PIN_ERRCLASS_SYSTEM_INDETERMINATE</td>
<td>2</td>
<td>The error was caused by a system failure during the “commit” phase of an operation. The error might not be repeatable, and the system might not save specific information about the error. There is a small window during the commit where a network failure can leave the system unsure of whether the commit occurred. This means it is up to the application to determine whether system data has been changed. This class of error is extremely rare, but you must deal with it carefully to avoid corrupting the data in the database. The transactional model of BRM guarantees that either all the changes within the indeterminate transaction are committed, or none of them are committed and the system data is left unchanged. If you find that no changes were made, resolve the system failure and retry the operation.</td>
</tr>
<tr>
<td>PIN_ERRCLASS_SYSTEM_RE RETRYABLE</td>
<td>3</td>
<td>The error was probably caused by a transient condition. Try the operation again. Common causes include a temporary shortage of system resources (perhaps caused by too many connections to the CM) or a failure of a network connection that you can route around. The error was detected before any data was committed to the database; no data has changed.</td>
</tr>
<tr>
<td>PIN_ERRCLASS_APPLICATION</td>
<td>4</td>
<td>The error was caused by a custom application passing invalid data to BRM or by a system failure within the client application. The error was detected before the requested operation was performed, so no data in the database has been changed. After you fix the error, retry the operation.</td>
</tr>
</tbody>
</table>

BRM Error Codes

Table 18–3 lists the BRM error codes.

Table 18–3 BRM Error Codes

<table>
<thead>
<tr>
<th>Error code</th>
<th>No.</th>
<th>Cause and solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIN_ERR_NONE</td>
<td>0</td>
<td>No error. This error code indicates a general condition, not a specific error. Some BRM routines use this error code to indicate that the routine was successful.</td>
</tr>
<tr>
<td>PIN_ERR_NO_MEM</td>
<td>1</td>
<td>There was insufficient memory to complete the attempted operation. Check system memory. Also check the shmsize and bigsize values in the Data Manager configuration file (pin.conf). If you see this error code with a custom application, check for memory leaks. Solaris: Check that the shared memory for the system is at least as great as the shared memory set for BRM.</td>
</tr>
<tr>
<td>PIN_ERR_NO_MATCH</td>
<td>2</td>
<td>BRM couldn’t find the value it was looking for. From the Data Manager, this error indicates a bad search template from the qm_flist. From the SDK FM, this error means that the FM cannot find the object it was told to modify.</td>
</tr>
</tbody>
</table>
### Table 18–3 (Cont.) BRM Error Codes

<table>
<thead>
<tr>
<th>Error code</th>
<th>No.</th>
<th>Cause and solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIN_ERR_NOT_FOUND</td>
<td>3</td>
<td>BRM couldn’t find a value. This error code doesn’t always indicate an error. For example, some opcodes look for a value in the configuration file, but if the value is not there, a default value can be used.</td>
</tr>
<tr>
<td>PIN_ERR_BAD_ARG</td>
<td>4</td>
<td>A required field in an flist is incorrect. This is always a serious error because the system will not work until the argument is fixed. The problem is usually a programming or data entry error.</td>
</tr>
<tr>
<td>PIN_ERR_BAD_XDR</td>
<td>5</td>
<td>The application unexpectedly lost the connection to the BRM database. Usually, this error means that the connection to the network was lost. If the network is working correctly, the BRM server might have stopped working. Look for errors in the CM log file. Also check for CM or DM core files.</td>
</tr>
<tr>
<td>PIN_ERR_BAD_FIRST_READ</td>
<td>6</td>
<td>No longer used.</td>
</tr>
<tr>
<td>PIN_ERR_BAD_READ</td>
<td>7</td>
<td>BRM couldn’t read from the network or some other IO device, probably because the connection was cut off unexpectedly.</td>
</tr>
<tr>
<td>PIN_ERR_NO_SOCKET</td>
<td>8</td>
<td>BRM couldn’t create a socket. The machine or process might be overloaded and have reached a limit on socket/file descriptors. The networking part of the operating system might have failed. Try restarting the machine. If that doesn’t work, report the problem to the OS vendor.</td>
</tr>
<tr>
<td>PIN_ERR_BAD_TYPE</td>
<td>9</td>
<td>BRM encountered an erroneous field or object type. This error usually results from programming errors in custom applications and FMIs, but the error might also result from a mismatch between a field and its corresponding field type. If seen when <code>pcpxdr_fld_list</code> is called, it means a down-level version of the libraries saw incompatible wire protocols or a new field type.</td>
</tr>
<tr>
<td>PIN_ERR_DUPLICATE</td>
<td>10</td>
<td>BRM couldn’t create a storable object because the requested ID is already used by another object.</td>
</tr>
<tr>
<td>PIN_COMPARE_EQUAL</td>
<td>11</td>
<td>This code doesn’t indicate an error. The billing FM uses this code for internal operations.</td>
</tr>
<tr>
<td>PIN_COMPARE_NOT_EQUAL</td>
<td>12</td>
<td>This code doesn’t indicate an error. The billing FM uses this code for internal operations.</td>
</tr>
<tr>
<td>PIN_ERR_MISSING_ARG</td>
<td>13</td>
<td>A required argument is missing. If the log file doesn’t indicate the field, see the specification for the opcode.</td>
</tr>
<tr>
<td>PIN_ERR_BAD_POID_TYPE</td>
<td>14</td>
<td>BRM encountered an erroneous object type. This is similar to error 9, but is more specific to object type. For example, BRM was expecting an account object but encountered an event object.</td>
</tr>
<tr>
<td>PIN_ERR_BAD_CRYPT</td>
<td>15</td>
<td>Packet header failed encryption/decryption. Possible corruption of data.</td>
</tr>
<tr>
<td>PIN_ERR_BAD_WRITE</td>
<td>16</td>
<td>Error while attempting to send data on the IP socket.</td>
</tr>
<tr>
<td>PIN_ERR_DUP_SUBSTRUCT</td>
<td>17</td>
<td>Information not available.</td>
</tr>
<tr>
<td>Error code</td>
<td>No.</td>
<td>Cause and solution</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>-----</td>
<td>------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>PIN_ERR_BAD_SEARCH_ARG</td>
<td>18</td>
<td>A required field in a search template or flist is incorrect. This is always a serious error because the system will not work until the argument is fixed. The problem is usually a programming or data entry error.</td>
</tr>
<tr>
<td>PIN_ERR_BAD_SEARCH_ARG_RECID</td>
<td>19</td>
<td>Invalid automatic number identification (ANI) in the search operation.</td>
</tr>
<tr>
<td>PIN_ERR_DUP_SEARCH_ARG</td>
<td>20</td>
<td>There are duplicate entries in the search argument list.</td>
</tr>
<tr>
<td>PIN_ERR_NONEXISTANT_POID</td>
<td>21</td>
<td>BRM cannot find the storable object in the database.</td>
</tr>
<tr>
<td>PIN_ERR_POID_DB_IS_ZERO</td>
<td>22</td>
<td>The database number is specified as zero. Make sure the routine is passing a valid database number that matches the number in the configuration file.</td>
</tr>
<tr>
<td>PIN_ERR_UNKNOWN_POID</td>
<td>23</td>
<td>The POID does not contain valid information, or the format is incorrect.</td>
</tr>
<tr>
<td>PIN_ERR_NO_SOCKETS</td>
<td>24</td>
<td>BRM couldn’t create a socket. The machine or process might be overloaded and have reached a limit on socket/file descriptors. The networking part of the operating system might have failed. Try restarting the machine. If that doesn’t work, report the problem to the OS vendor.</td>
</tr>
<tr>
<td>PIN_ERR_DM_ADDRESS_LOOKUP_FAILED</td>
<td>25</td>
<td>The Connection Manager couldn’t find the Data Manager. The Bind (or DNS) service is pointing to the wrong TCP/IP address, or the network is having problems. Try pinging the DM to verify the network connection. Check the DM pointer specified in the configuration file for the CM.</td>
</tr>
<tr>
<td>PIN_ERR_DM_CONNECT_FAILED</td>
<td>26</td>
<td>BRM couldn’t connect to the Data Manager. The configuration file for the CM might be pointing to the wrong DM, or the DM might not be running.</td>
</tr>
<tr>
<td>PIN_ERR_NAP_CONNECT_FAILED</td>
<td>27</td>
<td>An application couldn’t connect to the Connection Manager. The configuration file for the application might be pointing to the wrong CM. The CM might not be running, or there might not be any more available connections. Also check for network errors between the application and the CM. For example, no more TCP sockets might be available on the client or CM machine.</td>
</tr>
<tr>
<td>PIN_ERR_INVALID_RECORD_ID</td>
<td>28</td>
<td>The ID of the specified element in the array is not valid. The specified ID might be greater than the maximum record ID.</td>
</tr>
<tr>
<td>PIN_ERR_STALE_CONF</td>
<td>29</td>
<td>BRM found outdated values in pin.conf entries.</td>
</tr>
<tr>
<td>PIN_ERR_INVALID_CONF</td>
<td>30</td>
<td>Configuration data is missing or in an invalid format in the pin.conf files.</td>
</tr>
<tr>
<td>PIN_ERR_WRONG_DATABASE</td>
<td>31</td>
<td>The database number in the POID is not valid.</td>
</tr>
<tr>
<td>PIN_ERR_DUP_ARG</td>
<td>32</td>
<td>The flist has duplicate fields or elements.</td>
</tr>
<tr>
<td>PIN_ERR_BAD_SET</td>
<td>33</td>
<td>BRM couldn’t assign a storable object ID.</td>
</tr>
<tr>
<td>PIN_ERR_BAD_CREATE</td>
<td>34</td>
<td>A routine couldn’t create an object.</td>
</tr>
<tr>
<td>PIN_ERR_BAD_FIELD_NAME</td>
<td>35</td>
<td>Mapping error from field home to type.</td>
</tr>
<tr>
<td>PIN_ERR_BAD_OPCODE</td>
<td>36</td>
<td>Undefined opcode used.</td>
</tr>
</tbody>
</table>
### Table 18–3 (Cont.) BRM Error Codes

<table>
<thead>
<tr>
<th>Error code</th>
<th>No.</th>
<th>Cause and solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIN_ERR_TRANS_ALREADY_OPEN</td>
<td>37</td>
<td>An application attempted to open a transaction when one was already open on the same storable object. This error usually results from a programming error in a custom application or FM. The error sometimes appears in a series of cascading errors; look for a predecessor error.</td>
</tr>
<tr>
<td>PIN_ERR_TRANS_NOT_OPEN</td>
<td>38</td>
<td>An application attempted to commit or abort a transaction, but none had been opened. This error usually results from a programming error in a custom application or FM. The error sometimes shows up in a series of cascading errors; look for a predecessor error.</td>
</tr>
<tr>
<td>PIN_ERR_NULL_PTR</td>
<td>39</td>
<td>A routine couldn’t get a value because it was set to “null”. This error usually results from a programming error in a custom application or FM.</td>
</tr>
<tr>
<td>PIN_ERR_BAD_FREE</td>
<td>40</td>
<td>A routine tried, but failed, to free memory that was no longer needed. This error usually results from a programming error in a custom application or FM. This problem might cause memory leaks.</td>
</tr>
<tr>
<td>PIN_ERR_FILE_IO</td>
<td>41</td>
<td>Error while attempting to do an IO operation on a file.</td>
</tr>
<tr>
<td>PIN_ERR_NONEXISTANT_ELEMENT</td>
<td>42</td>
<td>The array in the specified storable object doesn’t have the specified element.</td>
</tr>
<tr>
<td>PIN_ERR_STORAGE</td>
<td>43</td>
<td>The database returned an error. Check the Data Manager log file for database-specific error messages. Also check the database server error logs.</td>
</tr>
<tr>
<td>PIN_ERR_TRANS_TOO_MANY_POIDS</td>
<td>44</td>
<td>BRM attempted transactions to too many Data Managers.</td>
</tr>
<tr>
<td>PIN_ERR_TRANS_LOST</td>
<td>45</td>
<td>The transaction was lost. The Data Manager failed during a transaction.</td>
</tr>
<tr>
<td>PIN_ERR_BAD_VALUE</td>
<td>46</td>
<td>BRM couldn’t interpret data from the database. The data isn’t valid in the current context, and BRM cannot resolve the conflict.</td>
</tr>
<tr>
<td>PIN_ERR_PARTIAL</td>
<td>47</td>
<td>When sending a batch of transactions to the credit card Data Managers, some of the credit cards were processed, but others were not.</td>
</tr>
<tr>
<td>PIN_ERR_NOT_YET_DONE</td>
<td>48</td>
<td>BRM has not yet completed an operation (such as an opcode or transaction). This is typically an internal debugging error code that isn’t displayed.</td>
</tr>
<tr>
<td>PIN_ERR_STREAM_IO</td>
<td>49</td>
<td>The application encountered an error while sending data to or from the BRM database. Usually, this error means that the connection to the network was lost. If the network is working correctly, the server might have stopped working. Look for errors in the CM log file. Also check for CM or DM core files.</td>
</tr>
<tr>
<td>PIN_ERR_STREAM_EOF</td>
<td>50</td>
<td>The application unexpectedly lost the connection to the BRM database. Usually, this error means that the connection to the network was lost. If the network is working correctly, the server might have stopped working. Look for errors in the CM log file. Also check for CM or DM core files.</td>
</tr>
<tr>
<td>Error code</td>
<td>No.</td>
<td>Cause and solution</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>PIN_ERR_OP_NOT_OUTSTANDING</td>
<td>51</td>
<td>No operation is in progress under this context.</td>
</tr>
<tr>
<td>PIN_ERR_OP_ALREADY_BUSY</td>
<td>52</td>
<td>There is an operation already in progress within this context.</td>
</tr>
<tr>
<td>PIN_ERR_OP_ALREADY_DONE</td>
<td>53</td>
<td>Certain operations can be done only once. This error indicates that an operation of this type is being attempted a second time.</td>
</tr>
<tr>
<td>PIN_ERR_NO_DATA_FIELDS</td>
<td>54</td>
<td>The Data Manager received an flist with no data fields.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Check the input flist.</td>
</tr>
<tr>
<td>PIN_ERR_PROHIBITED_ARG</td>
<td>55</td>
<td>BRM couldn’t create an object because one or more values were invalid or fields do not allow updating.</td>
</tr>
<tr>
<td>PIN_ERR_BAD_LOGIN_RESULT</td>
<td>56</td>
<td>The application couldn’t connect to the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Check the login name and password.</td>
</tr>
<tr>
<td>PIN_ERR_CM_ADDRESS_LOOKUP_FAILED</td>
<td>57</td>
<td>The application couldn’t find the computer running the CM. The Bind (or DNS) service is pointing to the wrong TCP/IP address, or the network is having problems.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Try pinging the CM to verify the network connection. Check the CM host name specified in the configuration file for the application.</td>
</tr>
<tr>
<td>PIN_ERR_BAD_LOGIN_REDIRECT_INFO</td>
<td>58</td>
<td>The redirection information received from the CMMP is incorrect.</td>
</tr>
<tr>
<td>PIN_ERR_TOO_MANY_LOGIN_REDIRECTS</td>
<td>59</td>
<td>Too many connect login redirects from the CMMP.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This error might have resulted from a loop in the configuration. Check the configuration files for the application, CM, and DM.</td>
</tr>
<tr>
<td>PIN_ERR_STEP_SEARCH</td>
<td>60</td>
<td>The step search operation didn’t find an expected STEP_NEXT/STEP_END.</td>
</tr>
<tr>
<td>PIN_ERR_STORAGE_DISCONNECT</td>
<td>61</td>
<td>BRM lost the connection with the database during the middle of a transaction and couldn’t re-establish the connection.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See “Configuring Oracle Databases” in BRM Installation Guide.</td>
</tr>
<tr>
<td>PIN_ERR_NOT_GROUP_ROOT</td>
<td>62</td>
<td>Not the root of a group.</td>
</tr>
<tr>
<td>PIN_ERR_BAD_LOCKING</td>
<td>63</td>
<td>Error while attempting to lock/unlock a heap storage.</td>
</tr>
<tr>
<td>PIN_ERR_AUTHORIZATION_FAIL</td>
<td>64</td>
<td>No information available.</td>
</tr>
<tr>
<td>PIN_ERR_NOT_WRITABLE</td>
<td>65</td>
<td>Tried to write a field that cannot be modified.</td>
</tr>
<tr>
<td>PIN_ERR_UNKNOWN_EXCEPTION</td>
<td>66</td>
<td>Unknown C++ exception.</td>
</tr>
<tr>
<td>PIN_ERR_START_FAILED</td>
<td>67</td>
<td>BRM couldn’t start the process.</td>
</tr>
<tr>
<td>PIN_ERR_STOP_FAILED</td>
<td>68</td>
<td>BRM couldn’t stop the process.</td>
</tr>
<tr>
<td>PIN_ERR_INVALID_QUEUE</td>
<td>69</td>
<td>No information available.</td>
</tr>
<tr>
<td>PIN_ERR_TOO_BIG</td>
<td>70</td>
<td>No information available.</td>
</tr>
<tr>
<td>PIN_ERR_BAD_LOCALE</td>
<td>71</td>
<td>BRM doesn’t understand the locale. Check the locale of the computer running the client application.</td>
</tr>
<tr>
<td>PIN_ERR_CONV_MULTIBYTE</td>
<td>72</td>
<td>A client application had a problem converting data from UTF8 format, as it is stored in the BRM database, to multibyte format. Either the client has the wrong locale or the data is corrupted.</td>
</tr>
<tr>
<td>PIN_ERR_CONV_UNICODE</td>
<td>73</td>
<td>A client application had a problem converting data from UTF8 format, as it is stored in the BRM database, into Unicode format. Either the client has the wrong locale or the data are corrupted.</td>
</tr>
<tr>
<td>Error code</td>
<td>No.</td>
<td>Cause and solution</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----</td>
<td>-----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>PIN_ERR_BAD_MBCS</td>
<td>74</td>
<td>The input flist includes a string that is not in valid multibyte format.</td>
</tr>
<tr>
<td>PIN_ERR_BAD_UTF8</td>
<td>75</td>
<td>The input flist includes a string that is not in valid Unicode format.</td>
</tr>
<tr>
<td>PIN_ERR_CANON_CONV</td>
<td>76</td>
<td>No information available.</td>
</tr>
<tr>
<td>PIN_ERR_UNSUPPORTED_LOCALE</td>
<td>77</td>
<td>BRM doesn’t support canonicalization for the locale of the client application.</td>
</tr>
<tr>
<td>PIN_ERR_CURRENCY_MISMATCH</td>
<td>78</td>
<td>A subordinate bill unit or sponsored account has a different account currency than the parent or sponsor account.</td>
</tr>
<tr>
<td>PIN_ERR_DEADLOCK</td>
<td>79</td>
<td>Two or more database sessions attempted to access the same database resource. Each session waits for another session to release locks on the resource. The database detects the deadlock and aborts one of the session’s operations. If you receive this error, retry the transaction or operation.</td>
</tr>
<tr>
<td>PIN_ERR_BACKDATE_NOT_ALLOWED</td>
<td>80</td>
<td>BRM cannot backdate the adjustment, write-off, or other transaction because the G/L report has already been posted.</td>
</tr>
<tr>
<td>PIN_ERR_CREDIT_LIMIT_EXCEEDED</td>
<td>81</td>
<td>No information available.</td>
</tr>
<tr>
<td>PIN_ERR_IS_NULL</td>
<td>82</td>
<td>The value is <strong>Null</strong> (not set).</td>
</tr>
<tr>
<td>PIN_ERR_DETAILED_ERR</td>
<td>83</td>
<td>A detailed error message uses an enhanced buffer (<strong>errbuf</strong>).</td>
</tr>
<tr>
<td>PIN_ERR_PERMISSION_DENIED</td>
<td>84</td>
<td>The attempted operation is not allowed; data is not viewable.</td>
</tr>
<tr>
<td>PIN_ERR_PDO_INTERNAL</td>
<td>85</td>
<td>An internal error occurred in the BRM data objects.</td>
</tr>
<tr>
<td>PIN_ERR_IPT_DNIS</td>
<td>86</td>
<td>The Dialed Number Identification Service (DNIS) is not authorized.</td>
</tr>
<tr>
<td>PIN_ERR_DB_MISMATCH</td>
<td>87</td>
<td>The database numbers do not match.</td>
</tr>
<tr>
<td>PIN_ERR_NO_CREDIT_BALANCE</td>
<td>88</td>
<td>No credit balance is available.</td>
</tr>
<tr>
<td>PIN_ERR_NOTHING_TO_BILL</td>
<td>89</td>
<td>There are no new items to bill.</td>
</tr>
<tr>
<td>PIN_ERR_MASTER_DOWN</td>
<td>90</td>
<td>The main BRM system is down.</td>
</tr>
<tr>
<td>PIN_ERR_OPCODE_HNDLR_INIT_FAI</td>
<td>91</td>
<td>The opcode handler initialization at JS failed.</td>
</tr>
<tr>
<td>PIN_ERR_STMT_CACHE</td>
<td>92</td>
<td>There is a problem with the statement cache.</td>
</tr>
<tr>
<td>PIN_ERR_CACHE_SIZE_ZERO</td>
<td>93</td>
<td>Tried to init <strong>cm_cache</strong> with zero size.</td>
</tr>
<tr>
<td>PIN_ERR_INVALID_OBJECT</td>
<td>94</td>
<td>The POID is invalid. This error occurs when an object is accessed whose database_number field of the POID is NULL.</td>
</tr>
<tr>
<td>PIN_ERR_VALIDATE_ADJUSTMENT</td>
<td>95</td>
<td>The validate adjustment policy opcode fails.</td>
</tr>
<tr>
<td>PIN_ERR_SYSTEM_ERROR</td>
<td>96</td>
<td>A generic system error during running application.</td>
</tr>
<tr>
<td>PIN_ERR_BILLING_ERROR</td>
<td>97</td>
<td>An error occurred during the billing run.</td>
</tr>
<tr>
<td>PIN_ERR_AUDIT_COMMIT_FAILED</td>
<td>98</td>
<td>Commit for the audit tables failed.</td>
</tr>
<tr>
<td>PIN_ERR_NOT_SUPPORTED</td>
<td>99</td>
<td>The operation is not supported.</td>
</tr>
<tr>
<td>PIN_ERR_INVALID_SER_FORMAT</td>
<td>100</td>
<td>The serialized format received from the database is incorrect.</td>
</tr>
<tr>
<td>Error code</td>
<td>No.</td>
<td>Cause and solution</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>-----</td>
<td>--------------------------------------------------------------</td>
</tr>
<tr>
<td>PIN_ERR_READ_ONLY_TXN</td>
<td>101</td>
<td>Cannot insert or update in a read-only transaction.</td>
</tr>
<tr>
<td>PIN_ERR_VALIDATION_FAILED</td>
<td>102</td>
<td>Deals or plans validation failed.</td>
</tr>
<tr>
<td>PIN_ERR_PROC_BIND</td>
<td>103</td>
<td>The binding for the Procedure arguments failed.</td>
</tr>
<tr>
<td>PIN_ERR_PROC_EXEC</td>
<td>104</td>
<td>The procedure returned an Application Specific error.</td>
</tr>
<tr>
<td>PIN_ERR_STORAGE_FAIlOVER</td>
<td>105</td>
<td>An Oracle RAC failover message.</td>
</tr>
<tr>
<td>PIN_ERR_RETRYABLE</td>
<td>106</td>
<td>A failover error occurred and is retryable if needed.</td>
</tr>
<tr>
<td>PIN_ERR_NOT_PRIMARY</td>
<td>107</td>
<td>Not the primary instance.</td>
</tr>
<tr>
<td>PIN_ERR_TIMEOUT</td>
<td>108</td>
<td>Request timeout.</td>
</tr>
<tr>
<td>PIN_ERR_CONNECTION_LOST</td>
<td>109</td>
<td>The connection has been lost.</td>
</tr>
<tr>
<td>PIN_ERR_BAD_ENC_SCHEME</td>
<td>116</td>
<td>The MD5 encryption scheme is configured; however, the AES encryption library is being used.</td>
</tr>
</tbody>
</table>
This document describes Oracle Communications Billing and Revenue Management (BRM) Pipeline Manager error messages.

Note:
- Many error descriptions include the string value. This string is replaced with the appropriate value by the module logging the error.
- Modules that aren’t listed in this document do not log module-specific error messages. However, modules can return pipeline framework error messages. For information on framework error messages, see "Pipeline Framework Error Messages".

Pipeline Framework Error Messages

Table 19–1 lists the Pipeline Framework error messages.

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_A_CUSTOMER_NOT_FOUND</td>
<td>A-Customer not found (value).</td>
</tr>
<tr>
<td>ERR_ACTIVATED_DATE_INVALID</td>
<td>Contract value has an invalid activation date.</td>
</tr>
<tr>
<td>ERR_ADD_DATABLOCK</td>
<td>Cannot add Datablock 'value' to EDR-C.</td>
</tr>
<tr>
<td>ERR_ALIAS_IS_IN_WRONG_BLOCK</td>
<td>The alias value is in the wrong block.</td>
</tr>
<tr>
<td>ERR_BAD_SCHEMA_SOURCE</td>
<td>Bad schema source: value.</td>
</tr>
<tr>
<td>ERR_BLOCK_DESC_NOT_FOUND</td>
<td>Block description not found (value).</td>
</tr>
<tr>
<td>ERR_BLOCKID_UNKNOWN</td>
<td>Can’t find an index for id value.</td>
</tr>
<tr>
<td>ERR_BUILD_DESC_TREE</td>
<td>Error while opening/reading description tree (value).</td>
</tr>
<tr>
<td>ERRCALENDAR_PLUGIN_INV</td>
<td>Calendar data module invalid.</td>
</tr>
<tr>
<td>ERR_CALLTYPE_INVALID</td>
<td>Invalid call type: value: value.</td>
</tr>
<tr>
<td>ERR_CALLTYPE_NOT_FOUND</td>
<td>No call type found for EDR (value).</td>
</tr>
<tr>
<td>ERR_CAN_NOT_GET_FACTORY</td>
<td>Cannot get factory 'value'.</td>
</tr>
<tr>
<td>ERRCANCEL_FAILED_INPUT</td>
<td>Cancel failed in input-controller.</td>
</tr>
<tr>
<td>ERRCANCEL_TRANSACTION</td>
<td>Module 'value' failed to cancel transaction 'value'.</td>
</tr>
</tbody>
</table>
Table 19–1 (Cont.) Pipeline Framework Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_CANNOT_DECRYPT_PASSWORD</td>
<td>Cannot decrypt password 'value'; user 'value'.</td>
</tr>
<tr>
<td>ERR_CANNOT_FIND_EVENT_HANDLER_PROC</td>
<td>Cannot locate the event handler daemon!</td>
</tr>
<tr>
<td>ERR_CANNOT_FORK</td>
<td>Cannot create child process.</td>
</tr>
<tr>
<td>ERR_CANNOT_GET_DMT_DMCONNECTION</td>
<td>value: Cannot get a DMT::DMConnection</td>
</tr>
<tr>
<td>ERR_CANNOT_INIT_DB_VERSION</td>
<td>Cannot initialize database version, error 'value'.</td>
</tr>
<tr>
<td>ERR_CANNOT_INIT_INPUT_STREAM</td>
<td>Cannot initialize the input stream object.</td>
</tr>
<tr>
<td>ERR_CANNOT_INIT_INPUT_STREAM_INTERFACE</td>
<td>Cannot initialize the input stream interface object.</td>
</tr>
<tr>
<td>ERR_CANNOT_INIT_OUTPUT_COLLECTION</td>
<td>Cannot initialize the output collection.</td>
</tr>
<tr>
<td>ERR_CANNOT_INIT_OUTPUT_MODULE</td>
<td>Cannot initialize the output module.</td>
</tr>
<tr>
<td>ERR_CANNOT_INIT_OUTPUT_STREAM</td>
<td>Cannot initialize the output stream object.</td>
</tr>
<tr>
<td>ERR_CANNOT_INIT_OUTPUT_STREAM_INTERFACE</td>
<td>Cannot initialize the output stream interface object.</td>
</tr>
<tr>
<td>ERR_CANNOT_JOIN_EVENT_HANDLER_PROC</td>
<td>Cannot connect to event handler process: value</td>
</tr>
<tr>
<td>ERR_CANNOT_OPEN_DATABASE</td>
<td>Cannot open database 'value'; user 'value'; password 'value'; server message 'value'.</td>
</tr>
<tr>
<td>ERR_CANNOT_RENAME_OUTPUT_FILE</td>
<td>Cannot rename temporary output file 'value'.</td>
</tr>
<tr>
<td>ERR_CHARGE_ITEM_INVALID</td>
<td>ChargeItem value invalid.</td>
</tr>
<tr>
<td>ERR_CHARGED_ZONE_NOT_FOUND</td>
<td>The EDR charged zone cannot be found for 'value'.</td>
</tr>
<tr>
<td>ERR_CIBER_RET</td>
<td>CIBER return: retReason value, retCode value, fieldID value, ruleID value.</td>
</tr>
<tr>
<td>ERR_CLIMAP_FILENAME_EMPTY</td>
<td>Empty cli mapping file name specified.</td>
</tr>
<tr>
<td>ERR_COMMIT_TRANSACTION</td>
<td>Module 'value' failed to commit transaction 'value'.</td>
</tr>
<tr>
<td>ERR_CON_ATTACHED_FIELD</td>
<td>The attached field information has not the right format. Must be BLOCKNAME.FIELDNAME, is value).</td>
</tr>
<tr>
<td>ERR_CONTROLLER_CONFIGURATION</td>
<td>Pipeline controller configuration has error in 'value'.</td>
</tr>
<tr>
<td>ERR_CONTROLLER_HAS_WRONG_TYPE</td>
<td>Pipeline controller has wrong type in 'value'.</td>
</tr>
<tr>
<td>ERR_CONVERSION_BAS_DATE</td>
<td>value could not be converted to BAS_Date.</td>
</tr>
<tr>
<td>ERR_CONVERSION_FAILED</td>
<td>EDR conversion failed (value).</td>
</tr>
<tr>
<td>ERR_CONVERSION_INT</td>
<td>value is no valid integer value.</td>
</tr>
<tr>
<td>ERR_CORBA_EXCEPTION</td>
<td>CORBA exception: value.</td>
</tr>
<tr>
<td>ERR_CREATE_ALIAS_MAP_INDEX</td>
<td>No AliasMap entry found for Reference 'value' and logical Name 'value'.</td>
</tr>
<tr>
<td>ERR_CREATE_EDR_INDEX</td>
<td>EDR index creation failed: value (name='value', key='value' and reference='value').</td>
</tr>
<tr>
<td>ERR_CREATE_INDEX</td>
<td>EDR index creation failed: value</td>
</tr>
<tr>
<td>ERR_CREATE_INPUT_PARSER</td>
<td>Failed to create input parser: value</td>
</tr>
<tr>
<td>ERR_CREATE_INSTANCE</td>
<td>Error creating instance of value</td>
</tr>
<tr>
<td>ERR_CREATE_OBJECT_FAILED</td>
<td>Cannot create object 'value' (invalid (NULL) pointer).</td>
</tr>
<tr>
<td>ERR_CREATE_OUTPUT_PARSER</td>
<td>Failed to create output parser: value</td>
</tr>
</tbody>
</table>
Table 19–1 (Cont.) Pipeline Framework Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_CREATE_SCRIPT</td>
<td>Error loading script value. value.</td>
</tr>
<tr>
<td>ERR_CREATE_THREAD_FAILED</td>
<td>Cannot create thread instance for 'value'; invalid thread body.</td>
</tr>
<tr>
<td>ERR_CUG_FILENAME_EMPTY</td>
<td>Empty closed user group file name specified.</td>
</tr>
<tr>
<td>ERR_CUST_A_IDENTIFICATION_UNKNOWN</td>
<td>Customer identification technique for used service ('value') not found</td>
</tr>
<tr>
<td>ERR_CUST_A_VALUE_NOTFOUND</td>
<td>Missing value for field 'value' of Customer A</td>
</tr>
<tr>
<td>ERR_CUST_FILE_VERSION</td>
<td>Illegal customer file version value.</td>
</tr>
<tr>
<td>ERR_CUST_FILENAME_EMPTY</td>
<td>Empty customer file name specified.</td>
</tr>
<tr>
<td>ERR_CUSTOMER_DATA_INVALID</td>
<td>Invalid customer data.</td>
</tr>
<tr>
<td>ERR_DAT_PREFDESC_INS_TREE_DB</td>
<td>Can’t insert line value from table value into prefix description table.</td>
</tr>
<tr>
<td>ERR_DAT_PREFDESC_INS_TREE_FILE</td>
<td>Can’t insert line value from file value into prefix description table.</td>
</tr>
<tr>
<td>ERR_DATA_INVALID</td>
<td>The data in field value in invalid.</td>
</tr>
<tr>
<td>ERR_DATA_PLUGIN_INVALID</td>
<td>Module 'value' is invalid.</td>
</tr>
<tr>
<td>ERR_DATA_PLUGIN_NOT_FOUND</td>
<td>Module 'value' cannot be found in the DataPool.</td>
</tr>
<tr>
<td>ERR_DATABASE</td>
<td>Database error 'value'.</td>
</tr>
<tr>
<td>ERR_DB_COMMIT_TRANSACTION</td>
<td>Cannot commit database transaction 'value'.</td>
</tr>
<tr>
<td>ERR_DB_CONNECTION_MODULE</td>
<td>Database connection module is invalid.</td>
</tr>
<tr>
<td>ERR_DB_CONNECTION_NOT_VALID</td>
<td>Could not connect to database.</td>
</tr>
<tr>
<td>ERR_DB_NUMBER_OF_ROWS</td>
<td>Statement 'value' does not return exactly one row.</td>
</tr>
<tr>
<td>ERR_DB_START_TRANSACTION</td>
<td>Error starting database transaction: 'value'</td>
</tr>
<tr>
<td>ERR_DB_STATEMENT_EXECUTE</td>
<td>Cannot execute database statement 'value', message 'value'.</td>
</tr>
<tr>
<td>ERR_DB_VERSION_CHECK</td>
<td>Wrong database version. Please check module and database version.</td>
</tr>
<tr>
<td>ERR_DB_VERSIONS_NOT_FOUND</td>
<td>Database versions 'value' not found.</td>
</tr>
<tr>
<td>ERR_DD_NOT_READ</td>
<td>Cannot read the data dictionary.</td>
</tr>
<tr>
<td>ERR_DEF_IS_INCOMPLETE</td>
<td>The field definition 'value' is incomplete.</td>
</tr>
<tr>
<td>ERR_DEFAULT_BLOCK_NOT_FOUND</td>
<td>The specified default block name value doesn’t exist in the description.</td>
</tr>
<tr>
<td>ERR_DEFAULT_WITH_WRONG_ID</td>
<td>Output Stream value : The default block has a wrong id. Check your format description.</td>
</tr>
<tr>
<td>ERR_DELETE_FILE</td>
<td>Cannot delete file 'value'.</td>
</tr>
<tr>
<td>ERR_DELETE_OUTPUT_FILE</td>
<td>'value': Cannot delete output file 'value'.</td>
</tr>
<tr>
<td>ERR_DIR_EMPTY</td>
<td>Reading from empty directory 'value'.</td>
</tr>
<tr>
<td>ERR_DIR_NOT_ACCESSIBLE</td>
<td>Directory 'value' is not accessible.</td>
</tr>
<tr>
<td>ERR_DIR_NOT_WRITEABLE</td>
<td>Directory value is not writable.</td>
</tr>
</tbody>
</table>
## Table 19–1  (Cont.) Pipeline Framework Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_DLOPEN_FAILED</td>
<td>Cannot open shared library 'value'; value. Make sure the LD_LIBRARY_PATH_64 environment variable includes Pipeline_homeLib.</td>
</tr>
<tr>
<td>ERR_DLSYM_FAILED</td>
<td>Cannot get address of generator function 'value'; value.</td>
</tr>
<tr>
<td>ERR_DONE_PATH_NOT_FOUND</td>
<td>Entry for done path not found in registry.</td>
</tr>
<tr>
<td>ERR_DOUBLE_ALIAS_NAME</td>
<td>The reference to the alias value exist more than one times.</td>
</tr>
<tr>
<td>ERR_DOUBLE_SEQ_NUMBER</td>
<td>Double sequence number found (sequence number: 'value').</td>
</tr>
<tr>
<td>ERR_DOUBLE_TRANS_MODULE</td>
<td>Transaction module 'value' was attached more than once.</td>
</tr>
<tr>
<td>ERR_DOUBLE_TRANSACTION_ID</td>
<td>Double transaction id 'value' found.</td>
</tr>
<tr>
<td>ERR_DUPLICATE_IRULE_PARAMETER</td>
<td>Duplicate iRule parameter 'value' found in file 'value'.</td>
</tr>
<tr>
<td>ERR_DUPLICATE_NUMPORTDATA</td>
<td>Duplicate number portability data found for the CLI value and the Portation TimeStamp value</td>
</tr>
<tr>
<td>ERR_EDR_ALIAS_NOT_FOUND</td>
<td>The specified field alias value couldn't founded.</td>
</tr>
<tr>
<td>ERR_EDR_BLOCK_NOT_FOUND</td>
<td>The specified block alias value couldn't founded.</td>
</tr>
<tr>
<td>ERR_EDR_BUILD_RECORD_NOT_FILLED</td>
<td>'value' - EDR buildt record field not filled.</td>
</tr>
<tr>
<td>ERR_EDR_CREATE</td>
<td>Failed to create new EDR container.</td>
</tr>
<tr>
<td>ERR_EDR_FACTORY_NOT_FOUND</td>
<td>EDR-Factory 'value' not found.</td>
</tr>
<tr>
<td>ERR_EDRTRACE_STREAMLOG_CREATION_FAIL</td>
<td>Error in EDR trace stream log creation.</td>
</tr>
<tr>
<td>ERR_EMPTY_CHARGEPACKET_LIST</td>
<td>No charge-packets found in charge breakdown record.</td>
</tr>
<tr>
<td>ERR_ERROR_PATH_NOT_FOUND</td>
<td>Entry for error path not found in registry.</td>
</tr>
<tr>
<td>ERR_ERROR_RATE_ALREADY_DEFINED</td>
<td>Error rate for 'value' already specified.</td>
</tr>
<tr>
<td>ERR_ERROR_RATE_VALUE_NOT_SPECIFIED</td>
<td>No value specified for error 'value'.</td>
</tr>
<tr>
<td>ERR_EVAL_ENVIRONMENT</td>
<td>Cannot evaluate environment 'value'.</td>
</tr>
<tr>
<td>ERR_FAILURE_FSM</td>
<td>Failure in finite state machine: value.</td>
</tr>
<tr>
<td>ERR_FILE_CLOSE_OS</td>
<td>value: Cannot close file 'value'.</td>
</tr>
<tr>
<td>ERR_FILE_EOF</td>
<td>Tried to read past end of file 'value'.</td>
</tr>
<tr>
<td>ERR_FILE_EXIST</td>
<td>File 'value' exist.</td>
</tr>
<tr>
<td>ERR_FILE_MOVE_OS</td>
<td>value: Cannot move file 'value' to 'value'.</td>
</tr>
<tr>
<td>ERR_FILE_NOT_FOUND</td>
<td>File 'value' not found.</td>
</tr>
<tr>
<td>ERR_FILE_NOT_MOVED</td>
<td>File 'value' could not be moved to 'value'.</td>
</tr>
<tr>
<td>ERR_FILE_NOT_WRITABLE</td>
<td>File 'value' is not writable.</td>
</tr>
<tr>
<td>ERR_FILE_OPEN_OS</td>
<td>value: Cannot open file 'value'.</td>
</tr>
<tr>
<td>ERR_FILE_READ_ERR</td>
<td>Error reading from file 'value'.</td>
</tr>
<tr>
<td>ERR_FILE_READ_OS</td>
<td>value: Error reading from file 'value'.</td>
</tr>
<tr>
<td>ERR_FILE_REMOVE_OS</td>
<td>value: Cannot remove file 'value'.</td>
</tr>
<tr>
<td>ERR_FILE_WRITE_ERR</td>
<td>Error writing into file 'value'.</td>
</tr>
<tr>
<td>ERR_FILE_WRITE_OS</td>
<td>value: Error writing into file 'value'.</td>
</tr>
<tr>
<td>Error message</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>ERR_FILENAME_MISSING</td>
<td>File name not set for <code>value</code>.</td>
</tr>
<tr>
<td>ERR_FLIST_INPUT_ERROR</td>
<td>Error while processing FLIST message: <code>value</code>.</td>
</tr>
<tr>
<td>ERR_GAP_IN_SEQ_NUMBER</td>
<td>Gap in sequence number found (sequence number: <code>value</code>).</td>
</tr>
<tr>
<td>ERR_GETTING_DATADESC</td>
<td>Failed to get the data description.</td>
</tr>
<tr>
<td>ERR_GRAMMAR_SYMBOL_LOOKUP</td>
<td>Symbol lookup for <code>value</code> failed: <code>value</code>.</td>
</tr>
<tr>
<td>ERR_ILLEGAL_STREAM_NUM</td>
<td>Tried to use illegal stream number <code>value</code> for output.</td>
</tr>
<tr>
<td>ERR_IN_RECEIVED_MESSAGE</td>
<td>Message <code>value</code> was invalid.</td>
</tr>
<tr>
<td>ERR_IN_SECTION</td>
<td>Error in section <code>value</code>.</td>
</tr>
<tr>
<td>ERR_INCORRECT_FILLER_LENGTH</td>
<td>Invalid record Filler length, expected: <code>value</code>, received: <code>value</code>.</td>
</tr>
<tr>
<td>ERR_INCORRECT_FORMAT_OBJ</td>
<td>The format description object couldn’t be founded or is invalid.</td>
</tr>
<tr>
<td>ERR_INDEX_NOT_CREATED</td>
<td>Couldn’t create the index for alias <code>value</code>.</td>
</tr>
<tr>
<td>ERR_INDEX_NOT_FOUND</td>
<td>Container index not found.</td>
</tr>
<tr>
<td>ERR_INIT_EDR_ITERATOR</td>
<td>Failed to initialize EDR iterator for <code>value</code>.</td>
</tr>
<tr>
<td>ERR_INIT_SEG_TARIFF_LINK</td>
<td>Failure during initialization of tariff segment link table.</td>
</tr>
<tr>
<td>ERR_INIT_TSC_MAPTABLE</td>
<td>Failed to init map table: <code>value</code>.</td>
</tr>
<tr>
<td>ERR_INIT_XERCES</td>
<td>Error: Xerces-c Initialization. Exception message: <code>value</code>.</td>
</tr>
<tr>
<td>ERR_INPUT_DONE_FILE_NOT_MOVED_TO_ERR</td>
<td><code>value</code>: Cannot move done file <code>value</code> to error file <code>value</code>.</td>
</tr>
<tr>
<td>ERR_INPUT_DONE_FILE_NOT_MOVED_TO_INPUT</td>
<td><code>value</code>: Cannot move done file <code>value</code> to input file <code>value</code>.</td>
</tr>
<tr>
<td>ERR_INPUT_FILE_NOT_MOVED</td>
<td><code>value</code>: Cannot move input file <code>value</code> to temporary file <code>value</code>.</td>
</tr>
<tr>
<td>ERR_INPUT_MAPPING_FAILED</td>
<td>Input mapping <code>value</code> failed: <code>value</code>.</td>
</tr>
<tr>
<td>ERR_INPUT_PATH_NOT_FOUND</td>
<td>Entry for input path not found in registry.</td>
</tr>
<tr>
<td>ERR_INPUT_REQUEST_ROLLBACK</td>
<td>The input has requested a rollback (reason=<code>value</code>).</td>
</tr>
<tr>
<td>ERR_INPUT_TEMP_FILE_NOT_MOVED</td>
<td><code>value</code>: Cannot move temporary input file <code>value</code> to input file <code>value</code>.</td>
</tr>
<tr>
<td>ERR_INPUT_TEMP_FILE_NOT_MOVED_TO_DONE_ERR</td>
<td><code>value</code>: Cannot move temporary file <code>value</code> to done or err file <code>value</code>.</td>
</tr>
<tr>
<td>ERR_INSERT_HASH</td>
<td>Failure during insert in hash map.</td>
</tr>
<tr>
<td>ERR_INSERTING_CLI</td>
<td>Error loading cli <code>value</code> (probably duplicated).</td>
</tr>
<tr>
<td>ERR_INSUFFICIENT_MEMORY</td>
<td>Insufficient memory available.</td>
</tr>
<tr>
<td>ERR_INVALID_DATABASE_VALUE</td>
<td>Database value for field <code>value</code> is invalid.</td>
</tr>
<tr>
<td>ERR_INVALID_DATE</td>
<td>Can't build date <code>value</code> for cli <code>value</code>.</td>
</tr>
<tr>
<td>ERR_INVALID_DATETIME</td>
<td><code>value</code>. Cannot build datetime <code>value</code> for cli <code>value</code>.</td>
</tr>
<tr>
<td>ERR_INVALID_FCI_COLL_ENTRIES</td>
<td>Invalid number of FCI collection entries (<code>value</code>).</td>
</tr>
</tbody>
</table>
Table 19–1  (Cont.) Pipeline Framework Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_INVALID_FCI_COLL_ORDER</td>
<td>Invalid order of FCI collection entries (value). This error occurs when buffers are configured in multiple function pools. In this configuration, each buffer must have a unique name.</td>
</tr>
<tr>
<td>ERR_INVALID_FIRST_CALL_TIMESTAMP</td>
<td>Invalid first call timestamp: value, calculated: value.</td>
</tr>
<tr>
<td>ERR_INVALID_HA_ROLE</td>
<td>The peer instance has already assumed the value role.</td>
</tr>
<tr>
<td>ERR_INVALID_INPUT_RECORD</td>
<td>Check length, numeric values or date fields for their correctness. (record:value)</td>
</tr>
<tr>
<td>ERR_INVALID_LAST_CALL_TIMESTAMP</td>
<td>Invalid last call timestamp: value, calculated: value.</td>
</tr>
<tr>
<td>ERR_INVALID_LINE_LENGTH</td>
<td>The input line length for record number value is invalid.</td>
</tr>
<tr>
<td>ERR_INVALID_PATTERN</td>
<td>Directory pattern 'value' is invalid.</td>
</tr>
<tr>
<td>ERR_INVALID_PLUGINSTATE</td>
<td>Invalid internal module state in value.</td>
</tr>
<tr>
<td>ERR_INVALID_QUEUE_SIZE</td>
<td>Queue size &lt; 0.</td>
</tr>
<tr>
<td>ERR_INVALID_RECORD_LENGTH</td>
<td>Defined RecordLength (value) does not match length (value) of read line.</td>
</tr>
<tr>
<td>ERR_INVALID_RECORD_NUMBER</td>
<td>Invalid number of records: value, counted: value.</td>
</tr>
<tr>
<td>ERR_INVALID_REG_BASE_NAME</td>
<td>Registry base name of 'value' does not match 'value'.</td>
</tr>
<tr>
<td>ERR_INVALID_REG_ENTRIES</td>
<td>Invalid Registry Entries. value</td>
</tr>
<tr>
<td>ERR_INVALID_REG_VALUE</td>
<td>Invalid value 'value' for 'value'.</td>
</tr>
<tr>
<td>ERR_INVALID_REJECT_STREAM_NUMBER</td>
<td>Stream number is out of range.</td>
</tr>
<tr>
<td>ERR_INVALID_SEQ_NUM</td>
<td>Invalid sequence number 'value'.</td>
</tr>
<tr>
<td>ERR_INVALID_SEQ_VALUE</td>
<td>The configuration value for value is invalid (value).</td>
</tr>
<tr>
<td>ERR_INVALID_SOCIAL_NO</td>
<td>Invalid social number 'value'.</td>
</tr>
<tr>
<td>ERR_INVALID_STATE</td>
<td>Received EDR invalid in the current state.</td>
</tr>
<tr>
<td>ERR_INVALID_THREAD_STATE</td>
<td>Invalid thread state in 'value'; value; value.</td>
</tr>
<tr>
<td>ERR_INVALID_TOKEN_COUNT</td>
<td>Number of HA role mediator token should be one but found 'value'.</td>
</tr>
<tr>
<td>ERR_INVALID_TOKEN_DB_NO</td>
<td>Invalid HA role mediator token database number. Found 'value' and expected to be 'value'.</td>
</tr>
<tr>
<td>ERR_LAST_LOAD_RELOAD_FAILED</td>
<td>The last load/reload operation has failed.</td>
</tr>
<tr>
<td>ERR_LEN_IS_MISSING</td>
<td>The first item in field definition value must a number.</td>
</tr>
<tr>
<td>ERR_LINE_NOT_IDENTIFIED</td>
<td>The line couldn't identified: value</td>
</tr>
<tr>
<td>ERR_LINE_NOT_INSERTED_DOUBLE</td>
<td>Could not insert line into message DB (double key). Line value</td>
</tr>
<tr>
<td>ERR_LINE_NOT_INSERTED_INVALID</td>
<td>Could not insert line into message DB (invalid key). Line value</td>
</tr>
<tr>
<td>ERR_LINK_TABLE_INVALID</td>
<td>The link table value is invalid.</td>
</tr>
<tr>
<td>ERR_LOADING_ABORTED</td>
<td>Loading data aborted after value records.</td>
</tr>
<tr>
<td>ERR_LOADING_CUSTOMER_DATA</td>
<td>Loading customer data failed.</td>
</tr>
<tr>
<td>ERR_LOADING_DBTABLE</td>
<td>Error while loading database table value.</td>
</tr>
</tbody>
</table>
### Pipeline Framework Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_LOADING_TIMEMODEL</td>
<td>Loading time model failed 'value'.</td>
</tr>
<tr>
<td>ERR_MAPPING_TABLE_INVALID</td>
<td>The mapping table is invalid.</td>
</tr>
<tr>
<td>ERR_MBI_INPUT_ERROR</td>
<td>Error while processing MBI message: value</td>
</tr>
<tr>
<td>ERR_MEM_MON_MEMORY_LIMIT</td>
<td>Reached specified memory usage limit. Usage: value KB, available: value KB</td>
</tr>
<tr>
<td>ERR_MEM_MON_PROCESS_LIMIT</td>
<td>Reached process size limit. Size: value KB, limit: value KB</td>
</tr>
<tr>
<td>ERR_MISSING_ARGUMENT</td>
<td>Argument 'value' not in message 'value'.</td>
</tr>
<tr>
<td>ERR_MISSING_LOG_FILE_NAME</td>
<td>Log output file name is missing.</td>
</tr>
<tr>
<td>ERR_MISSING_MESSAGE_FILE_NAME</td>
<td>Message file name is missing.</td>
</tr>
<tr>
<td>ERR_MISSING_REFERENCE_FIELD</td>
<td>Find some container references without a field reference in block 'value'.</td>
</tr>
<tr>
<td>ERR_MISSING_REFERENCE_NAME</td>
<td>Missing reference name in block description.</td>
</tr>
<tr>
<td>ERR_MISSING_VALUES_FOR_FIELD</td>
<td>Find a reference field entry without id's in block 'value'.</td>
</tr>
<tr>
<td>ERR_MODULE_NOT_EXIST</td>
<td>The module 'value' which was configured as an event originator does not exist.</td>
</tr>
<tr>
<td>ERR_MULTIPLE_RESTART_FILES</td>
<td>Found more than one restart file in directory 'value'.</td>
</tr>
<tr>
<td>ERR_NO_CLI</td>
<td>No cli in input record.</td>
</tr>
<tr>
<td>ERR_NO_CUSTOMER</td>
<td>No customer data for cli value in input record.</td>
</tr>
<tr>
<td>ERR_NO_CUSTOMER_DATA</td>
<td>No customer data present.</td>
</tr>
<tr>
<td>ERR_NO_CUSTOMER_PLUGIN</td>
<td>No customer plug-in present.</td>
</tr>
<tr>
<td>ERR_NO_DATABASE_PLUGIN</td>
<td>No database plug-in present.</td>
</tr>
<tr>
<td>ERR_NO_DEFAULT_OUTPUT_DEVICE</td>
<td>No default output device.</td>
</tr>
<tr>
<td>ERR_NO_DEFAULT_SENTENCE</td>
<td>There is no default sentence defined in the format description.</td>
</tr>
<tr>
<td>ERR_NO_DIR</td>
<td>Directory 'value' not accessible.</td>
</tr>
<tr>
<td>ERR_NO_EDRFACtorY</td>
<td>Can't get the factory to create EDRs in value.</td>
</tr>
<tr>
<td>ERR_NO_EVENTHANDLER_FOUND</td>
<td>Event handler not found in module 'value'.</td>
</tr>
<tr>
<td>ERR_NO_INDEX</td>
<td>Index 'value' not found.</td>
</tr>
<tr>
<td>ERR_NO_MESSAGE_FILE</td>
<td>There are no message file found. Path: value</td>
</tr>
<tr>
<td>ERR_NO_ORIGINAL_RECORD</td>
<td>Missing the original block.</td>
</tr>
<tr>
<td>ERR_NO_PATH_NAME</td>
<td>No path name given.</td>
</tr>
<tr>
<td>ERR_NO_REQUEST</td>
<td>Request value returned with no value.</td>
</tr>
<tr>
<td>ERR_NO_SEQ_VALUE</td>
<td>Sequence field &quot;value&quot; in sequence control file has no value.</td>
</tr>
<tr>
<td>ERR_NO_SPLITTING_PERFORMED</td>
<td>No splitting performed (spec-sys = value).</td>
</tr>
<tr>
<td>ERR_NO_SUBSCRIBER</td>
<td>No subscriber data for cli value in input record.</td>
</tr>
<tr>
<td>ERR_NOSP_ID_NOT_FOUND</td>
<td>NOSP-Id not found for Frm=value and AreaCode=value.</td>
</tr>
<tr>
<td>ERR_NOT_USABLE</td>
<td>The object 'value' is not usable.</td>
</tr>
</tbody>
</table>
### Table 19–1 (Cont.) Pipeline Framework Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_NOT_USABLE_REASON</td>
<td>Module is not usable: value.</td>
</tr>
<tr>
<td>ERR_NUMBER_OF_FIELDS_IN_RECORD</td>
<td>Found 'value' instead of 'value' fields in record 'value'.</td>
</tr>
<tr>
<td>ERR_OBJ_ALREADY_REGISTERED</td>
<td>'value' is already registered as 'value'.</td>
</tr>
<tr>
<td>ERR_OBJ_NOT_FOUND</td>
<td>The object 'value' could not be found.</td>
</tr>
<tr>
<td>ERR_OBJ_NOT_INITIALIZED</td>
<td>The object 'value' is not initialized.</td>
</tr>
<tr>
<td>ERR_OBJ_NOT_REGISTERABLE</td>
<td>The object 'value' could not be registered.</td>
</tr>
<tr>
<td>ERR_OBJ_NOT_REGISTERED</td>
<td>The object 'value' is not registered.</td>
</tr>
<tr>
<td>ERR_OFF_MIN_GREATER_MAX</td>
<td>The min offset is greater than the max offset.</td>
</tr>
<tr>
<td>ERR_ONLY_ONE_EXTERNAL_DATAFIELD</td>
<td>There can be only one external data field for 'value'.</td>
</tr>
<tr>
<td>ERR_OPEN_DIRFailed</td>
<td>Cannot open directory 'value'; error message 'value'.</td>
</tr>
<tr>
<td>ERR_OPEN_FILEFailed</td>
<td>Cannot open file 'value'.</td>
</tr>
<tr>
<td>ERR_OPEN_LOG_FILE</td>
<td>Cannot open log file 'value'.</td>
</tr>
<tr>
<td>ERR_OPEN_MESSAGE_FILE</td>
<td>Message file 'value' could not open.</td>
</tr>
<tr>
<td>ERR_OPEN_SOCIAL_FILE</td>
<td>Cannot open social number file 'value'.</td>
</tr>
<tr>
<td>ERR_OUTPUT_ALREADY_OPEN</td>
<td>Output stream already opened.</td>
</tr>
<tr>
<td>ERR_OUTPUT_MAPPING_FAILED</td>
<td>Output mapping failed: value.</td>
</tr>
<tr>
<td>ERR_OUTPUT_NOT_OPEN</td>
<td>Cannot close output stream (not open).</td>
</tr>
<tr>
<td>ERR_OUTPUT_PATH_NOT_FOUND</td>
<td>The output path does not exists or is not accessible.</td>
</tr>
<tr>
<td>ERR_OUTPUT_TEMP_FILE_NOT_MOVED_TO_OUTPUT</td>
<td>'value': Cannot move temporary file 'value' to output file 'value'.</td>
</tr>
<tr>
<td>ERR_PARAMETER_FILE_INVALID</td>
<td>The iRule parameter file 'value' has an invalid format.</td>
</tr>
<tr>
<td>ERR_PARSEDescriptions</td>
<td>Failed to parse EDR description: value</td>
</tr>
<tr>
<td>ERR_PARSE_ERROR_DATA</td>
<td>Parse error on plug-in data: value.</td>
</tr>
<tr>
<td>ERR_PARSE_ERROR_STREAM</td>
<td>Parse error on input stream: value.</td>
</tr>
<tr>
<td>ERR_PCM_ERROR</td>
<td>PCM Error: err: value field: value loc value errclass: value rec_id: value resvd: value resvd2: value  value- value</td>
</tr>
<tr>
<td>ERR_PIPELINE NOT_USABLE</td>
<td>The pipeline 'value' is not usable; PIPELINE DEACTIVATED; please check the pipeline log for error messages and start the pipeline manually.</td>
</tr>
<tr>
<td>ERR_PLUGIN_NOT_FOUND</td>
<td>Invalid plugin name : value.</td>
</tr>
<tr>
<td>ERR_PLUGIN_NOT_VALID</td>
<td>The module 'value' is invalid and cannot be used.</td>
</tr>
<tr>
<td>ERR_PLUGIN_TYPE_INVALID</td>
<td>Module 'value' has a wrong type.</td>
</tr>
<tr>
<td>ERR_PREFIX_DATA_NO_DELIM</td>
<td>Invalid delimiter count in line value.</td>
</tr>
<tr>
<td>ERR_PREPARE_COMMIT_TRANSACTION</td>
<td>Module 'value' failed to prepare commit transaction 'value'.</td>
</tr>
<tr>
<td>ERR_PRICE_PLUGIN_INV</td>
<td>Price model data module invalid.</td>
</tr>
<tr>
<td>ERR_RATEPLAN_NOT_FOUND</td>
<td>Rateplan 'value' not found in rateplan data-module.</td>
</tr>
<tr>
<td>ERR_READ_DIR_FAILED</td>
<td>Error reading from directory 'value'; error message 'value'.</td>
</tr>
<tr>
<td>ERR_READING_CONTRACT_PERIOD</td>
<td>Can’t convert contract period length 'value' for cli 'value'.</td>
</tr>
</tbody>
</table>
### Pipeline Framework Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_READING_FILE</td>
<td>Error checking read line. Exception caught in <code>value</code>; <code>value</code>; <code>value</code>.</td>
</tr>
<tr>
<td>ERR_READING_SPECIALIST_SYSTEM</td>
<td>Can't convert specialist system number <code>value</code> for cli <code>value</code>.</td>
</tr>
<tr>
<td>ERR_READONLY_FILE_NOT_PROCESSED</td>
<td>File <code>value</code> is not writable, contents not processed.</td>
</tr>
<tr>
<td>ERR_REC_DESC_NOT_FOUND</td>
<td>Record description not found (<code>value</code>).</td>
</tr>
<tr>
<td>ERR_RECYCANCEL_FAILED</td>
<td>PreRecycle: Failed to cancel transaction <code>value</code>. Cannot find stream name <code>value</code> in the recycle map.</td>
</tr>
<tr>
<td>ERR_RECY_CANNOT_SET_ITEM_TYPE</td>
<td>PreRecycle: Cannot set the transaction item type for transaction <code>value</code>.</td>
</tr>
<tr>
<td>ERR_RECY_DELETE_TMPINPUT_FILE</td>
<td>PreRecycle: Cannot delete temporary input file <code>value</code>.</td>
</tr>
<tr>
<td>ERR_RECYFILE_NOT_INSERT</td>
<td>Could not insert file name <code>value</code> into hash table.</td>
</tr>
<tr>
<td>ERR_RECYFILE_NOTMOVED</td>
<td>The file <code>value</code> could not be moved to <code>value</code> for recycling.</td>
</tr>
<tr>
<td>ERR_RECYFILEOPEN</td>
<td>The recycle database file <code>value</code> can’t be opening.</td>
</tr>
<tr>
<td>ERR_RECYFILE_WRITE</td>
<td>Couldn't write to recycle database file. Try line <code>value</code> to insert.</td>
</tr>
<tr>
<td>ERR_RECYROLLBACK_FAILED</td>
<td>PreRecycle: Failed to rollback transaction <code>value</code>. Cannot move file <code>value</code>.</td>
</tr>
<tr>
<td>ERR_RECYTEST_FILE_NOT_COPY</td>
<td>The file <code>value</code> could not copy to <code>value</code> for test recycling.</td>
</tr>
<tr>
<td>ERR_REDO_POOL_ENTRY_NOT_FOUND</td>
<td>Redo pool entry <code>value</code> not found in function pool.</td>
</tr>
<tr>
<td>ERR_REFERENCENAME_NOT_IN_DEF</td>
<td>The reference name <code>value</code> is not in the alias description.</td>
</tr>
<tr>
<td>ERR_REGENTRY_NOT_FOUND</td>
<td>Registry entry <code>value</code> not found.</td>
</tr>
<tr>
<td>ERR_REGLOCKFILE_EXISTS</td>
<td>Registry lock file <code>value</code> already exists.</td>
</tr>
<tr>
<td>ERR_REGNAME_NOT_FOUND</td>
<td>Registry name <code>value</code> not found.</td>
</tr>
<tr>
<td>ERR_REGPARSEFAILED</td>
<td>Registry parse failed near <code>value</code>.</td>
</tr>
<tr>
<td>ERR_REGSUBTREE_NOT_FOUND</td>
<td>Registry subtree <code>value</code> not found.</td>
</tr>
<tr>
<td>ERR_REGUPDATEFAILED</td>
<td>Command processing failed for <code>value</code>.</td>
</tr>
<tr>
<td>ERR_REGVALUEINVALID</td>
<td>Registry entry <code>value</code> has invalid value <code>value</code>.</td>
</tr>
<tr>
<td>ERR_REGVALUEISEMPTY</td>
<td>Found empty value for registry item, where a value was expected.</td>
</tr>
<tr>
<td>ERR_REJECTSTREAMNOTDEFINED</td>
<td>Reject-stream not defined in <code>value</code>.</td>
</tr>
<tr>
<td>ERR_RENAMELOGFILE</td>
<td>Cannot rename old logfile.</td>
</tr>
<tr>
<td>ERR_RESOLVESTREAMNUMBER</td>
<td>Failure while resolving stream number for <code>value</code>.</td>
</tr>
<tr>
<td>ERR_RETURNPATHNOTFOUND</td>
<td>Entry for return path not found in registry.</td>
</tr>
<tr>
<td>ERR_ROLLBACKFAILEDINPUT</td>
<td>Rollback failed in input-controller.</td>
</tr>
<tr>
<td>ERR_ROLLBACKTRANSACTION</td>
<td>Module <code>value</code> failed to rollback transaction <code>value</code>.</td>
</tr>
<tr>
<td>ERR_SCRIPTNOTEXE</td>
<td>External program (<code>value</code>) is not executable.</td>
</tr>
<tr>
<td>ERR_SCRIPTNOTEXIST</td>
<td>Cannot find external program (<code>value</code>).</td>
</tr>
<tr>
<td>ERR_SEGMENTNOTDEFINED</td>
<td>No segment defined for <code>value</code>.</td>
</tr>
</tbody>
</table>
### Table 19–1 (Cont.) Pipeline Framework Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_SEQ_ALREADY_PROCESSED</td>
<td>Stream with sequence number 'value' was already processed.</td>
</tr>
<tr>
<td>ERR_SEQ_CHECK_FAILED</td>
<td>Sequence check failed.</td>
</tr>
<tr>
<td>ERR_SEQ_ENTRY_NOT_FOUND</td>
<td>Cannot find entry &quot;value&quot; in the sequence control file.</td>
</tr>
<tr>
<td>ERR_SEQ_FILE_INVALID</td>
<td>Error reading / parsing sequence number file 'value'.</td>
</tr>
<tr>
<td>ERR_SEQ_GAP</td>
<td>Sequence number 'value' is too high.</td>
</tr>
<tr>
<td>ERR_SEQ_INIT</td>
<td>Default sequence file generated. Please check file content.</td>
</tr>
<tr>
<td>ERR_SEQ_MASTER_CONTROL</td>
<td>False master controller type in 'value' configured.</td>
</tr>
<tr>
<td>ERR_SEQ_MASTER_CONTROLLER</td>
<td>Unknown or wrong master controller for sequence sharing.</td>
</tr>
<tr>
<td>ERR_SEQ_MIN_GREATER_MAX</td>
<td>The min sequence number is greater than the max sequence number.</td>
</tr>
<tr>
<td>ERR_SEQ_SAVE</td>
<td>Error saving sequence information to stream.</td>
</tr>
<tr>
<td>ERR_SETUP_CALLTYPE</td>
<td>Failure during setup of calltype table.</td>
</tr>
<tr>
<td>ERR_SETUP_CZT_MAPTABLE</td>
<td>Error while setting up CZT map table from database.</td>
</tr>
<tr>
<td>ERR_SETUP_EDRFACRY</td>
<td>EDR factory setup failed: 'value'.</td>
</tr>
<tr>
<td>ERR_SETUP_FSM</td>
<td>Failure during setup of finite state machine.</td>
</tr>
<tr>
<td>ERR_SETUP_INPUT_GRAMMAR</td>
<td>Input grammar setup failed: 'value'.</td>
</tr>
<tr>
<td>ERR_SETUP_OUTPUT</td>
<td>Error setup output line. Exception caught in 'value'; 'value' (line='value').</td>
</tr>
<tr>
<td>ERR_SETUP_OUTPUT_GRAMMAR</td>
<td>Output grammar setup failed: 'value'.</td>
</tr>
<tr>
<td>ERR_SHUTDOWN_FAIL_TO_COMPLETE</td>
<td>Shutdown request fails to complete.</td>
</tr>
<tr>
<td>ERR_SOURCE_VALUE</td>
<td>Source parameter must be either 'Database' or 'File'.</td>
</tr>
<tr>
<td>ERR_SPECIAL_FUNCTIONS_FAILED</td>
<td>The routine 'specialFunctions' in pipeline 'value' failed.</td>
</tr>
<tr>
<td>ERR_STR_LEAVING_THREAD</td>
<td>Critical stream error. Shutting down pipeline.</td>
</tr>
<tr>
<td>ERR_STREAM_NOT_FOUND</td>
<td>Could not create any statistic informations for this device.</td>
</tr>
<tr>
<td>ERR_STREAM_TO_EDR_FAILED</td>
<td>Stream to EDR conversion failed. EDR container created, but not written to input buffer.</td>
</tr>
<tr>
<td>ERR_SYSCATALOG_ENTRY_NOT_FOUND</td>
<td>System catalog entry 'value' not found</td>
</tr>
<tr>
<td>ERR_SYSTEM_ERROR</td>
<td>Unexpected error, 'value'.</td>
</tr>
<tr>
<td>ERR_TAM_ABORT_REQUESTED</td>
<td>Abort requested for transaction manager 'value'.</td>
</tr>
<tr>
<td>ERR_TAM_ENTRY_NOT_FOUND</td>
<td>Cannot find entry &quot;value&quot; in the transaction manager map.</td>
</tr>
<tr>
<td>ERR_TAM_ENTRY_NOT_REMOVED</td>
<td>Cannot remove entry &quot;value&quot; from the transaction manager map.</td>
</tr>
<tr>
<td>ERR_TAM_FILE_READ_ERR</td>
<td>Error reading from binary transaction log file 'value', message 'value'.</td>
</tr>
<tr>
<td>ERR_TAM_FILE_WRITE_ERR</td>
<td>Error writing into binary transaction log file 'value', message 'value'.</td>
</tr>
<tr>
<td>ERR_TAM_INIT_FAILED</td>
<td>Failed to init transaction manager 'value'.</td>
</tr>
<tr>
<td>Error message</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ERR_TAM_STREAM_NOT_FOUND</td>
<td>Cannot find stream name &quot;value&quot; for transaction id &quot;value&quot;.</td>
</tr>
<tr>
<td>ERR_TAP3_FATAL</td>
<td>TAP3 Fatal: Field Name: value, Tag: value, Error Code: value, Description: value.</td>
</tr>
<tr>
<td>ERR_TAP3_RET</td>
<td>TAP3 return: sev(value), errCode(value), tag(value), depth(value), offset(value), arrayIdx(value), ruleID(value).</td>
</tr>
<tr>
<td>ERR_TAP3_SEVERE</td>
<td>TAP3 Severe: Field Name: value, Tag: value, Error Code: value, Description: value.</td>
</tr>
<tr>
<td>ERR_TAP3_WARNING</td>
<td>TAP3 Warning: Field Name: value, Tag: value, Error Code: value, Description: value.</td>
</tr>
<tr>
<td>ERR_TARIFF_PLUGIN_INV</td>
<td>Tariff model data module invalid.</td>
</tr>
<tr>
<td>ERR_TEMP_FILE_NOT_MOVED</td>
<td>Cannot move temporary input file 'value'.</td>
</tr>
<tr>
<td>ERR_THREAD_EXCEPTION</td>
<td>Exception detected in 'value'; value; value.</td>
</tr>
<tr>
<td>ERR_THREAD_STACKSET_FAILED</td>
<td>Failed to set stack size of thread</td>
</tr>
<tr>
<td>ERR_TIME_PLUGIN_INV</td>
<td>Time model data module invalid.</td>
</tr>
<tr>
<td>ERR_TMPFILE_NOT_MOVED</td>
<td>Temporary file 'value' could not be moved to 'value'.</td>
</tr>
<tr>
<td>ERR_TOKEN_ACCESS_TIMEOUT</td>
<td>Timeout while accessing HA role mediator token for read or update.</td>
</tr>
<tr>
<td>ERR_TOKEN_READ_FAILED</td>
<td>Failed to read HA role mediator token.</td>
</tr>
<tr>
<td>ERR_TOKEN_UPDATE_FAILED</td>
<td>Failed to update HA role mediator token.</td>
</tr>
<tr>
<td>ERR_TRACE_START_POINT_NOT_FOUND</td>
<td>TraceStartPoint not found.</td>
</tr>
<tr>
<td>ERR_TRACEPOINTS</td>
<td>TraceEndPoint is less than TraceStartPoint.</td>
</tr>
<tr>
<td>ERR_TRANS_ID_REG_ENTRY_NOT_FOUND</td>
<td>Registry entry 'value' not found in transaction id information file 'value'.</td>
</tr>
<tr>
<td>ERR_TRANS_ID_REG_INVALID_VALUE</td>
<td>Invalid value 'value' for 'value' in transaction id information file 'value'.</td>
</tr>
<tr>
<td>ERR_TRANSFER_CUTOFF_VIOLATED</td>
<td>TransferCutOff Date (value) violated with value.</td>
</tr>
<tr>
<td>ERR_UNKNOWN_ALIGNMENT</td>
<td>Unknown alignment text in value. It set to left.</td>
</tr>
<tr>
<td>ERR_UNKNOWN_COL_TYPE</td>
<td>Unknown colType (value) in section info.</td>
</tr>
<tr>
<td>ERR_UNKNOWN_DEFAULT_SENTENCE</td>
<td>Output Stream value : The default line couldn't be identified. Check your format description.</td>
</tr>
<tr>
<td>ERR_UNKNOWN_DISCARD_FKT</td>
<td>Valid functions are [Discard or Skip]</td>
</tr>
<tr>
<td>ERR_UNKNOWN_EVENT_TYPE</td>
<td>Event value has unknown event type.</td>
</tr>
<tr>
<td>ERR_UNKNOWN_FIELD_NAME</td>
<td>Unknown field name (value) in section info.</td>
</tr>
<tr>
<td>ERR_UNKNOWN_ROW_TYPE</td>
<td>Unknown rowType (value) in section info.</td>
</tr>
<tr>
<td>ERR_UNKNOWN_SPLITTING_RULES</td>
<td>Unknown type of splitting rules 'value'.</td>
</tr>
<tr>
<td>ERR_USR_PROCESS_KILLED</td>
<td>Killed external process 'value' after it timed out.</td>
</tr>
<tr>
<td>ERR_VALUE_CONV_FAIL</td>
<td>Error converting value(s): value.</td>
</tr>
<tr>
<td>ERR_VERSION_CHECK_FAILED</td>
<td>Version check for database 'value' and 'value' failed.</td>
</tr>
<tr>
<td>ERR_WRITE_DEF_EDR_NOT_FOUND</td>
<td>Registry entry 'WriteDefaultEdr' not found.</td>
</tr>
<tr>
<td>ERR_WRITE_FILE</td>
<td>Cannot create/write file 'value'.</td>
</tr>
</tbody>
</table>
### Table 19–1 (Cont.) Pipeline Framework Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_WRONG_TOKEN_COUNT</td>
<td>Wrong token count in input file. Line: <em>value</em></td>
</tr>
<tr>
<td>ERR_XML_INPUT_MAPPING_FAILED</td>
<td>EDR XML generation failed: Input mapping <em>value</em> failed: <em>value</em></td>
</tr>
<tr>
<td>ERR_XML_PARSE_EDR</td>
<td>Exception parsing XML: near Attribute: <em>value</em> <em>value</em></td>
</tr>
<tr>
<td>ERR_XML_PARSE_SAX</td>
<td>Exception parsing XML: Line: <em>value</em> Column: <em>value</em> <em>value</em></td>
</tr>
<tr>
<td>ERR_XML_PARSE_UNKNOWN</td>
<td>Unknown exception parsing XML</td>
</tr>
<tr>
<td>ERR_XML_PARSE_XML</td>
<td>Exception parsing XML: <em>value</em></td>
</tr>
<tr>
<td>ERR_ZONE_PLUGIN_INV</td>
<td>Zone model data module invalid.</td>
</tr>
<tr>
<td>ERR_ZONENTRY_NOT_FOUND</td>
<td>Cannot find entry in zone model <em>value</em> for origin <em>value</em>, destin <em>value</em>, call date <em>value</em> and service <em>value</em></td>
</tr>
<tr>
<td>ERR_ZONEMODEL_NOT_CONFIGURED</td>
<td>Zone model <em>value</em> has not been configured.</td>
</tr>
<tr>
<td>ERR_ZONEMODEL_NOT_FOUND</td>
<td>Zonemodel-Id <em>value</em> not found in zone data-module.</td>
</tr>
<tr>
<td>ERR_ZONETREE_NOT_FOUND</td>
<td>Cannot find digit tree in configuration data for zone model <em>value</em></td>
</tr>
<tr>
<td>FORMAT_DESC_IS_INCOMPLETE</td>
<td>The format description is incomplete (HEADER, DETAIL, TRAILER).</td>
</tr>
<tr>
<td>INVALID_FORMAT_DESC</td>
<td>The format description for <em>value</em> is invalid.</td>
</tr>
<tr>
<td>UNKNOWN_LOGLEVEL</td>
<td>The specified log level is unknown. Valid values are normal, warning, minor, major and critical.</td>
</tr>
<tr>
<td>WRN_CANNOT_DETERMINE_OUTSTREAMNAME</td>
<td>Cannot determine the output file name for streamname <em>value</em>.</td>
</tr>
<tr>
<td>WRN_CCENTRY_INVALID</td>
<td>Invalid call class map entry: <em>value</em>.</td>
</tr>
<tr>
<td>WRN_CLI_NOT_FOUND</td>
<td>CLI <em>value</em> not found.</td>
</tr>
<tr>
<td>WRN_CONTRACT_NOT_FOUND</td>
<td>Contract <em>value</em> not found.</td>
</tr>
<tr>
<td>WRN_CZTENTRY_INVALID</td>
<td>Invalid CZT map entry: <em>value</em>.</td>
</tr>
<tr>
<td>WRN_DEST_CLI_NOT_FOUND</td>
<td>Destination cli <em>value</em> not found.</td>
</tr>
<tr>
<td>WRN_EQUAL_TARIFFIND_DATE</td>
<td>Both tariff indicators have same date for contract <em>value</em>.</td>
</tr>
<tr>
<td>WRN_FILE_REMOVE_OS</td>
<td><em>value</em>: Cannot remove file <em>value</em>.</td>
</tr>
<tr>
<td>WRN_ILLEGAL_SPECIALDAYRATE</td>
<td>Illegal values in special dayrate <em>value</em>.</td>
</tr>
<tr>
<td>WRN_INVALID_ACTIVATED_DATE</td>
<td>Invalid activation date, ignoring contract <em>value</em>.</td>
</tr>
<tr>
<td>WRN_INVALID_CLI</td>
<td>Ignoring invalid cli <em>value</em>.</td>
</tr>
<tr>
<td>WRN_INVALID_CLI_RANGE</td>
<td>Ignoring invalid cli range (value).</td>
</tr>
<tr>
<td>WRN_INVALID_HISTORY_DATE</td>
<td>Contract <em>value</em> has an invalid history date <em>value</em>, using <em>value</em></td>
</tr>
<tr>
<td>WRN_NO_ENDTRANSACTION</td>
<td>A beginTransaction arrives before the endTransaction in <em>value</em>.</td>
</tr>
<tr>
<td>WRN_NO_SEQUENCE_NUMBER_ADDEDTO_</td>
<td>No new sequencenumber generated for sequence.</td>
</tr>
<tr>
<td>TRANSACTION</td>
<td>WRN_NO_STREAMLOG_DEFINED</td>
</tr>
<tr>
<td>WRN_NO_VALID_ENTRY</td>
<td>Entry <em>value</em> in file <em>value</em> is invalid and ignored.</td>
</tr>
<tr>
<td>WRN_REG_ENTRY_OBSOLETE</td>
<td>Obsolete registry entry: <em>value</em></td>
</tr>
</tbody>
</table>
Table 19–2 lists the DAT_AccountBatch error messages.

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WRN_SEMAPHORE_NOT_PROCESSED</td>
<td>Semaphore was not processed; check spelling.</td>
</tr>
<tr>
<td>WRN_TXNLOGGING_OFF</td>
<td>Transaction logging is off, make sure that you are doing testing only!</td>
</tr>
<tr>
<td>WRN_ZONEMAP_INVALID</td>
<td>Invalid zone map entry: value.</td>
</tr>
<tr>
<td>ERR_UNLINK_FILE_ERROR</td>
<td>Error value while attempting to unlink of temp file: value</td>
</tr>
<tr>
<td>ERR_OPEN_FILE_ERROR</td>
<td>Error value while attempting to open of temp file: value</td>
</tr>
<tr>
<td>ERR_WRITE_FILE_ERROR</td>
<td>Error value while attempting to write of temp file: value</td>
</tr>
<tr>
<td>ERR_CLOSE_FILE_ERROR</td>
<td>Error value while attempting to close of temp file: value</td>
</tr>
<tr>
<td>ERR_RENAME_FILE_ERROR</td>
<td>Error value while attempting to rename of temp file: value</td>
</tr>
<tr>
<td>ERR_TXN_TIMEOUT</td>
<td>Timeout while waiting for the next request in a transaction: value</td>
</tr>
<tr>
<td>ERR_RELEASE_OBJ_LOCK</td>
<td>Error while releasing lock for object: value</td>
</tr>
<tr>
<td>ERR_REPLENISH_POID_CACHE_FAILED</td>
<td>Error while processing poids: value.</td>
</tr>
<tr>
<td>ERR_PROCESS_EXIT</td>
<td>Attempt to exit process due to signal.</td>
</tr>
<tr>
<td>ERR_DELETION_ASS_CBD_FAILURE</td>
<td>Failure in deletion of ASS_CBD block.</td>
</tr>
<tr>
<td>ERR_DELETION_CP_FAILURE</td>
<td>Failure in deletion of CP block.</td>
</tr>
<tr>
<td>ERR_DELETION_TP_FAILURE</td>
<td>Failure in deletion of TP block.</td>
</tr>
<tr>
<td>ERR_CONNECT_REJECTED</td>
<td>Connect from ‘value’ rejected</td>
</tr>
<tr>
<td>ERR_INCORRECT_FILE_NAME_SPECIFICATION</td>
<td>Error encountered in building the output file name from the given specification: ‘value’ for the input file - ‘value’. Defaulting to regular file naming technique for this file .</td>
</tr>
<tr>
<td>ERR_IGNORE_REGISTRY_ENTRY</td>
<td>Registry entry - Name : ‘value’ Value : ‘value’ is ignored value</td>
</tr>
<tr>
<td>ERR_START_OVERLOAD_DETECTION</td>
<td>Failed to start/restart overload detection, value.</td>
</tr>
<tr>
<td>ERR_ZONE_VALUE_NOT_FOUND</td>
<td>ZoneValue not found for ZM-Id=value, Date=value, SC=value, A#=value, B#=value.</td>
</tr>
<tr>
<td>ERR_SESSION_PUT_ON_HOLD</td>
<td>Session value is put on hold due to being passive.</td>
</tr>
<tr>
<td>ERR_SESSION_REJECTED</td>
<td>Session value is rejected due to being passive.</td>
</tr>
</tbody>
</table>

Pipeline Manager Module Error Messages

Table 19–2 lists the DAT_AccountBatch error messages.
## Pipeline Manager Module Error Messages

**Table 19–2  DAT_AccountBatch Error Messages**

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_ACCOUNT_DB_UPDATE_FAILED</td>
<td>Database update failed for account ((value)) at time ((value)).</td>
</tr>
<tr>
<td>ERR_ACCOUNTBUSY_ACCOUNT_OBJ_FIELD_NOT_FOUND</td>
<td>Busy account obj field not found in flist for job: (value).</td>
</tr>
<tr>
<td>ERR_ACCOUNTBUSY_BATCH_OBJ_NOT_FOUND</td>
<td>Batch obj not found in flist for job: (value).</td>
</tr>
<tr>
<td>ERR_ACCOUNTBUSY_BATCH_OBJ_NOT_FOUND</td>
<td>Batch obj not found in flist for job: (value).</td>
</tr>
<tr>
<td>ERR_ACCOUNTBUSY_JOB_ALREADY_REMOVED</td>
<td>Busy job with the id does not exists: (value).</td>
</tr>
<tr>
<td>ERR_ACCOUNTBUSY_JOB_ALREADY_REMOVED</td>
<td>Busy job with the id does not exists : (value).</td>
</tr>
<tr>
<td>ERR_ACCOUNTBUSY_JOB_ID_FIELD_NOT_FOUND</td>
<td>Busy job field not found in flist: (value).</td>
</tr>
<tr>
<td>ERR_ACCOUNTBUSY_JOB_ID_FIELD_NOT_FOUND</td>
<td>Busy job field not found in flist: (value).</td>
</tr>
<tr>
<td>ERR_BAD_VALUE</td>
<td>Null object poid in AddOrderedBalanceGroup event</td>
</tr>
<tr>
<td>ERR_BALANCE_GR_NOT_FOUND</td>
<td>Balance group not found for given ID</td>
</tr>
<tr>
<td>ERR_BALANCE_GROUP_UPDATE</td>
<td>Customer balance group update error ((value)).</td>
</tr>
<tr>
<td>ERR_BILL_INFO_UPDATE</td>
<td>Customer blinfo update error ((value)).</td>
</tr>
<tr>
<td>ERR_BILLING_INFO_NOT_FOUND</td>
<td>Did not find billing information</td>
</tr>
<tr>
<td>ERR_CUSTOMER_ACCOUNT_NOT_FOUND</td>
<td>Customer Account not found ((value)).</td>
</tr>
<tr>
<td>ERR_CUSTOMER_EDR_PARSING</td>
<td>Customer EDR parsing error ((value)).</td>
</tr>
<tr>
<td>ERR_CUSTOMER_INVALID_ITEM_POID</td>
<td>Customer item POID not valid ((value)).</td>
</tr>
<tr>
<td>ERR_CUSTOMER_LOGIN_ACCOUNT_NOT_FOUND</td>
<td>Customer account not found after login ((value)).</td>
</tr>
<tr>
<td>ERR_CUSTOMER_LOGIN_INTERNAL_ERROR</td>
<td>Customer login internal error ((value)).</td>
</tr>
<tr>
<td>ERR_CUSTOMER_LOGIN_NOT_FOUND</td>
<td>Customer login not found ((value)).</td>
</tr>
<tr>
<td>ERR_CUSTOMER_LOGIN_NOT_VALID_FOR_TIME</td>
<td>Customer login not valid for time ((value)).</td>
</tr>
<tr>
<td>ERR_CUSTOMER_LOGIN_SERVICE_NOT_FOUND</td>
<td>Customer service not found ((value)).</td>
</tr>
<tr>
<td>ERR_CUSTOMER_NO_VALID_PRODUCT</td>
<td>Customer product not valid ((value)).</td>
</tr>
<tr>
<td>ERR_CUSTOMER_NO_VALID_PRODUCT_RATING</td>
<td>Customer product rating not valid ((value)).</td>
</tr>
<tr>
<td>ERR_CUSTOMER_SERVICE_NOT_FOUND</td>
<td>Customer Service not found ((value)).</td>
</tr>
<tr>
<td>ERR_DISCOUNT_DATA_STRING</td>
<td>Error building discount data string for service ((value)).</td>
</tr>
<tr>
<td>ERR_DUPLICATE_ACCOUNTBUSY_JOB_ADDED</td>
<td>Busy job with the id already exists : (value).</td>
</tr>
<tr>
<td>ERR_EVENT_ORDER_MISSING_IN_MEMORY_PROFILE</td>
<td>EventOrder Profile object is missing for account ((value)).</td>
</tr>
<tr>
<td>ERR_EVENT_ORDER_MISSING_SCRATCH_PAD_ITEM</td>
<td>EventOrderImpl::doUpdateEventOrderData() is unable to find the ScratchPadItem with moniker ((value)) and pipeline/transaction ((value)).</td>
</tr>
<tr>
<td>ERR_FIRST_USAGE_ITEM_ALREADY_COMMITTED</td>
<td>First Usage product/discount with id ((value)). has been already committed in the pipeline.</td>
</tr>
<tr>
<td>ERR_FIRST_USAGE_OBJECT_ALREADY_INITIALIZED</td>
<td>First Usage product/discount with id ((value)) has been already used and initialized in probably same or different transaction, so not initializing the validity with current time.</td>
</tr>
<tr>
<td>Error message</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ERR_GET_RANGE_ITEMS</td>
<td>DAT_LoginDbObject::getPoidRangeItems failure: Reason=value</td>
</tr>
<tr>
<td>ERR_INITAccounts_CANCELLED</td>
<td>Thread=value has cancelled in DAT::InitCustomerThread::initAccounts method - value</td>
</tr>
<tr>
<td>ERR_INIT_BALANCE_GROUPS_CANCELLED</td>
<td>Thread=value has cancelled in DAT::InitCustomerThread::initBalanceGroups method - value</td>
</tr>
<tr>
<td>ERR_INIT_BILL_INFOS_CANCELLED</td>
<td>Thread=value has cancelled in DAT::InitCustomerThread::initBillInfo method - value</td>
</tr>
<tr>
<td>ERR_INIT_DELETED_ORDERED_BALANCE_GROUPS_CANCELLED</td>
<td>Thread=value has cancelled in DAT::InitCustomerThread::initDeletedOrderedBalance Groups method - value</td>
</tr>
<tr>
<td>ERR_INIT_GROUP_SHARING_CHARGES_CANCELLED</td>
<td>Thread=value has cancelled in DAT::InitCustomerThread::initGroupSharingCharges method - value</td>
</tr>
<tr>
<td>ERR_INIT_GROUP_SHARING_DISCOUNTS_CANCELLED</td>
<td>Thread=value has cancelled in DAT::InitCustomerThread::initGroupSharingDiscounts method - value</td>
</tr>
<tr>
<td>ERR_INIT_GROUP_SHARING_PROFILES_CANCELLED</td>
<td>Thread=value has cancelled in DAT::InitCustomerThread::initGroupSharingProfiles method - value</td>
</tr>
<tr>
<td>ERR_INIT_LOGIN_CANCELLED</td>
<td>Thread=value has cancelled in DAT::InitCustomerThread::initLogins method - value</td>
</tr>
<tr>
<td>ERR_INIT_MAPPING_TABLES_CANCELLED</td>
<td>Thread=value has cancelled in DAT::InitCustomerThread::initMappingTable method - value</td>
</tr>
<tr>
<td>ERR_INIT_ORDERED_BALANCE_GROUPS_BILLINFO</td>
<td>Thread=value has inconsistent data in DAT::InitCustomerThread::initOrderedBalanceGroups methodfor account - value</td>
</tr>
<tr>
<td>ERR_INIT_ORDERED_BALANCE_GROUPS_CANCELLED</td>
<td>Thread=value has cancelled in DAT::InitCustomerThread::initOrderedBalanceGroups method - value</td>
</tr>
<tr>
<td>ERR_INIT_PROFILES_CANCELLED</td>
<td>Thread=value has cancelled in DAT::InitCustomerThread::initProfiles method - value</td>
</tr>
<tr>
<td>ERR_INIT_PURCHASED_DISCOUNTS_CANCELLED</td>
<td>Thread=value has cancelled in DAT::InitCustomerThread::initPurchasedDiscounts method - value</td>
</tr>
<tr>
<td>ERR_INIT_PURCHASED_PRODUCTS_CANCELLED</td>
<td>Thread=value has cancelled in DAT::InitCustomerThread::initPurchasedProducts method - value</td>
</tr>
<tr>
<td>ERR_INIT_SERVICE_CANCELLED</td>
<td>Thread=value has cancelled in DAT::InitCustomerThread::initServices method - value</td>
</tr>
<tr>
<td>ERR_INIT_THREAD_DIED</td>
<td>Thread-value has died with an expection in DAT_InitCustomerThread::run() - value</td>
</tr>
<tr>
<td>ERR_INSERTING_CUST_CREATE_EVENT_ORDER_PROFILE</td>
<td>Unable to insert EventOrderProfile (value) for newly created account (value) during a CustCreate event.</td>
</tr>
<tr>
<td>ERR_INVALID_ENTRY_FOR_BUSINESS_PARAM</td>
<td>Invalid value for Business Parameter: value Value: value</td>
</tr>
<tr>
<td>ERR_INVALID_OUTPUT_STREAM</td>
<td>Invalid output stream (value).</td>
</tr>
<tr>
<td>ERR_INVALID_TYPE_CAST</td>
<td>Error on type cast value.</td>
</tr>
<tr>
<td>Error message</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ERR_LISTENER_NOT_FOUND</td>
<td>Listener 'value' not found.</td>
</tr>
<tr>
<td>ERR_LOOKUP_CONSTANT_ITEM</td>
<td>Cannot find requested item (value) in the ConstantItem Pool.</td>
</tr>
<tr>
<td>ERR_MAP_MERGE_THREAD_CANCELLED</td>
<td>DAT_MapMergeThread::value has cancelled because one child thread died with exception - value</td>
</tr>
<tr>
<td>ERR_MAP_MERGE_THREAD_DIED</td>
<td>DAT_MapMergeThread::value has died with an exception: value</td>
</tr>
<tr>
<td>ERR_MAPPINg_TABLE_UPDATE</td>
<td>Customer mapping table update error (value).</td>
</tr>
<tr>
<td>ERR_MULTI_THREAD_INIT</td>
<td>DAT_InitCustomerThread failed to create and start: Reason=value</td>
</tr>
<tr>
<td>ERR_MULTI_THREAD_MAP_MERGE</td>
<td>DAT_MapMergeThread failed to create and start: Reason=value</td>
</tr>
<tr>
<td>ERR_NOT_ENOUGH_CONNECTIONS</td>
<td>Not enough connections available to coincide with the &quot;Threads&quot; registry value. Define a Connections registry value greater than or equal to the Threads value.</td>
</tr>
<tr>
<td>ERR_OBG_RESOLVE_ID</td>
<td>Error in resolving OBG Id (value).</td>
</tr>
<tr>
<td>ERR_REQUIRED_REG_ENTRY_MISSING</td>
<td>A required registry entry is missing.</td>
</tr>
<tr>
<td>ERR_REQUIRED_REG_ENTRY_NOT_CONFIGURED</td>
<td>Entry not configured for DAT_Account: value.</td>
</tr>
<tr>
<td>ERR_RERATING_IN_PROGRESS</td>
<td>ReRating is currently running, EDR not processed : value</td>
</tr>
<tr>
<td>ERR_RW_DAT_ACCOUNT_EXCEPTION</td>
<td>CreateScratchPadItem() failed because item with resourceHash=value for Pipeline/Transaction=value already exists.</td>
</tr>
<tr>
<td>ERR_SCRATCH_PAD_ITEM_ALREADY_EXISTS</td>
<td>CreateScratchPadItem() failed because item with resourceHash=value for Pipeline/Transaction=value already exists.</td>
</tr>
<tr>
<td>ERR_SCRATCH_PAD_ITEM_IS_READ_DIRTY</td>
<td>WriteData() failed because item with resourceHash=value has been updated by another transaction.</td>
</tr>
<tr>
<td>ERR_SCRATCH_PAD_ITEM_LOCKED_BY_ANOTHER_TRANSACTION</td>
<td>Unable to Read or Write ScratchPadItem because it is locked by another transaction. ResourceHash=value Pipeline/Transaction=value</td>
</tr>
<tr>
<td>ERR_SCRATCH_PAD_ITEM_NOT_FOUND</td>
<td>ScratchPadItem not found. ResourceHash=value. Pipeline Transaction Hash=value. FunctionName=value</td>
</tr>
<tr>
<td>ERR_SCRATCH_PAD_ITEM_NULL</td>
<td>Attempting to call AdoptAndLock() will a NULL ScratchPadItem. Pipeline Transaction Hash=value</td>
</tr>
<tr>
<td>ERR_SCRATCH_PAD_NOT_FOUND</td>
<td>ScratchPad not found. Pipeline Transaction Hash=value. FunctionName=value</td>
</tr>
<tr>
<td>ERR_SERVICE_DB_UPDATE_FAILED</td>
<td>Account database update failed</td>
</tr>
<tr>
<td>ERR_SERVICE_NOT_CONFIGURED</td>
<td>Service not found (value).</td>
</tr>
<tr>
<td>ERR_SERVICE_OBJECT_NOT_FOUND</td>
<td>Service object not found for particular service Id : value</td>
</tr>
<tr>
<td>ERR_SERVICE_OBJECT_UPDATE_FAILED</td>
<td>Service object update failed for a particular service id : value</td>
</tr>
<tr>
<td>ERR_SUBSCRIPTION_SERVICE_NOT_FOUND</td>
<td>Subscription service not found (value).</td>
</tr>
</tbody>
</table>
Table 19–3 lists the DAT_AccountRealtime error messages.

Table 19–3  DAT_AccountRealtime Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_THREAD_CANCELLED</td>
<td>Thread=value has cancelled because some child thread gets an error in DAT_InitCustomerThread::run(): Info=value. A fatal error occurred in one child thread during DAT_AccountBatch multi-threaded initialization. When this occurs, other non-failing threads safely shutdown and, as a result, this error message is displayed. There is no immediate resolution.</td>
</tr>
<tr>
<td>ERR_THREAD_DIED_UNEXPECTED</td>
<td>A fatal error occurred during DAT_AccountBatch multi-threaded initialization. There is no immediate resolution; instead, a help ticket should be opened.</td>
</tr>
<tr>
<td>ERR_UNKNOWN_DAT_ACCOUNT_EXCEPTION</td>
<td>Unknown Exception encountered in: (value)</td>
</tr>
<tr>
<td>ERR_UPDATEBILLING_STATE_BADDATASTRING</td>
<td>BillInfo::updateBillingState() failed with invalid dataStringM (value)</td>
</tr>
<tr>
<td>WRN_EVENT_PROCESSING_FORBILLINFO FAILED</td>
<td>BillInfo update failed for account ID : value</td>
</tr>
<tr>
<td>WRN_INVALIDACCOUNT_INPROFILE</td>
<td>Warning, one or more account profiles point to an invalid account ID.</td>
</tr>
<tr>
<td>WRN_INVALIDSERVICE_INPROFILE</td>
<td>Warning, one or more service profiles point to an invalid service ID.</td>
</tr>
<tr>
<td>WRN_MODIFYPROFILE_NOSERVICE</td>
<td>Warning, unable to complete the Modify/CreateProfile event. Cannot locate the ERA's Service Object in the CustomerData ServiceMap. ServiceID = value.</td>
</tr>
<tr>
<td>WRN_MULTI_THREAD_MAP_MERGE</td>
<td>Login value found in multiple threads but update failed during map merge</td>
</tr>
<tr>
<td>WRN_REGENTRY_INVALID</td>
<td>Warning, Registry entry for DAT_Account is invalid value</td>
</tr>
<tr>
<td>WRN_SYSTEMPRODUCT_MAPISEMPTY</td>
<td>System product map is empty : value</td>
</tr>
<tr>
<td>WRN_SYSTEMPRODUCTNOTFOUND</td>
<td>System product not found from system product map for particular product Id. : value</td>
</tr>
</tbody>
</table>

DAT_AccountRealtime

Table 19–3 lists the DAT_AccountRealtime error messages.

Table 19–3  DAT_AccountRealtime Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_ACCCRT_MESSAGE</td>
<td>Error message for DAT_AccountRealtime plugin module: ‘value’.</td>
</tr>
<tr>
<td>ERR_NOTIMPLEMENTED</td>
<td>Method not implemented in DAT_AccountRealtime plugin module: ‘value’.</td>
</tr>
</tbody>
</table>

DAT_BalanceBatch

Table 19–4 lists the DAT_BalanceBatch error messages.
### Table 19–4  DAT_BalanceBatch Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_BALANCE_ATTACH_TRANS_MODULE</td>
<td>Cannot attach transaction module \textit{value}.</td>
</tr>
<tr>
<td>ERR_BALANCE_DATABASE</td>
<td>Database operation failed in DAT_BalanceBatch: \textit{value}</td>
</tr>
<tr>
<td>ERR_BALANCE_DETACH_MODULE</td>
<td>Could not detach pipeline \textit{value} (not attached).</td>
</tr>
<tr>
<td>ERR_BALANCE_GET_POID_RANGE</td>
<td>Get poidrange failure: Reason=\textit{value}</td>
</tr>
<tr>
<td>ERR_BALANCE_INIT_THREAD_DIED</td>
<td>Thread=\textit{value} has died with an exception in DAT_InitCustomerThread::run(): Run() - \textit{value}</td>
</tr>
<tr>
<td>ERR_BALANCE_INVALID_BALANCEDATA</td>
<td>Invalid balance data during \textit{value}.</td>
</tr>
<tr>
<td>ERR_BALANCE_INVALID_EDRTRANSACTION</td>
<td>No transaction for this EDR on the transaction list in \textit{value}.</td>
</tr>
<tr>
<td>ERR_BALANCE_INVALID_STATE</td>
<td>Invalid transaction state, transId \textit{value}.</td>
</tr>
<tr>
<td>ERR_BALANCE_INVALID_TRANSACTION</td>
<td>Invalid transaction during \textit{value}.</td>
</tr>
<tr>
<td>ERR_BALANCE_INVALID_TRANSACTIONDATA</td>
<td>Invalid transaction data during \textit{value}.</td>
</tr>
<tr>
<td>ERR_BALANCE_LISTENER_NOT_FOUND</td>
<td>Listener \textit{value} not found.</td>
</tr>
<tr>
<td>ERR_BALANCE_MERGE_THREAD_DIED</td>
<td>DAT_MapMergeThread::\textit{value} has died with an exception: \textit{value}</td>
</tr>
<tr>
<td>ERR_BALANCE_MESSAGE</td>
<td>Error message for DAT_Balance plugin module: \textit{value}.</td>
</tr>
<tr>
<td>ERR_BALANCE_MISSING_BALANCE_GROUP</td>
<td>Balance group missing.</td>
</tr>
<tr>
<td>ERR_BALANCE_PROCESS_EVENT_ERROR</td>
<td>Could not process event because there is an unknown error for event: \textit{value}.</td>
</tr>
<tr>
<td>ERR_BALANCE_PROCESSING_EVENT_BEGIN</td>
<td>Could not begin event transaction for event Id \textit{value}.</td>
</tr>
<tr>
<td>ERR_BALANCE_PROCESSING_EVENT_COMMIT</td>
<td>Could not commit event transaction for event Id \textit{value}.</td>
</tr>
<tr>
<td>ERR_BALANCE_PROCESSING_EVENT_ROLLBACK</td>
<td>Could not rollback event transaction for event Id \textit{value}.</td>
</tr>
<tr>
<td>ERR_BALANCE_THREAD_DIED_UNEXPECTED</td>
<td>Thread=\textit{value} unexpectedly died: Info=\textit{value}</td>
</tr>
<tr>
<td>ERR_BALANCE_THREAD_INIT</td>
<td>Initial Thread for loading balance failed to create and start: Reason=\textit{value}</td>
</tr>
<tr>
<td>ERR_BALANCE_THREAD_MERGE</td>
<td>Merge Thread for loading balance failed to create and start: Reason=\textit{value}</td>
</tr>
<tr>
<td>ERR_BALANCE_TRANSACTION_MISMATCH</td>
<td>Transactions mismatch.</td>
</tr>
<tr>
<td>ERR_BALANCE_UPDATE_BALANCE</td>
<td>Error while updating the balance.</td>
</tr>
<tr>
<td>WRN_BALANCE_DEADLOCK_BTN_EDRTRANS</td>
<td>Deadlock between edr transactions on \textit{value}.</td>
</tr>
<tr>
<td>WRN_BALANCE_DEADLOCK_BTN_TRANS</td>
<td>Deadlock between currentpipelineId \textit{value} and another pipelineId \textit{value}.</td>
</tr>
<tr>
<td>WRN_BALANCE_GROUP_LOCKED</td>
<td>Processing on hold as BG is locked: \textit{value}.</td>
</tr>
<tr>
<td>WRN_BALANCE_GROUP_NOT_FOUND</td>
<td>Balance group \textit{value} not found.</td>
</tr>
<tr>
<td>WRN_BALANCE_INVALID_TRANSACTION</td>
<td>Current transaction is invalid or has errors during \textit{value}.</td>
</tr>
<tr>
<td>WRN_BALANCE_INVALID_TRANSACTION_ID</td>
<td>Invalid transaction Id : \textit{value}.</td>
</tr>
<tr>
<td>WRN_BALANCE_MERGE_THREAD</td>
<td>Login \textit{value} found in multiple threads but update failed during balance merge</td>
</tr>
<tr>
<td>WRN_INVALID_CONSUMPTION_RULE</td>
<td>Invalid consumption rule \textit{value} for resourceld \textit{value} and balance group \textit{value}</td>
</tr>
</tbody>
</table>
DAT_BalanceRealtime

Table 19–5 lists the DAT_BalanceRealtime error messages.

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_BALRT_EXECUTING_OPCODE</td>
<td>Error executing the opcode: 'value'.</td>
</tr>
<tr>
<td>ERR_BALRT_GETTING_FLIST_FIELD</td>
<td>Error getting flist field: 'value'.</td>
</tr>
<tr>
<td>ERR_BALRT_MESSAGE</td>
<td>Error message for DAT_BalanceRealtime plugin module: 'value'.</td>
</tr>
</tbody>
</table>

DAT_ConnectionManager

Table 19–6 lists DAT_ConnectionManager error messages.

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_ALL_SERVER_CONNECTIONS_DOWN</td>
<td>All the server connections are down.</td>
</tr>
<tr>
<td>ERR_CREATE_LOGIN_FLIST_FAILED</td>
<td>Create Login flist failed.</td>
</tr>
<tr>
<td>ERR_INFRANET_GDD_INIT_FAILED</td>
<td>Can't initialize Infranet GDD (value)</td>
</tr>
<tr>
<td>ERR_INVALID_PROBE_VALUE</td>
<td>Incorrect probe value received for sending DPR.</td>
</tr>
<tr>
<td>ERR_LOGIN_FAILED</td>
<td>Login to CM (value) failed.</td>
</tr>
<tr>
<td>ERR_LOGIN_TO_CM</td>
<td>Login to CM failed for userid (value)</td>
</tr>
<tr>
<td>ERR_LOGOUT_TO_CM</td>
<td>Logout from CM failed for userid (value)</td>
</tr>
<tr>
<td>ERR_OPCODESCALL_FAILED</td>
<td>Opcode call failed (value).</td>
</tr>
<tr>
<td>ERR_SERVER_CONNECT_FAILURE</td>
<td>Connection to server (value) failed, strerror is (value)</td>
</tr>
<tr>
<td>ERR_SERVER_CONNECTION_LOST</td>
<td>Server connection lost (value).</td>
</tr>
<tr>
<td>ERR_SERVER_RECONNECT_FAILURE</td>
<td>Server re-connection failed (value).</td>
</tr>
<tr>
<td>ERR_CLOSE_CONNECTION_FAILED</td>
<td>Failed to close connection for socket id: (value)</td>
</tr>
</tbody>
</table>

DAT_ConnectionPool

Table 19–7 lists the DAT_ConnectionPool error messages.

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_ALL_CM_CONNECTIONS_DOWN</td>
<td>All the CM connections are down.</td>
</tr>
<tr>
<td>ERR_CM_CONNECT_FAILURE</td>
<td>Connection to CM (value) failed, strerror is (value)</td>
</tr>
<tr>
<td>ERR_CM_CONNECTION_LOST</td>
<td>CM connection lost (value).</td>
</tr>
<tr>
<td>ERR_CM_RECONNECT_FAILURE</td>
<td>CM re-connection failed (value).</td>
</tr>
<tr>
<td>ERR_CONNECT_FAILED</td>
<td>Connect call failed from CM (value).</td>
</tr>
<tr>
<td>ERR_CREATE_LOGIN_FLIST_FAILED</td>
<td>Create Login flist failed.</td>
</tr>
<tr>
<td>ERR_INFRANET_GDD_INIT_FAILED</td>
<td>Can’t initialize Infranet GDD (value)</td>
</tr>
<tr>
<td>ERR_LOGIN_FAILED</td>
<td>Login to CM (value) failed.</td>
</tr>
<tr>
<td>ERR_LOGIN_TO_CM</td>
<td>Login to CM failed for userid (value)</td>
</tr>
</tbody>
</table>
Table 19–7  (Cont.) DAT_ConnectionPool Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_LOGOUT_TO_CM</td>
<td>Logout from CM failed for userid (value)</td>
</tr>
<tr>
<td>ERR_OPCODECALL_FAILED</td>
<td>Opcode call failed (value).</td>
</tr>
<tr>
<td>ERR_SYSTEM_ERROR</td>
<td>Unexpected error (value).</td>
</tr>
</tbody>
</table>

DAT_Currency

Table 19–8 lists the DAT_Currency error message.

Table 19–8  DAT_Currency Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_REGULAR_EXP</td>
<td>Error in Regular Expression Compilation, Desc : value</td>
</tr>
</tbody>
</table>

DAT_Discount

Table 19–9 lists the DAT_Currency error message.

Table 19–9  DAT_Discount Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_ATTACH_DAT_DISCOUNT</td>
<td>Could not attach the account balance manager as 'value' to DAT_DiscountPlugIn.</td>
</tr>
<tr>
<td>ERR_DAT_DSC_GENERIC</td>
<td>FATAL ERROR 'value' line 'value' msg 'value' detail 'value'.</td>
</tr>
<tr>
<td>ERR_DB_CONNECT</td>
<td>Database connection is invalid. Possible solution is to restart DB &amp; send reconnect signal. Error: value</td>
</tr>
<tr>
<td>ERR_DETERMINE_STEP</td>
<td>Could not determine the step of related resource id: 'value'.</td>
</tr>
<tr>
<td>ERR_DISCOUNT_DUPLICATE</td>
<td>Cannot insert new discount model 'value'.</td>
</tr>
<tr>
<td>ERR_DSC_EXCLUSION_REG_SETTING</td>
<td>Error in discount exclusion registry setting</td>
</tr>
<tr>
<td>ERR_DSCMISSING_DEF</td>
<td>'value'</td>
</tr>
<tr>
<td>ERR_DSC_TIMEFRAME_DEF</td>
<td>'value'</td>
</tr>
<tr>
<td>ERR_EVENT_REGISTERED</td>
<td>Event 'value' could not be registered to DAT_Listener.</td>
</tr>
<tr>
<td>ERR_ISCRIPT_VALIDATION_FAILED</td>
<td>IScript validation failed. 'value'.</td>
</tr>
<tr>
<td>ERR_REGEXP</td>
<td>Invalid regular expression 'value'.</td>
</tr>
<tr>
<td>ERR_RELOAD_EXTDATA_FAILURE</td>
<td>Re-Init of data in Discount Functional PlugIn or Balance Data PlugIn Failed.</td>
</tr>
<tr>
<td>ERR_RELOAD_FAILURE</td>
<td>Reloading discount pricing data failed.</td>
</tr>
<tr>
<td>ERR_THRESHOLD_TO_SET_TO_MAX</td>
<td>Discount Step Threshold_To value: value is inappropriate, setting it to maximum: value.</td>
</tr>
<tr>
<td>WRN_WRONGVALUE_SET_TO_REGPARAM</td>
<td>Unexpected value set for registry parameter value.</td>
</tr>
</tbody>
</table>

DAT_ExchangeRate

Table 19–10 lists the DAT_ExchangeRate error message.
Table 19–10  DAT_ExchangeRate Error Message

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_DAT_EXCHANGERATE_INS_LIST_DB</td>
<td>Error in line ‘value’ in database table ‘value’.</td>
</tr>
</tbody>
</table>

**DAT_InterConnect**

Table 19–11 lists the DAT_InterConnect error messages.

Table 19–11  DAT_InterConnect Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_GETTING_CIBER_OCC</td>
<td>Unknown network model: value.</td>
</tr>
<tr>
<td>ERR_GETTING_NETWORK_MODEL</td>
<td>Could not get network operator for value.</td>
</tr>
<tr>
<td>ERR_GETTING_PRODUCT_GROUP</td>
<td>Could not get product group for value.</td>
</tr>
<tr>
<td>ERR_LOADING_CIBER_OCC</td>
<td>Loading IFW_CIBER_OCC failed (value).</td>
</tr>
<tr>
<td>ERR_LOADING_ICPRODUCT</td>
<td>Loading IFW_ICPRODUCT failed (value).</td>
</tr>
<tr>
<td>ERR_LOADING_ICPRODUCT_CNF</td>
<td>Loading IFW_ICPRODUCT_CNF failed (value).</td>
</tr>
<tr>
<td>ERR_LOADING_NETWORK_MODEL</td>
<td>Loading IFW_NETWORKMODEL failed (value).</td>
</tr>
<tr>
<td>ERR_LOADING_NETWORK_OPERATOR</td>
<td>Loading IFW_NETWORK_OPERATOR failed (value).</td>
</tr>
<tr>
<td>ERR_LOADING_POI</td>
<td>Loading IFW_POI failed (value).</td>
</tr>
<tr>
<td>ERR_LOADING_SWITCH</td>
<td>Loading IFW_SWITCH failed (value).</td>
</tr>
<tr>
<td>ERR_LOADING_TRUNK</td>
<td>Loading IFW_TRUNK failed (value).</td>
</tr>
<tr>
<td>ERR_LOADING_TRUNK_CNF</td>
<td>Loading IFW_TRUNK_CNF failed (value).</td>
</tr>
<tr>
<td>ERR_SETUP_ICPRODUCT_CNF_ENTRY</td>
<td>Error while setting up IFW_ICPRODUCT_CNF table from database. Reason: value.</td>
</tr>
</tbody>
</table>

**DAT_ItemAssign**

Table 19–12 lists the DAT_ItemAssign error messages.

Table 19–12  DAT_ItemAssign Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_FAILED_TO_GENERATE_MAP_TABLE</td>
<td>Failed to generate Tag and Type map table.</td>
</tr>
<tr>
<td>ERR_FAILED_TO_RESERVE_POID_IDS</td>
<td>Failed to reserve the Poid IDs.</td>
</tr>
<tr>
<td>ERR_FSM_CREATION_FAILED</td>
<td>Failed to get data from db or FSM creation failed value</td>
</tr>
<tr>
<td>ERR_INVALID_ITEM_POID_LIST</td>
<td>Item Poid List from DAT_Account is invalid.</td>
</tr>
<tr>
<td>ERR_NO_ITEM_TAG</td>
<td>Failed to get itemTag value</td>
</tr>
<tr>
<td>ERR_NO_TYPE_FOUND_FOR_TAG</td>
<td>No matching type found for given item tag.</td>
</tr>
<tr>
<td>ERR_SET_ITEM_POID_LIST_FAILED</td>
<td>Failed to set Item Poid List. value</td>
</tr>
</tbody>
</table>

**DAT_Listener**

Table 19–13 lists the DAT_Listener error messages.
Table 19–13  DAT_Listener Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_CONVERTING_FLIST</td>
<td>FLIST string cannot be converted.</td>
</tr>
<tr>
<td>ERR_CONVERTING_FLIST_TO_STR</td>
<td>Compact FLIST cannot be converted to string.</td>
</tr>
<tr>
<td>ERR_OPENING_QUEUE</td>
<td>Error: Could not open the Queue errorCode:</td>
</tr>
<tr>
<td>ERR_DEQUEUE_EVENT</td>
<td>Dequeue event exception (&quot;value&quot;).</td>
</tr>
<tr>
<td>ERR_DEQUEUE_NOT_ENABLED</td>
<td>Dequeueing for queue &quot;value&quot; is disabled.</td>
</tr>
<tr>
<td>ERR_ENQUEUE_EVENT</td>
<td>Enqueue event exception (&quot;value&quot;).</td>
</tr>
<tr>
<td>ERR_GETTING_FLIST_FIELD</td>
<td>Cannot get field &quot;value&quot; from FLIST.</td>
</tr>
<tr>
<td>ERR_OPENING_LOG_FILE</td>
<td>Fail to open the log file.</td>
</tr>
<tr>
<td>ERR_PURGE_EVENT_EXCEPT</td>
<td>Purging redundant events from queue &quot;value&quot; failed (exception = &quot;value&quot;).</td>
</tr>
<tr>
<td>ERR_PURGE_EVENT_RET</td>
<td>Purging redundant events from queue &quot;value&quot; failed (retValue = &quot;value&quot;).</td>
</tr>
<tr>
<td>ERR_QUEUE_NOT_FOUND</td>
<td>Queue &quot;value&quot; does not exist.</td>
</tr>
<tr>
<td>ERR_QUEUE_NOT_INSTALLED</td>
<td>Database queueing infrastructure has not been installed.</td>
</tr>
<tr>
<td></td>
<td>This error occurs when the DAT_Listener registry value, QueueName does not</td>
</tr>
<tr>
<td></td>
<td>exist in the table user_queues. See &quot;Installing and Configuring the Account</td>
</tr>
<tr>
<td></td>
<td>Synchronization DM&quot; in BRM Installation Guide.</td>
</tr>
<tr>
<td>ERR_RECEIVE_EVENT</td>
<td>Delivery of bus. event to DAT plugin failed (receiveEvent()).</td>
</tr>
<tr>
<td>ERR_STATVIEW_NO_ACCESS</td>
<td>Queue statistics view value cannot be accessed.</td>
</tr>
<tr>
<td>ERR_WRITING_LOG_FILE</td>
<td>Fail to write to the log file.</td>
</tr>
<tr>
<td>ERROR_REG_ENTRY_NOT_FOUND</td>
<td>Error: Registry entry not found for value.</td>
</tr>
<tr>
<td>WRN_NO_EVENTS</td>
<td>No events registered.</td>
</tr>
</tbody>
</table>

DAT_ModelSelector

Table 19–14 lists the DAT_ModelSelector error messages.

Table 19–14  DAT_ModelSelector Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_DATABASE</td>
<td>FATAL ERROR &quot;value&quot; line &quot;value&quot; msg &quot;value&quot; detail &quot;value&quot;.</td>
</tr>
<tr>
<td>ERR_DETAIL_DUPLICATE</td>
<td>Cannot insert new detail &quot;value&quot;.</td>
</tr>
<tr>
<td>ERR_DETAIL_NULL</td>
<td>Cannot find detail &quot;value&quot;.</td>
</tr>
<tr>
<td>ERR_DUPLICATE_INDEX</td>
<td>Cannot insert new index &quot;value&quot;.</td>
</tr>
<tr>
<td>ERR_ELEM_GET_NEXT</td>
<td>Error getting element from the flist. Error msg : &quot;value&quot;.</td>
</tr>
<tr>
<td>ERR_GENERIC</td>
<td>FATAL ERROR &quot;value&quot; line &quot;value&quot; msg &quot;value&quot;.</td>
</tr>
<tr>
<td>ERR_INDEX_NOT_FOUND</td>
<td>Cannot find index &quot;value&quot;.</td>
</tr>
<tr>
<td>ERR_MODEL_SELECTOR_DUPLICATE</td>
<td>Cannot insert new model selector &quot;value&quot;.</td>
</tr>
</tbody>
</table>
Table 19–15  DAT_NumberPortability Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_CLOSE_NP_FILE</td>
<td>Error closing Number Portability data file value.</td>
</tr>
<tr>
<td>ERR_NUM_PORT_RELOAD</td>
<td>Error reloading data from the Number Portability data file value.</td>
</tr>
<tr>
<td>ERR_NUM_PORT_DELTLOAD</td>
<td>Error while appending additional Number Portability data from the file value.</td>
</tr>
</tbody>
</table>

DAT_PortalConfig

Table 19–16 lists the DAT_PortalConfig error messages.

Table 19–16  DAT_PortalConfig Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_CBP_DATA_TYPE_MISMATCH</td>
<td>Data Type mismatch for param name value.</td>
</tr>
<tr>
<td>ERR_CBP_GROUP_DATA_NOT_FOUND</td>
<td>Could not find entry for group name value in Map</td>
</tr>
<tr>
<td>ERR_CBP_PARAM_DATA_NOT_FOUND</td>
<td>Could not find entry for group name value, param name value in Map</td>
</tr>
<tr>
<td>ERR_LOADING_CBP_DATA</td>
<td>Could not load CBP Data.</td>
</tr>
<tr>
<td>ERR_LOADING_OOD_DATA</td>
<td>Could not load OOD Data.</td>
</tr>
</tbody>
</table>

DAT_Price

Table 19–17 lists the DAT_Price error messages.
Table 19–17  DAT_Price Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_APPEND_CONFIG</td>
<td>Config entry could not be appended to Pricemodel-Step.</td>
</tr>
<tr>
<td>ERR_INSERTInto_MAP</td>
<td>Cannot insert entry into memory map.</td>
</tr>
<tr>
<td>ERR_INSERT_STEP</td>
<td>Cannot insert Pricemodel-Step onto RUM.</td>
</tr>
<tr>
<td>ERR_INVALID_GL_ACCOUNT</td>
<td>Current GL/Account 'value' does not match initial value 'value' for PM=value, RES=value, RUM=value.</td>
</tr>
<tr>
<td>ERR_INVALID_REVENUE_GROUP</td>
<td>Current RevenueGroup 'value' does not match initial value 'value' for PM=value, RES=value, RUM=value.</td>
</tr>
<tr>
<td>ERR_PRICE_MODEL_CONFIG_NOT_FOUND</td>
<td>Cannot find PriceModel config from the PriceModel Step object.</td>
</tr>
<tr>
<td>ERR_PRICE_MODEL_STEP_NOT_FOUND</td>
<td>Cannot find PriceModel step from the RUM object.</td>
</tr>
<tr>
<td>ERR_PRICEMODEL_NOT_FOUND</td>
<td>Cannot find PriceModel 'value' in table IFW_PRICEMODEL.</td>
</tr>
<tr>
<td>ERR_RESOURCE_LNK_NOT_FOUND</td>
<td>Cannot find the ResourceLnk object.</td>
</tr>
<tr>
<td>ERR_RESOURCE_NOT_FOUND</td>
<td>Cannot find resource 'value' in table IFW_RESOURCE.</td>
</tr>
<tr>
<td>ERR_RUM_NOT_FOUND</td>
<td>Cannot find the RUM from the ResourceLnk object.</td>
</tr>
<tr>
<td>WRN_NO_PRICEMODEL_STEPS</td>
<td>PM=value has no valid entries in IFW_PRICEMODEL_STEP configured. Skipped loading.</td>
</tr>
</tbody>
</table>

DAT_Rateplan

Table 19–18 lists the DAT_Rateplan error messages.

Table 19–18  DAT_Rateplan Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_INVALID_RUM_IN_RUMGROUP</td>
<td>Found rum group(s) in IFW_RUMGROUP with no entry in IFW_RUMGROUPS_LNK (&quot;value&quot;).</td>
</tr>
<tr>
<td>ERR_INVALID_RUM_IN_SERVICE</td>
<td>Found rum group(s) in IFW_SERVICE with no entry in IFW_RUMGROUP (&quot;value&quot;).</td>
</tr>
<tr>
<td>ERR_INVALID_SPLITTING_TYPE</td>
<td>IFW_RATEPLAN.SPLITTING has invalid value 'value'. Possible values are '0', '1', '2', '3'.</td>
</tr>
</tbody>
</table>

DAT_Recycle

Table 19–19 lists the DAT_Recycle error messages.

Table 19–19  DAT_Recycle Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_QUEUE_FILE_NOT_OPENED</td>
<td>Error opening or creating queue file value.</td>
</tr>
<tr>
<td>ERR_QUEUE_FILE_READ</td>
<td>Error reading queue file value.</td>
</tr>
<tr>
<td>ERR_QUEUE_FILE_WRITE</td>
<td>Error writing queue file value.</td>
</tr>
</tbody>
</table>
### DAT_ResubmitBatch Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_CREATE_TEMP_FILE</td>
<td>Cannot create temporary file, Error value.</td>
</tr>
<tr>
<td>ERR_ENQUEUE_DATA</td>
<td>Error enqueuing data: value.</td>
</tr>
<tr>
<td>ERR_INVALID_OPERATION</td>
<td>Operation value, value.</td>
</tr>
<tr>
<td>ERR_OPENFILE_FAILED</td>
<td>Cannot open file value, Error value.</td>
</tr>
<tr>
<td>ERR_PROCESS_RESUBMIT_JOB</td>
<td>Error occurred while processing ResubmitJob: value.</td>
</tr>
<tr>
<td>ERR_REMOVE_OLD_ITEMS</td>
<td>Error occurred while removing already Processed Items</td>
</tr>
<tr>
<td>WRN_PIPELINE_NOT_FOUND</td>
<td>Pipeline value not found for resubmitted batch value.</td>
</tr>
<tr>
<td>WRN_RENAME_FAILED</td>
<td>Cannot rename value to value, Error value.</td>
</tr>
<tr>
<td>WRN_RESUBMITINFO_NOTFOUND</td>
<td>ResubmitInfo not found for Pipeline value and File value.</td>
</tr>
</tbody>
</table>

### DAT_ScenarioReader Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_DATATYPE_MISMATCH</td>
<td>Datatype for value does not match value 'value'.</td>
</tr>
<tr>
<td>ERR_GROUPING_NOT_FOUND</td>
<td>Grouping value does not exist.</td>
</tr>
<tr>
<td>ERR_INVALID_DATATYPE</td>
<td>Datatype value is invalid.</td>
</tr>
<tr>
<td>ERR_INVALID_FLUSHMODE</td>
<td>Flushmode value is invalid.</td>
</tr>
<tr>
<td>ERR_INVALID_FUNCTION</td>
<td>Function value is invalid.</td>
</tr>
<tr>
<td>ERR_INVALID_VALUE</td>
<td>Value 'value' is invalid.</td>
</tr>
<tr>
<td>ERR_NO_CLASSITEMS</td>
<td>No classitems defined for class 'value'.</td>
</tr>
</tbody>
</table>

### DAT_USC_Map Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_FSM_ENGINE_FAILED</td>
<td>FSM Engine Failed for zone model value.</td>
</tr>
<tr>
<td>WRN_INVALID_BITVEC_MATCH_FOR_SD</td>
<td>DAT_USC_Map_ZoneModel::Pattern Matching - bitVec match error for SD (value).</td>
</tr>
<tr>
<td>WRN_INVALID_PATH_NUM_FOR_SD</td>
<td>DAT_USC_Map_ZoneModel::Pattern Matching - invalid path number for SD (value).</td>
</tr>
<tr>
<td>WRN_INVALID_QTY_VAL</td>
<td>DAT_USC_Map_ZoneModel::Invalid quantity value for entry (value).</td>
</tr>
<tr>
<td>WRN_INVALID_TIME_FRAME</td>
<td>DAT_USC_Map_ZoneModel::Invalid timeframe for entry (value).</td>
</tr>
<tr>
<td>WRN_NO_USC_ENTRY_FOR_PATTERN</td>
<td>DAT_USC_Map_ZoneModel::No usc-entries found during pattern matching.</td>
</tr>
</tbody>
</table>
Table 19–22  (Cont.) DAT_USC_Map Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WRN_NO_USC_ENTRY_FOR_PATTERN_AND_SD</td>
<td>DAT_USC_Map_ZoneModel::Pattern Matching - no usc-entries found for SD (value).</td>
</tr>
<tr>
<td>WRN_NO_USC_MAP_ENTRY_FOR_ZONE_MODEL</td>
<td>DAT::USC_Map::No Usc Entries found for zone model ID (value).</td>
</tr>
<tr>
<td>WRN_NO_VALID_USC_ENTRY</td>
<td>DAT_USC_Map_ZoneModel::No Valid USC entry mapping found.</td>
</tr>
</tbody>
</table>

**DAT_Zone**

Table 19–23 lists the DAT_Zone error messages.

Table 19–23  DAT_Zone Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_INSERT_ZONEMODEL</td>
<td>Error inserting zone model into configuration.</td>
</tr>
<tr>
<td>ERR_INV_ZONECONFIG_LINE</td>
<td>Error invalid zone config line: value.</td>
</tr>
<tr>
<td>ERR_INV_ZONECONFIG_ROW</td>
<td>Error invalid values in INT_STANDARD_ZONE: ZONEMODEL=value, ORIGIN_AREACODE=value, DESTIN_AREACODE=value, SERVICECODE=value, VALID_FROM=value, VALID_TO=value, ZONE_WS=value, ZONE_RT=value, ALT_ZONEMODEL=value.</td>
</tr>
<tr>
<td>ERR_INVALID_BEAT</td>
<td>Error: Invalid value for Beat (must be greater than 0).</td>
</tr>
<tr>
<td>ERR_ZONEMODEL_NOT_IN_CONFIG</td>
<td>Cannot find zone model 'value' in configuration data.</td>
</tr>
<tr>
<td>WRN_INVALID_AREACODE</td>
<td>AreaCode contains non-digit characters. Error=value.</td>
</tr>
<tr>
<td>WRN_DUPLICATE_SERVICETYPE</td>
<td>Duplicate entry for ServiceType=value corresponding to ServiceCode=value.</td>
</tr>
</tbody>
</table>

**EXT_InEasyDB**

Table 19–24 lists the EXT_InEasyDB error messages.

Table 19–24  EXT_InEasyDB Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_ERROR_FILE_NOT_EXIST</td>
<td>Jobfile ‘value’ doesn’t exist for rollback.</td>
</tr>
<tr>
<td>ERR_READING_DATA_FROM_DATABASE</td>
<td>Error reading data from database -&gt; do data from database and no eof.</td>
</tr>
<tr>
<td>WRN_FILE_NOT_MOVED</td>
<td>Jobfile couldn’t moved to actual temp-file.</td>
</tr>
</tbody>
</table>

**EXT_PipelineDispatcher**

Table 19–25 lists the EXT_PipelineDispatcher error messages.

Table 19–25  EXT_PipelineDispatcher Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_WRONG_INFILEMGR</td>
<td>Input file manager ‘value’ is not of type EXT_InFileManager</td>
</tr>
<tr>
<td>ERR_RENAME_FILE_FAILED</td>
<td>Failed to rename file ‘value’</td>
</tr>
</tbody>
</table>
### FCT_Account

Table 19–26 lists the FCT_Account error messages.

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_EMPTY_SERVICE_FIELD_MAP</td>
<td>No service -&gt; edr field mapping entries for pipeline ‘value’ in alias map.</td>
</tr>
<tr>
<td>ERR_JOB_RERATING_ACCOUNT</td>
<td>This Account is currently being Rerated.</td>
</tr>
</tbody>
</table>

### FCT_AccountRouter

Table 19–27 lists the FCT_AccountRouter error messages.

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_DATA_MODULE_IS_NOT_A_ROUTER</td>
<td>Data Module (value) is not configured as a Router.</td>
</tr>
<tr>
<td>ERR_DB_ROUTING_FAILED</td>
<td>Splitting failed (value).</td>
</tr>
<tr>
<td>ERR_JOB_AMT_MIGRATION</td>
<td>Job is under migration state and being directed.</td>
</tr>
<tr>
<td>ERR_REGISTRY_KEY_ERROR</td>
<td>Error found in registry key value pair (value).</td>
</tr>
</tbody>
</table>

### FCT_AggreGate

Table 19–28 lists the FCT_AggreGate error messages.

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_CTLFILE_NOT_CREATED</td>
<td>Control File value could not be created. Reason : value</td>
</tr>
<tr>
<td>ERR_DATABLOCK_NOT_FOUND</td>
<td>EDR datablock ‘value’ not found.</td>
</tr>
<tr>
<td>ERR_EDR_ITERATOR_FAILURE</td>
<td>EDR indexes mismatch: value.</td>
</tr>
<tr>
<td>ERR_NO_DEPENDENT_CLASS_DEFINED</td>
<td>No dependent class defined for class ‘value’ and classitem ‘value’.</td>
</tr>
<tr>
<td>ERR_SCENARIO_NOT_DEFINED</td>
<td>Scenario ‘value’ not defined.</td>
</tr>
<tr>
<td>WRN_NO_SCENARIOS_CONFIGURED</td>
<td>No scenarios configured.</td>
</tr>
<tr>
<td>WRN_SCENARIO_NOT_ACTIVE</td>
<td>Scenario ‘value’ is not active.</td>
</tr>
</tbody>
</table>

### FCT_APN_Map

Table 19–29 lists the FCT_APN_Map error messages.

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_GPRS_GSMW_AMBIGUITY</td>
<td>GPRS and GSMW extensions present. This is ambiguous. (value).</td>
</tr>
<tr>
<td>ERR_INIT_APN_MAPTABLE</td>
<td>Initialise of map table failed (value).</td>
</tr>
<tr>
<td>ERR_INIT_EDR_ITERATOR_CHARGE_PACKET</td>
<td>Failed to initialise charge packet iterator (value).</td>
</tr>
<tr>
<td>ERR_INIT_EDR_ITERATOR_ZONE_PACKET</td>
<td>Failed to initialise zone packet iterator (value).</td>
</tr>
</tbody>
</table>
Table 19–29  (Cont.) FCT_APN_Map Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_RAZ_MAP_NOT_USABLE</td>
<td>Run after zoning map table not usable (value).</td>
</tr>
<tr>
<td>ERR_RAZ_MAP_TABLE_NOT_INITIALISED</td>
<td>Run after zoning map table not initialised (value).</td>
</tr>
<tr>
<td>ERR_RBZ_MAP_TABLE_INVALID</td>
<td>Run before zoning map table not usable (value).</td>
</tr>
<tr>
<td>ERR_RBZ_MAP_TABLE_NOT_INITIALISED</td>
<td>Run before zoning map table not initialised (value).</td>
</tr>
</tbody>
</table>

**FCT_ApplyBalance**

Table 19–30 lists the FCT_ApplyBalance error messages.

Table 19–30  FCT_ApplyBalance Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_APPLYBAL_CANCEL_DEMANDED</td>
<td>EDR belongs to a cancel demanded transaction.</td>
</tr>
<tr>
<td>ERR_APPLYBAL_CANCEL_EDR</td>
<td>EDR transaction cancel demanded.</td>
</tr>
<tr>
<td>ERR_APPLYBAL_EDR_ITERATOR</td>
<td>Could not reset EDR iterator.</td>
</tr>
<tr>
<td>ERR_APPLYBAL_NO_ACCOUNT_BALANCE</td>
<td>Could not create/find balance for BG/Resource: value.</td>
</tr>
<tr>
<td>ERR_APPLYBAL_NO_PREFIX</td>
<td>No prefix specified for the notification file name</td>
</tr>
<tr>
<td>ERR_APPLYBAL_PLUGIN_INVALID_STATE</td>
<td>Required action does not fit to current state 'value'.</td>
</tr>
<tr>
<td>ERR_APPLYBAL_REALTIME</td>
<td>Apply Balance plugin not required for Realtime mode.</td>
</tr>
<tr>
<td>ERR_APPLYBAL_ROLLBACK_EDR</td>
<td>EDR transaction rollback demanded</td>
</tr>
<tr>
<td>ERR_NO_SUBBALIMPACT</td>
<td>No sub-balance impact created for BG/Resource 'value', discount not granted for this EDR. Make sure the resource is valid and there is a check for available resource in the configuration.</td>
</tr>
</tbody>
</table>

**FCT_BatchSuspense**

Table 19–31 lists the FCT_BatchSuspense error messages.

Table 19–31  FCT_BatchSuspense Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_NO_DEFAULT_BATCH_SUSPENSE_REASON</td>
<td>No default batch suspense reason.</td>
</tr>
<tr>
<td>ERR_INVALID_RESUBMIT_INFO</td>
<td>Invalid resubmit information received for batch name value and pipeline value.</td>
</tr>
<tr>
<td>ERR_NO_BATCH_SUSPENSE_REASON</td>
<td>No batch suspense reason in the /config/suspense_reason_code object.</td>
</tr>
</tbody>
</table>

**FCT_BillingRecord**

Table 19–32 lists the FCT_BillingRecord error messages.

Table 19–32  FCT_BillingRecord Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_BALANCE_NOT_FOUND</td>
<td>Error adding discount balance info: value.</td>
</tr>
</tbody>
</table>
**FCT_CallAssembling**

Table 19–33 lists the FCT_CallAssembling error messages.

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_BAD_EDR_FILE_STREAM</td>
<td>std::fstream operation has failed for file: <em>value</em></td>
</tr>
<tr>
<td>ERR_CALL_MISSING_FROM_DELETION_VECTOR</td>
<td>The rejected call <em>value</em> is missing from the CallDeletionVector.</td>
</tr>
<tr>
<td>ERR_CANNOT_CLEANUP_EDR_FILE</td>
<td>Cannot remove file: <em>value</em> during EDRFileManager::cleanupEDRFiles() operation.</td>
</tr>
<tr>
<td>ERR_CANNOT_CREATE_INDEX_FILE_STREAM</td>
<td>Unable to create the EDR index std::fstream for file: <em>value</em></td>
</tr>
<tr>
<td>ERR_CANNOT_FIND_DEFAULT_OUTPUT_STREAM</td>
<td>Cannot lookup the defaultOutputStream during an UpgradeFlushLimit semaphore. Attempted location: <em>value</em></td>
</tr>
<tr>
<td>ERR_CANNOT_OPEN_INDEX_FILE_STREAM</td>
<td>Unable to open the EDR index std::fstream for file: <em>value</em></td>
</tr>
<tr>
<td>ERR_CHAIN_REFERENCE_MISSING</td>
<td>Chain reference is missing.</td>
</tr>
<tr>
<td>ERR_CREATING_CONTAINER_INDEX</td>
<td>Unable to create DETAIL container index from EDRFactory.</td>
</tr>
<tr>
<td>ERR_DELETION_MISSING_CALL_RECORD</td>
<td>Error, expected CallRecord missing from map during CallDeletionVector cleanup. <em>value</em></td>
</tr>
<tr>
<td>ERR_DOM_EXCEPTION</td>
<td>DOMException caught in <em>value</em>: Message=<em>value</em></td>
</tr>
<tr>
<td>ERR_EDR_ALREADY_CLOSED</td>
<td>The edr with the following chain reference is already closed: Chain reference= <em>value</em>, LongDurationIndicator= <em>value</em>, StartTimestamp= <em>value</em></td>
</tr>
<tr>
<td>ERR_EDR_FILE_DOESNT_EXIST</td>
<td>EDRFile::initialize() failed because file: <em>value</em> does not exist.</td>
</tr>
<tr>
<td>ERR_EDR_FILE_NOT_INDEXED</td>
<td>The following file: <em>value</em> is referenced, but not available in the EDR file index.</td>
</tr>
<tr>
<td>ERR_EMPTY_MESSAGE</td>
<td>Reject Message <em>value</em> is missing arguments.</td>
</tr>
<tr>
<td>ERR_F_SEGMENT_ALREADY_RECEIVED</td>
<td>Error : An F segment has been already received with the same chain reference</td>
</tr>
<tr>
<td>ERR_FLUSH_LIMIT_IN_PROGRESS</td>
<td>Error: semaphore FlushLimit=<em>value</em> is already in progress, FlushLimit must complete before sending UpgradeFlushLimit.</td>
</tr>
<tr>
<td>ERR_INCLUSIVE_FLUSHZONE_LOGIC</td>
<td>Error InclusiveLogic, Errant Flushzone for ChainReference <em>value</em>.</td>
</tr>
<tr>
<td>ERR_INCLUSIVE_LOGIC_FLUSHZONE_OUT_OF_POSITION</td>
<td>Error InclusiveLogic, Flushzone is out-of-position.</td>
</tr>
<tr>
<td>ERR_INCLUSIVE_LOGIC_TOO_MANY_ACTIVE_CALL_SECTIONS</td>
<td>Error InclusiveLogic, SingleElementCallSection has too many ACTIVE CallSections.</td>
</tr>
<tr>
<td>ERR_INDEX_FILE_RENAME_FAILED</td>
<td>Unable to rename tmp to index file: <em>value</em> during IndexFile::commitFile() operation.</td>
</tr>
<tr>
<td>ERR_INVALID_CHAIN_REFERENCE</td>
<td>Could not find data for chain reference <em>value</em>.</td>
</tr>
<tr>
<td>ERR_INVALID_STATE_INDICATOR</td>
<td>State <em>value</em> is unexpected.</td>
</tr>
<tr>
<td>ERR_INVALID_TRANSID_IN_REJECT_MSG</td>
<td>The transaction id sent by FCT_Reject to FCT_CallAssembling is invalid: <em>value</em>.</td>
</tr>
</tbody>
</table>
### Table 19–33 (Cont.) FCT_CallAssembling Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_L_SEGMENT_ALREADY_RECEIVED</td>
<td>Error: An L segment has been already received with the same chain reference</td>
</tr>
<tr>
<td>ERR_LATEPARTIAL_EDR</td>
<td>Late EDR received and marked invalid; chain reference = value.</td>
</tr>
<tr>
<td>ERR_MISUSE_VIRTUAL_FUNCTION</td>
<td>Error, virtual function 'value' is not allowed.</td>
</tr>
<tr>
<td>ERR_MULTI_CALL_SECTION_MISSING_ELEMENTS</td>
<td>The MultiDataElementCallSection is missing DataElements.</td>
</tr>
<tr>
<td>ERR_NO_CHAIN_REFERENCE</td>
<td>Chain reference is missing in message value.</td>
</tr>
<tr>
<td>ERR_NOT_CA_WORKFILE</td>
<td>File: value is not a Call Assembling workfile.</td>
</tr>
<tr>
<td>ERR_PRODUCING_DEFAULT_EDR</td>
<td>Error: cannot produce default EDR for container 'value' during a flush operation.</td>
</tr>
<tr>
<td>ERR_RECYCLED_CALL_RECORD_MISSING</td>
<td>Error, CallRecord missing for recycled call, 'value'.</td>
</tr>
<tr>
<td>ERR_RECYCLED_EDR_NOT_FOUND_IN_MAP</td>
<td>Error, RecycleRequest failed for chain ref 'value'.</td>
</tr>
<tr>
<td>ERR_REJECT_CALL_RECORD_MISSING</td>
<td>Error, finding CallRecord during FCT_Reject AssemblyLogic lookup request. 'value'</td>
</tr>
<tr>
<td>ERR_REJECT_CALL_SECTION_MISSING</td>
<td>Error, finding CallSection during FCT_Reject notification request. ChainRef='value', StartTime='value'</td>
</tr>
<tr>
<td>ERR_REJECTED_EDR_NOT_IN_WORKFILE</td>
<td>The rejected edr is no longer in workfile. Chain reference= 'value', LongDurationIndicator= 'value', StartTimestamp= 'value'</td>
</tr>
<tr>
<td>ERR_RESTORE_ASSEMBLY_LOGIC_FAILED</td>
<td>Error, unable to restore AssemblyLogic in CallRecord::restore()</td>
</tr>
<tr>
<td>ERR_SAX_EXCEPTION</td>
<td>SAXException caught in value: Message=value</td>
</tr>
<tr>
<td>ERR_SPURIOUS_MESSAGE</td>
<td>Ignoring spurious message value.</td>
</tr>
<tr>
<td>ERR_UNUSED_EDRS_IN_PROCESS_RESULT</td>
<td>Unable to serialize an index item to the EDR index file named: 'value'</td>
</tr>
<tr>
<td>ERR_UNKNOWN_WORKFILE_VERSION</td>
<td>Cannot read workfile: value - because of unknown version: value</td>
</tr>
<tr>
<td>ERR_UNUSED_EDRS_IN_PROCESS_RESULT</td>
<td>Error, un-used edrs remaining in the destructed ProcessResult object.</td>
</tr>
<tr>
<td>ERR_UPGRADE_FLUSH_LIMIT_IN_PROGRESS</td>
<td>Error: semaphore UpgradeFlushLimit=value is already in progress, UpgradeFlushLimit must complete before sending FlushLimit.</td>
</tr>
<tr>
<td>ERR_UPGRADE_MODE_FAILURE</td>
<td>CallAssembly Error while Upgrading EDR with ChainRef=value and StartTime=value</td>
</tr>
<tr>
<td>ERR_VALID_DETAIL_DURING_FLUSH</td>
<td>During flush operation, the restored EDR with CHAINREFERENCE==value does not pass the isValidDetail() test.</td>
</tr>
<tr>
<td>ERR_XML_EXCEPTION</td>
<td>XMLException caught in value: Message=value</td>
</tr>
<tr>
<td>ERR_XML_IMPORT_FILE_MISSING</td>
<td>Unable to process ImportDataFromXml semaphore because supplied XML file 'value' is missing.</td>
</tr>
<tr>
<td>ERR_XML_MEMORY_EXCEPTION</td>
<td>Xerces OutOfMemoryException caught in value: Message=value</td>
</tr>
</tbody>
</table>
**FCT_CarrierIcRating**

Table 19–34 lists the FCT_CarrierIcRating error messages.

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_ICPRODUCT_INVALID</td>
<td>No valid entry in IFW_ICPRODUCT_RATE found for NM=value, NO=value, ICPRODUCT=value and Date=value.</td>
</tr>
<tr>
<td>ERR_ICPRODUCT_NETWORKMODEL_NOT_FOUND</td>
<td>Network model ‘value’ not found.</td>
</tr>
<tr>
<td>ERR_ICPRODUCT_NOT_FOUND</td>
<td>ICPRODUCT for GROUP/TR+DIR=value, DATE=value, SNW=value, DNW=value, A#=value, B#=value, C#=value, RecT=value, SCODE=value, SCLASS=value and UC=value not found (Reason ‘value’).</td>
</tr>
<tr>
<td>ERR_INVALID_MODELTYPE</td>
<td>value</td>
</tr>
<tr>
<td>ERR_TRUNK_NOT_FOUND</td>
<td>No entry found for trunk under GSM extension.</td>
</tr>
</tbody>
</table>

**FCT_CiberOcc**

Table 19–35 lists the FCT_CiberOcc error messages.

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_CIBEROCC_NETWORKMODEL_NOT_FOUND</td>
<td>The network model declared by registry parameter EdrNetworkModel cannot be found in the data module.</td>
</tr>
<tr>
<td>ERR_CIBEROCC_NOT_FOUND</td>
<td>A valid entry cannot be found for the source network (NM=value, SN=value, and DATE=value) in IFW_CIBER_OCC.</td>
</tr>
<tr>
<td>ERR_CREATE_OCC_EDR</td>
<td>An OCC EDR container cannot be created.</td>
</tr>
</tbody>
</table>

**FCT_CliMapping**

Table 19–36 lists the FCT_CliMapping error messages.

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_EDR_FACTORY</td>
<td>Error failed to get the edr factory (value).</td>
</tr>
<tr>
<td>WRN_CLIENTRY_INVALID</td>
<td>Setup cli Map entry failed (value).</td>
</tr>
</tbody>
</table>

**FCT_CreditLimitCheck**

Table 19–37 lists the FCT_CreditLimitCheck error messages.

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WRN_NO_BG_OBJECT</td>
<td>No Balance Group Object found</td>
</tr>
</tbody>
</table>

**FCT_CustomerRating**

Table 19–38 lists the FCT_CustomerRating error messages.
Table 19–38  FCT_CustomerRating Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_CUSTOMER_NOT_FOUND</td>
<td>No customer datablock found.</td>
</tr>
<tr>
<td>ERR_INIT_SLA_TABLE</td>
<td>Error during initialisation from IFW_SLA table. See pipeline log for additional information.</td>
</tr>
<tr>
<td>ERR_RATEPLAN_NOT_DEFINED</td>
<td>No rateplan defined for customer account ‘value’.</td>
</tr>
<tr>
<td>WRN_CUSTOMER_NOT_FOUND</td>
<td>No customer datablock found. Using Default-Rateplan ‘value’ to continue.</td>
</tr>
</tbody>
</table>

FCT_DataDump

Table 19–39 lists the FCT_DataDump error messages.

Table 19–39  FCT_DataDump Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_INSERT_EVENT</td>
<td>Failed to insert event ‘value’</td>
</tr>
</tbody>
</table>

FCT_Discard

Table 19–40 lists the FCT_Discard error messages

Table 19–40  FCT_Discard Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WRN_FIELD_NOT_FOUND</td>
<td>EDR field ‘value’ does not exist.</td>
</tr>
</tbody>
</table>

FCT_Discount

Table 19–41 lists the FCT_Discount error messages.

Table 19–41  FCT_Discount Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_ACCOUNT_CANCEL</td>
<td>Could not cancel transaction ‘value’.</td>
</tr>
<tr>
<td>ERR_ACCOUNT_COMMIT</td>
<td>Could not commit transaction ‘value’.</td>
</tr>
<tr>
<td>ERR_ACCOUNT_COMMIT_RESTART</td>
<td>Could not commit transaction ‘value’ on restart.</td>
</tr>
<tr>
<td>ERR_ACCOUNT_PREPARECOMMIT</td>
<td>Could not prepare commit transaction ‘value’.</td>
</tr>
<tr>
<td>ERR_ACCOUNT.Rollback</td>
<td>Could not rollback transaction ‘value’.</td>
</tr>
<tr>
<td>ERR_BEGIN_DSC_TRANSACTION</td>
<td>Cannot start the transaction ‘value’.</td>
</tr>
<tr>
<td>ERR_BEGIN_EDR</td>
<td>Cannot begin EDR transaction.</td>
</tr>
<tr>
<td>ERR_CANCEL_DEMANDED_EDR</td>
<td>EDR belongs to a cancel demanded transaction.</td>
</tr>
<tr>
<td>ERR_CANCEL_EDR</td>
<td>EDR transaction cancel demanded.</td>
</tr>
<tr>
<td>ERR_CANNOT_COMPILE_SCRIPT</td>
<td>Failed to compile the following IScript: ‘value’.</td>
</tr>
<tr>
<td>ERR_COMMIT_EDR</td>
<td>Cannot commit EDR transaction.</td>
</tr>
<tr>
<td>ERR_CURRENCY_RESID_NOT_FOUND</td>
<td>Failed to find this currency from DAT::Currency: ‘value’.</td>
</tr>
<tr>
<td>ERR_DISCOUNT_DETACH_MODULE</td>
<td>Could not detach pipeline ‘value’ (not attached).</td>
</tr>
</tbody>
</table>
| ERR_DSC_CONF_NOT_FOUND            | Cannot find configuration for discount model ‘value’, date ‘value’.
Table 19–41 (Cont.) FCT_Discoun Message

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_EDR_ITERATOR</td>
<td>Could not reset EDR iterator.</td>
</tr>
<tr>
<td>ERR_EDRPACK_NOT_READY_DSC</td>
<td>Not all EDR fields needed for discounting are filled.</td>
</tr>
<tr>
<td>ERR_END_DSC_TRANSACTION</td>
<td>Cannot start the transaction ‘value’.</td>
</tr>
<tr>
<td>ERR_EXPR_REF_CP</td>
<td>Expression referencing Charge Packet amount when there is no Charge Packet: ‘value’.</td>
</tr>
<tr>
<td>ERR_GETTING_BG_ID</td>
<td>Failed to get balance group id for account: ‘value’.</td>
</tr>
<tr>
<td>ERR_INVALID_BASE_AMOUNT</td>
<td>Discount with no Charge Packet (Billing Time discount) cannot reference CP amount: ‘value’.</td>
</tr>
<tr>
<td>ERR_INVALID_BASE_EXPR</td>
<td>Invalid base amount value. Expression: ‘value’.</td>
</tr>
<tr>
<td>ERR_INVALID_COND_AMOUNT</td>
<td>Invalid condition amount value. Expression: ‘value’.</td>
</tr>
<tr>
<td>ERR_INVALID_DISCOUNT_TYPE</td>
<td>Invalid discount type ‘value’.</td>
</tr>
<tr>
<td>ERR_INVALID_DRUM_AMOUNT</td>
<td>Invalid DRUM amount value. Expression: ‘value’.</td>
</tr>
<tr>
<td>ERR_INVALID_GRANT_TYPE</td>
<td>The grant type ‘value’ is invalid.</td>
</tr>
<tr>
<td>ERR_INVALID_THRESHOLD_AMOUNT</td>
<td>Invalid threshold_to amount value. Expression: ‘value’</td>
</tr>
<tr>
<td>ERR_INVALID_THRESHOLD_TYPE</td>
<td>Invalid threshold type ‘value’.</td>
</tr>
<tr>
<td>ERR_NO_ACCOUNT_BALANCE</td>
<td>Could not create/find balance for BG/Resource: ‘value’.</td>
</tr>
<tr>
<td>ERR_PLUGIN_INVALID_STATE</td>
<td>Required action does not fit to current state ‘value’.</td>
</tr>
<tr>
<td>ERR_REJECT_EDR</td>
<td>EDR rejection demanded.</td>
</tr>
<tr>
<td>ERR_ROLLBACK_EDR</td>
<td>EDR transaction rollback demanded.</td>
</tr>
</tbody>
</table>

FCT_DroppedCall

Table 19–42 lists the FCT_DroppedCall error messages.

Table 19–42 FCT_DroppedCall Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_DATA_TYPE_MISMATCH</td>
<td>Datatype of ContinuationCallField = valuedoes not match the datatype of DroppedCallField = value</td>
</tr>
<tr>
<td>ERR_DROPPED_CALL_UPDATION_FAILED</td>
<td>Updation of Dropped Call entry in the memory map failed : value</td>
</tr>
<tr>
<td>ERR_FILE_ID_NOT_FOUND</td>
<td>File ID Not Found</td>
</tr>
<tr>
<td>ERR_FILE_FORMAT_INcorrect</td>
<td>File Format is not correct</td>
</tr>
<tr>
<td>ERR_INSERT_INTO_MAP_FAILED</td>
<td>Insertion into the DroppedCallInfoMap failed : value</td>
</tr>
<tr>
<td>ERR_REMOVING_FILE</td>
<td>Could not remove file value</td>
</tr>
<tr>
<td>ERR_RESTORE_FAILED</td>
<td>Restore of the file failed</td>
</tr>
<tr>
<td>ERR_SERIALIZE_FAILED</td>
<td>Serialize to the file failed</td>
</tr>
<tr>
<td>ERR_UNABLE_TO_RETRIEVE_XML_ID</td>
<td>Unable to retrieve XML Id</td>
</tr>
</tbody>
</table>
### Table 19–42 (Cont.) FCT_DroppedCall Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_VALUE_FROM_XML_ID</td>
<td>Unable to retrieve the value from XML Id</td>
</tr>
<tr>
<td>ERR_VALUE_PROFILE_ERA</td>
<td>Invalid value of DROPPED_CALL Profile ERA value = value. Profile ERA value is not set, So the default behavior is assumed.</td>
</tr>
<tr>
<td>WRN_FILE_NOT_FOUND</td>
<td>Main data file not found.</td>
</tr>
</tbody>
</table>

### FCT_DuplicateCheck

Table 19–43 lists the FCT_DuplicateCheck error messages.

### Table 19–43 FCT_DuplicateCheck Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_BULKINSERTION_FAILED</td>
<td>Bulk Inserion Failed in the table value from file value. Error : value.</td>
</tr>
<tr>
<td>ERR_DUP_IND_FIELD_TYPE</td>
<td>Wrong duplicate indicator field type (requires INTEGER).</td>
</tr>
<tr>
<td>ERR_FLUSH_TO_FILE</td>
<td>Could not flush data to file.</td>
</tr>
<tr>
<td>ERR_INSERT_MAP_FAILED</td>
<td>Insertion into the DupCheck Map failed from File : value.</td>
</tr>
<tr>
<td>ERR_INSERTION_FAILED</td>
<td>Insertion in the table value failed . Error: value.</td>
</tr>
<tr>
<td>ERR_NO_INDEXSPACE_CONF</td>
<td>IndexSpaceName entry is not specified in the FCT_DuplicateCheck module in the registry. It is mandatory if the Database mode is configured.</td>
</tr>
<tr>
<td>ERR_NO_TABLESPACE_CONF</td>
<td>TableSpaceName entry is not specified in the FCT_DuplicateCheck module in the registry. It is mandatory if the Database mode is configured.</td>
</tr>
<tr>
<td>ERR_PROC_EXEC_FAILED</td>
<td>Procedure: value execute failed , value.</td>
</tr>
<tr>
<td>ERR_PROC_MISSING</td>
<td>Procedure: value does not exist.</td>
</tr>
<tr>
<td>ERR_REMOVE_FILE</td>
<td>Could not remove file value.</td>
</tr>
<tr>
<td>WRN_DUP_RECORD</td>
<td>Duplicate record found in the file.</td>
</tr>
<tr>
<td>WRN_EMPTY_FIELD</td>
<td>One key field is empty, field ignored.</td>
</tr>
<tr>
<td>WRN_MAIN_FILE_NOT_FOUND</td>
<td>Main data file not found.</td>
</tr>
<tr>
<td>WRN_NO_ROOT_FIELD</td>
<td>Defined field is not in root block.</td>
</tr>
</tbody>
</table>

### FCT_EnhancedSplitting

Table 19–44 lists the FCT_EnhancedSplitting error messages.

### Table 19–44 FCT_EnhancedSplitting Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_ADD_SPLITTING_PATTERNS</td>
<td>Failed to add patterns to fsm (value): value.</td>
</tr>
<tr>
<td>ERR_INSERT_SPECSYS</td>
<td>Failed to insert ‘value=value’ into the mapping table.</td>
</tr>
</tbody>
</table>
Table 19–44 (Cont.) FCT_EnhancedSplitting Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_NO_SPLITTING_ENTRY</td>
<td>No matching splitting rule found.</td>
</tr>
<tr>
<td>ERR_NO_STREAM_FOR_SPECYS</td>
<td>No output stream for specialist system ‘value’.</td>
</tr>
<tr>
<td>ERR_SETUP_SPLITTING_ENTRY</td>
<td>Setup for splitting entry (value) failed: value.</td>
</tr>
</tbody>
</table>

FCT_ExchangeRate

Table 19–45 lists the FCT_ExchangeRate error messages.

Table 19–45 FCT_ExchangeRate Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_EXCHANGERATE_BRK_HEADERDATE</td>
<td>File creation date is invalid: ‘value’.</td>
</tr>
<tr>
<td>ERR_EXCHANGERATE_FILEDATE_NOT_EXIST</td>
<td>File date doesn’t exist: ‘value’.</td>
</tr>
</tbody>
</table>

FCT_Filter_Set

Table 19–46 lists the FCT_Filter_Set error messages.

Table 19–46 FCT_Filter_Set Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_INVALID_DISCOUNT_ID</td>
<td>Invalid discount object id ‘value’.</td>
</tr>
<tr>
<td>ERR_INVALID_PRODUCT_ID</td>
<td>Invalid product id ‘value’.</td>
</tr>
</tbody>
</table>

FCT_IRules

Table 19–47 lists the FCT_IRules error messages.

Table 19–47 FCT_IRules Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_ILLEGAL_CUSTOMER_KEY</td>
<td>Illegal customer key ‘value’.</td>
</tr>
<tr>
<td>ERR_ILLEGAL_LICENSE_KEY</td>
<td>Illegal license key ‘value’.</td>
</tr>
<tr>
<td>ERR_RULE_DESCRIPTION</td>
<td>Failed to create ruleset from description: value.</td>
</tr>
<tr>
<td>ERR_RULE_SETUP</td>
<td>Failed to setup rule ‘value’: value.</td>
</tr>
<tr>
<td>ERR_RULESET_FILE</td>
<td>value:line value: value.</td>
</tr>
<tr>
<td>ERR_RULESET_NOT_FOUND</td>
<td>Ruleset ‘value’ not found.</td>
</tr>
</tbody>
</table>

FCT_IScriptPlugIn

Table 19–48 lists the FCT_IScriptPlugIn error messages

Table 19–48 FCT_IScriptPlugIn Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_SCRIPT_NOT_USABLE</td>
<td>The iScript ‘value’ is not usable.</td>
</tr>
</tbody>
</table>
Table 19–49 FCT_ItemAssign Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_INVALID_ITEM_POID</td>
<td>Invalid item poid returned from DAT_ItemAssign.</td>
</tr>
</tbody>
</table>

Table 19–50 FCT_MainRating Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_INVALID_ADDON_TYPE</td>
<td>Addon-Type in IFW_RATEPLAN_CNF for RP=value, RP-V=value, IC=value, SCode=value, SClass=value, TM=value and TZ=value is invalid. Not in 'P'ercentage, 'A'bsolute or 'N'ew value.</td>
</tr>
<tr>
<td>ERR_PRICEMODEL_CONFIG_NOT_FOUND</td>
<td>Pricemodel-Config not found for PM=value, RES=value, RUM=value, Step=value, Frame=value and Date=value.</td>
</tr>
<tr>
<td>ERR_PRICEMODEL_NOT_FOUND</td>
<td>Pricemodel 'value' not found. See process-log-file for invalid pricemodels not loaded. Packet no. 'value'.</td>
</tr>
<tr>
<td>ERR_PRICEMODEL_RUM_NOT_FOUND</td>
<td>Pricemodel-RUM not found for PM=value, RES=value and RUM=value. Packet no. 'value'.</td>
</tr>
<tr>
<td>ERR_PRICEMODEL_STEP_NOT_FOUND</td>
<td>Pricemodel-Step not found for PM=value, RES=value and RUM=value. Packet no. 'value'.</td>
</tr>
<tr>
<td>ERR_RATE_PRICEMODEL_NOT_FOUND</td>
<td>Pricemodel not found (IFW_RATEPLAN_CNF) for RP=value, RP-V=value, IC=value, SCode=value, SClass=value, TM=value and TZ=value. Packet no. 'value'.</td>
</tr>
<tr>
<td>ERR_RATEPLAN_VERSION_DATE_NOT_FOUND</td>
<td>Rateplan-Version not found (IFW_RATEPLAN_VER) for Rateplan=value and Date=value. Packet no. 'value'.</td>
</tr>
<tr>
<td>ERR_RATEPLAN_VERSION_ID_NOT_FOUND</td>
<td>Rateplan-Version not found (IFW_RATEPLAN_VER) for Rateplan=value and Version=value. Packet no. 'value'.</td>
</tr>
<tr>
<td>ERR_RUM_GROUP_NOT_FOUND</td>
<td>Found no valid rum group for packet no. 'value'.</td>
</tr>
<tr>
<td>ERR_TIMEMODEL_NOT_FOUND</td>
<td>Timemodel not found (IFW_RATEPLAN_CNF) for RP=value, RP-V=value, IC=value, SCode=value and SClass=value. Packet no. 'value'.</td>
</tr>
<tr>
<td>ERR_TIMEZONE_NOT_FOUND</td>
<td>Timezone not found in TimeModel=value for Date=value. Packet no. 'value'.</td>
</tr>
</tbody>
</table>

Table 19–51 FCT_Opcode Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_ALL_CM_CONNECTIONS_LOST</td>
<td>All CM connections lost.</td>
</tr>
<tr>
<td>ERR_RECV_DATA</td>
<td>Error while receiving data from CM (value)</td>
</tr>
</tbody>
</table>
Table 19–51  (Cont.) FCT_Opcode Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_RESTORE_DATA</td>
<td>Error while restoring data (value)</td>
</tr>
<tr>
<td>ERR_SEND_DATA</td>
<td>Error occurred while sending data to CM (value)</td>
</tr>
<tr>
<td>ERR_SERIALIZE_EDR</td>
<td>Exception occured while serializing edr (value)</td>
</tr>
</tbody>
</table>

FCT_PreRating

Table 19–52 lists the FCT_PreRating error messages.

Table 19–52  FCT_PreRating Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_ASS_CBD_NOT_FOUND</td>
<td>No ASSOCIATED_CHARGE_BREAKDOWN block is found</td>
</tr>
<tr>
<td>ERR_NO_RATEPLAN</td>
<td>No rateplan-code or -id defined in DETAL.ASS_CBD.CP</td>
</tr>
<tr>
<td>ERR_RATEPLAN_NOT_A_NUMBER</td>
<td>Rateplan-Id 'value' is not a number.</td>
</tr>
<tr>
<td>ERR_RATEPLAN_TYPE_INV</td>
<td>Rateplan-Type 'value' does not match valid values ('R').</td>
</tr>
<tr>
<td>ERR_RATEPLAN_VERSION_NOT_FOUND</td>
<td>Rateplan-Version not found for Id (value) and Date (value).</td>
</tr>
<tr>
<td>ERR_ZONE_VALUE_NOT_FOUND</td>
<td>ZoneValue not found for ZM-Id=value, Date=value, SC=value, A#=value, B#=value.</td>
</tr>
</tbody>
</table>

FCT_PreSuspense

Table 19–53 lists the FCT_PreSuspense error messages.

Table 19–53  FCT_PreSuspense Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_DATATYPE_NOT_SUPPORTED</td>
<td>Datatype EDR::FieldDescr::value is not supported.</td>
</tr>
<tr>
<td>ERR_NO_QUERYABLE_FIELDS_TABLE</td>
<td>No queryable fields specified in registry.</td>
</tr>
<tr>
<td>WRN_NO_QUERYABLE_FIELDS</td>
<td>No queryable fields specified for table value.</td>
</tr>
</tbody>
</table>

FCT_RateAdjust

Table 19–54 lists the FCT_RateAdjust error message.

Table 19–54  FCT_RateAdjust Error Message

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_ADD_RATEADJUST</td>
<td>Failure while adding rate adjust entry (value).</td>
</tr>
</tbody>
</table>

FCT_Reject

Table 19–55 lists the FCT_Reject error message.

Table 19–55  FCT_Reject Error Message

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_REJECT_UNKNOWN_STREAM</td>
<td>Error : Stream 'value' is unknown.</td>
</tr>
</tbody>
</table>
FCT_Rounding

Table 19–56 lists the FCT_Rounding error message.

Table 19–56  FCT_Rounding Error Message

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_APP_NAME_NOT_FOUND</td>
<td>The Application Type name value provided for the Registry Entry value is not supported.</td>
</tr>
</tbody>
</table>

FCT_SegZoneNoCust

Table 19–57 list the FCT_SegZoneNoCust error message.

Table 19–57  FCT_SegZoneNoCust Error Message

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_INIT_SEG_ZONE_LINK</td>
<td>Failure during initialization of segment zone link table.</td>
</tr>
</tbody>
</table>

FCT_Suspense

Table 19–58 lists the FCT_Suspense error messages.

Table 19–58  FCT Suspense Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_ASS_SUSPENSE_EXT_MISSING</td>
<td>No ASS_SUSPENSE_EXT data block. FCT_PreSuspense is not active.</td>
</tr>
<tr>
<td>ERR_EFENTRY_INVALID</td>
<td>Failed to setup suspense mapping entry (value)</td>
</tr>
<tr>
<td>ERR_NO_DEFAULT_SUSPENSE_REASON</td>
<td>No default suspense reason and subreason specified.</td>
</tr>
<tr>
<td>ERR_NO_MATCHING_ENTRY_IN_EDR_FLD_MAP</td>
<td>No matching entry found in suspense edr field map table.</td>
</tr>
<tr>
<td>ERR_REJECT_STREAM_ERROR</td>
<td>RejectStream registry must be set to “value”.</td>
</tr>
<tr>
<td>WRN_FIELD_MAP_NOT_FOUND</td>
<td>Registry entry ‘value’ not found.</td>
</tr>
</tbody>
</table>

FCT_Timer

Table 19–59 lists the FCT_Timer error messages.

Table 19–59  FCT_Timer Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_CAN_NOT_SCHEDULE_TIMER</td>
<td>Can't schedule timer.</td>
</tr>
<tr>
<td>ERR_CAN_NOT_RESCHEDULE_TIMER</td>
<td>Can't re-schedule timer id 'value'.</td>
</tr>
<tr>
<td>ERR_CAN_NOT_RESET_TIMER</td>
<td>Can't reset timer id 'value'.</td>
</tr>
<tr>
<td>ERR_CAN_NOTCANCEL_TIMER</td>
<td>Can't cancel timer id 'value'.</td>
</tr>
</tbody>
</table>

FCT_TriggerBill

Table 19–60 lists the FCT_TriggerBill error message.
Table 19–60  FCT_TriggerBill Error Message

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_TRIGGER_BILLING</td>
<td>TriggerBilling is required.</td>
</tr>
</tbody>
</table>

FCT_UoM_Map

Table 19–61 lists the FCT_UoM_Map error messages.

Table 19–61  FCT_UoM_Map Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_INIT_PBC_SERVICE_MAP_TABLE</td>
<td>Failed to init. service map table (value).</td>
</tr>
<tr>
<td>ERR_INIT_PBC_SERVICE_RGL_MAP_TABLE</td>
<td>Failed to init. rgl map table (value).</td>
</tr>
<tr>
<td>ERR_INIT_UoM_MAPTABLE</td>
<td>Failed to initialise the UoM map table (value).</td>
</tr>
<tr>
<td>ERR_NO_SERVICE_CODE_NAME_SUPPLIED</td>
<td>No service code value was supplied for the mapping (value).</td>
</tr>
</tbody>
</table>

FCT_UsageClassMap

Table 19–62 lists the FCT_UsageClassMap error message.

Table 19–62  FCT_UsageClassMap Error Message

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_UCENTRY_INVALID</td>
<td>Failed to setup usage class mapping entry (value).</td>
</tr>
</tbody>
</table>

FCT_USC_Map

Table 19–63 lists the FCT_USC_Map error messages.

Table 19–63  FCT_USC_Map Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_SETUP_USC_MAPTABLE</td>
<td>Failed to initialise the USC map table (value).</td>
</tr>
<tr>
<td>ERR_USC_GROUP_VALUE_NOT_FOUND</td>
<td>No specified for USC group registry parameter.</td>
</tr>
</tbody>
</table>

INP_GenericStream

Table 19–64 lists the INP_GenericStream error message.

Table 19–64  INP_GenericStream Error Message

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_INP_GENERICSTREAM_PARSE_ERROR_STREAM</td>
<td>Parse error on input stream: value.</td>
</tr>
</tbody>
</table>

INP_Realtime

Table 19–65 lists the INP_Realtime error messages.
<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_REALTIME_BLOCKINDEX_NOT_IN_FACTORY</td>
<td>RealtimePipeline: Container BlockIndex is missing in Factory: value</td>
</tr>
<tr>
<td>ERR_REALTIME_CANNOT_CREATE_DOMBUILDER</td>
<td>RealtimePipeline: Unable to create a DOMBuilder parser from DOMImplementationLS object.</td>
</tr>
<tr>
<td>ERR_REALTIME_CANNOT_DUPLICATE_EDR</td>
<td>RealtimePipeline: Unexpected to duplicate EDR.</td>
</tr>
<tr>
<td>ERR_REALTIME_COMPILE_ISCRIPT_FAILED</td>
<td>RealtimePipeline: Unable to compile iscript mapping file: value</td>
</tr>
<tr>
<td>ERR_REALTIME_CREATE_DEFAULT_EDR FAILED</td>
<td>RealtimePipeline: Unable to create a default edr for CM error propagation.</td>
</tr>
<tr>
<td>ERR_REALTIME_CREATING_CONTAINER_INDEX</td>
<td>RealtimePipeline: Unable to create EDR Container Index for container name: value</td>
</tr>
<tr>
<td>ERR_REALTIME_EDR_FIELD_MAPPING_MISMATCH</td>
<td>RealtimePipeline: EDRField type mismatch error, unable to map value:value to EDR field type: value in function readConstants()</td>
</tr>
<tr>
<td>ERR_REALTIME_EDR_TYPE_MISMATCH</td>
<td>RealtimePipeline: Type mismatch error, unable to map pin_fld_t:value to EDR field: value in function appendValueToEDR()</td>
</tr>
<tr>
<td>ERR_REALTIME_EXECUTE_ISCRIPT_FAILED</td>
<td>RealtimePipeline: Failed to execute realtime iscript.</td>
</tr>
<tr>
<td>ERR_REALTIME_INDEX_NOT_IN_FACTORY</td>
<td>RealtimePipeline: Container Index is missing in Factory: value</td>
</tr>
<tr>
<td>ERR_REALTIME_INSERT_FLIST_EXT</td>
<td>RealtimePipeline: IScript Flist extension is missing.</td>
</tr>
<tr>
<td>ERR_REALTIME_MISSING_INPUT_MAP_LIST</td>
<td>RealtimePipeline: XML element list &lt;InputMap&gt; missing in xml file: value</td>
</tr>
<tr>
<td>ERR_REALTIME_MISSING_INPUT_MAP_NODE</td>
<td>RealtimePipeline: XML element &lt;InputMap&gt; missing in xml file: value</td>
</tr>
<tr>
<td>ERR_REALTIME_MISSING_OPCODE_LIST</td>
<td>RealtimePipeline: XML element list &lt;OpcodeMap&gt; missing in xml file: value</td>
</tr>
<tr>
<td>ERR_REALTIME_MISSING_OPCODE_NODE</td>
<td>RealtimePipeline: XML element &lt;OpcodeMap&gt; missing in xml file: value</td>
</tr>
<tr>
<td>ERR_REALTIME_MISSING_REQUIRED_FLIST_FIELD</td>
<td>RealtimePipeline: The expected Input Flist field is missing: value</td>
</tr>
<tr>
<td>ERR_REALTIME_NULL_EDR_FACTORY</td>
<td>RealtimePipeline: Null EDR factory returned from edrFactory()</td>
</tr>
<tr>
<td>ERR_REALTIME_OBS_FLIST_EDR_CONVERSION_FAILED</td>
<td>Realtime Pipeline: Conversion of Input flist to EDR failed.</td>
</tr>
<tr>
<td>ERR_REALTIME_OPEN_ISCRIPT_FAILED</td>
<td>RealtimePipeline: Unable to open iscript mapping file: value</td>
</tr>
<tr>
<td>ERR_REALTIME_PRODUCING_EDR</td>
<td>RealtimePipeline: Unable to produce EDR Container from index named: value</td>
</tr>
<tr>
<td>ERR_REALTIME_PROPAGATE_BAD_DATABLOCK_VALUE</td>
<td>RealtimePipeline: PropagateBlock blockname:value, unable to find target EDR::DatablockValue at index [value,value]</td>
</tr>
<tr>
<td>ERR_REALTIME_PROPAGATE_BLOCK.Clone</td>
<td>RealtimePipeline: PropagateBlock blockname:value, unable to clone source EDR::Datablock at index value</td>
</tr>
<tr>
<td>ERR_REALTIME_PROPAGATE_BLOCK_INVALID_ARG</td>
<td>RealtimePipeline: PropagateBlock blockname:value is not a valid candidate for propagation.</td>
</tr>
</tbody>
</table>
Table 19–66  (Cont.) INP_Realtime Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_REALTIME_PROPAGATE_BLOCK_INVALID_DEPTH</td>
<td>RealtimePipeline: PropagateBlock blockname:value has an invalid depth for propagation.</td>
</tr>
<tr>
<td>ERR_REALTIME_PROPAGATE_BLOCK_LOOKUP_SOURCE_BLOCK</td>
<td>RealtimePipeline: PropagateBlock blockname:value has missing source EDR::Datablock at index value.</td>
</tr>
<tr>
<td>ERR_REALTIME_PROPAGATE_BLOCK_MISSING</td>
<td>RealtimePipeline: PropagateBlock blockname:value is not a member of the current EDR.</td>
</tr>
<tr>
<td>ERR_REALTIME_PUSH_EDR_FAILED</td>
<td>RealtimePipeline: Failed to push EDRs onto InputDevice.</td>
</tr>
<tr>
<td>ERR_REALTIME_READ_REGISTRY_FAILED</td>
<td>RealtimePipeline: ReadRegistry() failed for INP::Realtime module.</td>
</tr>
<tr>
<td>ERR_REALTIME_REG_INTERPRETER</td>
<td>RealtimePipeline: IScript interpreter is missing.</td>
</tr>
<tr>
<td>ERR_REALTIME_SET_INTERNAL_TRANSACION_ID_FAILED</td>
<td>RealtimePipeline: Failed to set Transaction Id in Internal block.</td>
</tr>
<tr>
<td>ERR_REALTIME_UNKNOWN_EXCEPTION</td>
<td>RealtimePipeline: Unknown exception in function: value</td>
</tr>
</tbody>
</table>

INP_Reycle

Table 19–66 lists the INP_Reycle error messages.

Table 19–66  INP_Reycle Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_ADD_HEADER_RECORD_ERROR</td>
<td>Error while adding header record.</td>
</tr>
<tr>
<td>ERR_ADD_TRAILER_RECORD_ERROR</td>
<td>Error while adding trailer record.</td>
</tr>
<tr>
<td>ERR_INP_RECYCLE_CANNOT_CONVERT_EDR</td>
<td>Cannot convert EDR to new container desc, field is either missing or type has changed: value.</td>
</tr>
<tr>
<td>ERR_INP_RECYCLE_CANNOT_FIND_EDR_VERSION</td>
<td>INP::EdrVersionConverter::getEdrVersion(..) is unable to find the EDR version attribute in the inputed xml.</td>
</tr>
<tr>
<td>ERR_INP_RECYCLE_DOM_PARSE_ERRORS</td>
<td>DOMParser errors encountered during parse operation in function value.</td>
</tr>
<tr>
<td>ERR_INP_RECYCLE_EDR_MISSING_FM</td>
<td>EDR xml missing FM_ELEMENT in function value.</td>
</tr>
<tr>
<td>ERR_INP_RECYCLE_EMPTY_QUERY_RESULT</td>
<td>Query for edr_fld_map_buf_t buffer returned empty data in function: value.</td>
</tr>
<tr>
<td>ERR_INP_RECYCLE_INVALID_READER</td>
<td>Invalid reader in function: value - value.</td>
</tr>
<tr>
<td>ERR_INP_RECYCLE_NO_DB_CONNECTION</td>
<td>Unable to get DB Connection in function: value - value.</td>
</tr>
<tr>
<td>ERR_INP_RECYCLE_NO_DB_SELECTOR</td>
<td>Unable to get DB Selector in function: value - value.</td>
</tr>
<tr>
<td>ERR_INP_RECYCLE_NO_DB_TABLES</td>
<td>Unable to get DB Tables edr_field_mapping_t and edr_fld_map_buf_t in function: value.</td>
</tr>
<tr>
<td>ERR_INP_RECYCLE_ROOT_ELEMENT_MISSING</td>
<td>EDR root XML element not found in function value.</td>
</tr>
<tr>
<td>ERR_INP_RECYCLE_SAX2PARSE_ERRORS</td>
<td>SAX2XMLReader errors encountered during parse operation in function value.</td>
</tr>
<tr>
<td>ERR_INP_RECYCLE_UNEXPECTED_EXCEPTION</td>
<td>Unexpected Exception in function value.</td>
</tr>
<tr>
<td>ERR_PROCESS_HEADER_ERROR</td>
<td>Error in parsing header record.</td>
</tr>
<tr>
<td>ERR_PROCESS_RECORD_ERROR</td>
<td>Error in processing record: value.</td>
</tr>
</tbody>
</table>
Table 19–66 (Cont.) INP_Recycle Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_PUSH_EDR_ERROR</td>
<td>Error in pushing record to pipeline, record number: value.</td>
</tr>
<tr>
<td>ERR_RECYCLING_DATA_MODULE_NOT_FOUND</td>
<td>Cannot find recycling data module.</td>
</tr>
<tr>
<td>ERR_REG_TRAILER_NOT_SUPPORTED</td>
<td>Trailer record is not supported by INP_Recycle.</td>
</tr>
</tbody>
</table>

NET_EM

Table 19–67 lists the NET_EM error messages.

Table 19–67 NET_EM Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_CLOSE_REALTIME_TRANSACTION</td>
<td>Failed to close realtime transaction in pipeline 'value'.</td>
</tr>
<tr>
<td>ERR_CREATE_CONTEXT_FAILED</td>
<td>PCM_CREATE_CONTEXT failed for socket 'value'.</td>
</tr>
<tr>
<td>ERR_LSOCK_BIND</td>
<td>Unix Sock bind error for file name 'value'.</td>
</tr>
<tr>
<td>ERR_OPCODE_NOT_CONFIGURED</td>
<td>Opcode 'value' is not configured.</td>
</tr>
<tr>
<td>ERR_OPEN_REALTIME_TRANSACTION</td>
<td>Failed to open realtime transaction in pipeline 'value'.</td>
</tr>
<tr>
<td>ERR_RTP_ARE_NOT_READY</td>
<td>All Realtime Pipelines NOT ready.</td>
</tr>
<tr>
<td>ERR_SOCK_ACCEPT</td>
<td>Accept failed for socket 'value', errno 'value'.</td>
</tr>
<tr>
<td>ERR_SOCK_BIND</td>
<td>TCP/IP Socket bind error 'value', errno 'value'.</td>
</tr>
<tr>
<td>ERR_SOCK_LISTEN</td>
<td>Listen failed for socket 'value', errno 'value'.</td>
</tr>
<tr>
<td>ERR_SOCKET_BIND</td>
<td>Socket bind error.</td>
</tr>
</tbody>
</table>

OUT_DB

Table 19–68 lists the OUT_DB error messages.

Table 19–68 OUT_DB Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_DB_ROLLBACK_TRANSACTION</td>
<td>Database transaction rollback failed for stream 'value'.</td>
</tr>
<tr>
<td>ERR_DB_STREAM_CLOSE</td>
<td>Database could not closed for stream 'value'.</td>
</tr>
<tr>
<td>ERR_DB_STREAM_OPEN</td>
<td>Database open failed for stream 'value'.</td>
</tr>
<tr>
<td>ERR_FILE_NOT_CONSISTENT</td>
<td>Parameter 'value' file not consistent.</td>
</tr>
<tr>
<td>ERR_INDEXLIST_NOT_CREATED</td>
<td>Couldn’t create index table for edr-container fields.</td>
</tr>
</tbody>
</table>

OUT_GenericStream

Table 19–69 lists the OUT_GenericStream error messages.

Table 19–69 OUT_GenericStream Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_OUTPUT_PARSE_ERROR</td>
<td>Parse error on output file: value.</td>
</tr>
<tr>
<td>ERR_STREAM_IS_EMPTY_RETURN</td>
<td>Function streamIsEmpty() needs boolean return type.</td>
</tr>
</tbody>
</table>
OUT Realtime

Table 19–70 lists the OUT Realtime error messages.

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_OUT_REALTIME_CREDIT_LIMIT_CHECK_FAILED</td>
<td>RealtimePipeline: CreditLimitCheck failed.</td>
</tr>
<tr>
<td>WRN_OUT_REALTIME_REVERSE_RATINGPLIED</td>
<td>RealtimePipeline: Reverse Rating Applied.</td>
</tr>
</tbody>
</table>

Pipeline Utility Error Messages

Table 19–71 lists the LoadIFWConfig error messages.

LoadIFWConfig

Table 19–71 LoadIFWConfig Error Messages

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_CREATE_OBJECT_FAILED</td>
<td>Cannot create object value (invalid (NULL) pointer).</td>
</tr>
<tr>
<td>ERR_INVALID_PATTERN</td>
<td>Directory pattern value is invalid.</td>
</tr>
<tr>
<td>ERR_NO_DIR</td>
<td>Directory value not accessible.</td>
</tr>
<tr>
<td>ERR_NO_PATH_NAME</td>
<td>No path name given.</td>
</tr>
<tr>
<td>ERR_OBJ_NOT_INITIALIZED</td>
<td>The object value is not initialized.</td>
</tr>
<tr>
<td>ERR_REG_NAME_NOT_FOUND</td>
<td>Registry name value not found.</td>
</tr>
<tr>
<td>ERR_REG_PARSE_FAILED</td>
<td>Registry parse failed near value.</td>
</tr>
<tr>
<td>ERR_REG_SUBTREE_NOT_FOUND</td>
<td>Registry subtree value not found.</td>
</tr>
<tr>
<td>ERR_REG_VALUE_IS_EMPTY</td>
<td>Found empty value for registry item, where a value was expected.</td>
</tr>
<tr>
<td>ERR_SYSTEM_ERROR</td>
<td>Unexpected error, errno value value value</td>
</tr>
<tr>
<td>WRN_REG_ENTRY_OBSOLETE</td>
<td>Obsolete registry entry: value</td>
</tr>
</tbody>
</table>

OMF Error Messages

Table 19–72 lists the OMF error messages.

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR_SNMP_LOST_CONNECTION</td>
<td>Lost connection to master SNMP agent.</td>
</tr>
<tr>
<td>ERR_SNMP_NOT_REGISTERED</td>
<td>Failed to register the mib table OID value for probe value at master SNMP agent.</td>
</tr>
<tr>
<td>ERR_SNMP_NOT_UNREGISTERED</td>
<td>Failed to un-register the mib table OID value for probe value at master SNMP agent.</td>
</tr>
<tr>
<td>ERR_SNMP_SUBAGENT_MIB_INIT</td>
<td>Failed to initialize SNMP subagent mib. Registration will be re-attempted periodically.</td>
</tr>
</tbody>
</table>
Part V describes how to use Oracle In-Memory Database (IMDB) Cache Manager in your Oracle Communications Billing and Revenue Management (BRM) system. It includes the following chapters:

- Using Oracle IMDB Cache Manager
- Creating and Initializing the BRM Cache Group Schema
- Customizing BRM for In-Memory Database Cache DM
Using Oracle IMDB Cache Manager

This document provides information about using the Oracle In-Memory Database (IMDB) Cache to cache Oracle Communications Bill and Revenue Management (BRM) objects in memory.

About Oracle IMDB Cache Manager

IMDB Cache Manager is an optional BRM component that enables you to use both:

- **Oracle IMDB Cache**: An in-memory database for caching only performance-critical data from the BRM database.
- **BRM database**: An external relational database management system (RDBMS) for storing all BRM data.

The Oracle IMDB Cache Manager package includes only the IMDB Cache Data Manager (DM). The DM provides the interface between the CM and IMDB Cache, and between the CM and the BRM database. The DM determines whether to route CM requests to IMDB Cache or to the BRM database based on where the data resides (called the *residency type*). For more information about IMDB Cache DM, see *About IMDB Cache DM*.

About Caching BRM Objects In Memory

You can cache a subset of your BRM database tables in an in-memory database by using IMDB Cache. Caching BRM data improves transaction response times and increases system throughput:

- Creating, modifying, deleting, and searching for objects in memory eliminates the network connections for communicating with the BRM database server, thus improving transaction response times.
- Caching database tables in IMDB Cache reduces the workload on the BRM database, thus improving the overall system throughput.

For more information about IMDB Cache, see *About Oracle IMDB Cache*.

About BRM Cache Groups

IMDB Cache uses cache groups to cache BRM database tables in memory. A cache group can have a single BRM table or a group of related tables (root table and one or more child tables). IMDB Cache synchronizes data between the cache groups and the BRM database.
BRM cache groups store BRM data that require low latency and high throughput for fast access. For example:

- Subscriber data to process authentication, authorization, and accounting (AAA) requests
- Subscriber data accessed by customer service representatives (CSRs)
- Account, bill unit, and item data needed to perform billing

The default BRM cache groups include the subscriber cache group and event cache groups. BRM cache groups are defined as ASYNCHRONOUS WRITETHROUGH (AWT) cache group and Dynamic AWT cache group types, where updates in the cache group are asynchronously propagated to the BRM database. Additionally, BRM cache groups are also defined as Global AWT cache group types to make the data available to all members in a cache grid.

The subscriber cache group is defined as Global AWT cache group. If the requested data is not in the Global AWT cache group, Oracle IMDB Cache returns an error.

The event cache groups are defined as Dynamic AWT cache group. By default, data in a Dynamic AWT cache group is deleted using the least-recently-used (LRU) aging policy. If the requested data is not in the Dynamic cache group, Oracle IMDB Cache forwards the request to the BRM database, gets the response, caches the data, and sends the response to the client.

---

**Note:** For the default BRM cache groups, aging is only implemented for event cache groups.

---

See *Oracle In-Memory Database Cache User’s Guide* for more information about cache group types and aging policy.

You can view the default BRM cache group definitions in the `BRM_Home/bin/pin_tt_schema_gen.values` configuration file.

You create the BRM cache groups when you configure In-Memory Database (IMDB) Cache DM. BRM cache groups are created in the BRM data store in the Oracle IMDB Cache.

When you create the BRM data store, you set the size of the data store specified by the `PermSize` cache attribute. The `PermSize` cache attribute value specifies the size of the shared-memory segment in the data store. This is the maximum size to which data can grow in the cache groups. You can increase the value of `PermSize` when you connect to the data store the first time, but you cannot decrease it. If you attempt to decrease the value of `PermSize`, IMDB Cache issues a warning message.

---

**Note:** When loading data from the BRM database or when creating new accounts in the BRM cache groups, IMDB Cache reports an out-of-memory error if the data in the BRM cache groups exceeds the size of the `PermSize` attribute.

---

**Guidelines for Setting Cache Group Size**

Not all BRM objects can be loaded into the BRM cache groups. The amount of data that can be loaded is dependent on the size of the shared-memory segment of the IMDB Cache for the BRM data store. The default value for `PermSize` is 32 (megabytes). You can change this value based on the size of data to be cached. To estimate the size of the data store, you can use the `ttSize` utility.
See the discussion on specifying the size of a data store in the *Oracle TimesTen In-Memory Database Operations Guide* for more information.

**About Loading BRM Objects into IMDB Cache**

---

**Important:** If you have existing BRM data created on a BRM system before IMDB Cache Manager was installed, you must run the `load_pin_uniqueness` utility before loading BRM objects into the Oracle IMDB Cache.

---

You load BRM objects into IMDB Cache by using the `pin_tt_schema_gen` utility. You run the utility on the BRM database to generate schema and load SQL scripts, which are used to create and initialize your BRM cache groups in IMDB Cache.

The `pin_tt_schema_gen` utility generates the SQL files based on the BRM cache group specifications in the `pin_tt_schema_gen.values` files.

You manually run the schema and load SQL scripts on IMDB Cache to create and initialize the BRM cache groups schema.

For more information, see "Creating and Initializing the BRM Cache Group Schema".

**About Loading Migrated Accounts into the Cache Groups**

If you migrated accounts into the BRM database after your BRM system is up and running, you can load the migrated data into the BRM cache groups using the `pin_tt_schema_gen` utility. To perform the migration without downtime or without blocking operations on the objects already in the cache, load only the objects that are not already cached in the Oracle IMDB Cache data store.

To reduce the impact on your system, Oracle recommends that you perform the load in batches.

**About Managing Data in IMDB Cache**

Database tables cached in IMDB Cache are stored in shared memory, which is a fixed size. You must manage the data in the cache to avoid running out of shared-memory space.

**Managing Fast-Growing Tables**

To manage fast-growing tables, such as `/event` tables, in BRM cache groups, use the IMDB Cache Aging feature.

IMDB Cache DM uses *usage-based* aging to remove least-recently-used (LRU) data, based on a range of threshold values that specifies when LRU aging is activated and deactivated.

The aging policy for BRM cache groups is defined in the cache group schema definitions in the `pin_tt_schema_gen.values` file. Aging is enabled by default for DYNAMIC cache groups. See "Creating and Initializing the BRM Cache Group Schema".

**About Purging Expired Reservation and Active-Session Objects**

When account balances in the BRM database are updated at the end of a session, reservation objects and active-session objects are removed from IMDB Cache. The reservations are released and session objects in the database are rated. However, when
a session is terminated abnormally (for example, because of a timeout error or not receiving a stop-accounting request), the session remains open and the reservation objects and the related objects in that session remain in the database cache, using up shared-memory space. You remove expired reservation and active session objects and release shared-memory space by using the `pin_clean_rsvns` and `pin_clean_asos` utilities.

For information on how to run these utilities, see "pin_clean_rsvns" and "pin_clean_asos".

**About Purging Closed Bills, Items, Journals, and Expired Sub-Balances**

You remove closed bills, items, journals, and expired account sub-balances by using the `pin_purge` utility.

---

**Note:** Do not use the `pin_sub_bal_cleanup` utility that is installed with BRM 7.4 to purge expired account sub-balances; using the `pin_sub_bal_cleanup` utility results in data in IMDB Cache and the BRM database being unsynchronized and may result in propagation errors. In a BRM system with IMDB Cache DM, you must use the `pin_purge` utility to purge expired account sub-balances.

---

See "pin_purge" for more information on how to run this utility.

**Using the IMDB Cache Grid to Partition Data**

An IMDB Cache Grid is a collection of Oracle IMDB Caches that work together to cache data from the Oracle database. Cached data is distributed between the cache grid members. The cache grid manages information about the data location and provides applications access to the cached data in all the grid members with location transparency. Logical partitioning is a mechanism that provides scalability by using the Oracle IMDB Cache Grid architecture. You can configure cache grids to increase system performance and manage system growth.

Typically in a BRM system, data is distributed among multiple database schemas. You can improve scalability by distributing BRM data among multiple logical partitions in addition to the multiple schemas. You configure a cache grid for each database schema. In this configuration, data from a single database schema is distributed among multiple logical partitions (Oracle IMDB Caches). Logical partitioning essentially provides a second level of data partitioning. As your subscriber base grows, you can add additional database schemas and logical partitions. For example, for a single-schema database with 50,000 subscribers, you can create two logical partitions and distribute 25,000 subscriber accounts in each partition. When the subscriber base grows to 100,000, you can add an additional database schema with two more logical partitions to distribute the additional 50,000 new subscriber accounts.

In a high-availability system, a cache grid consists of one or more high-availability nodes (a pair of an active and a standby Oracle IMDB Cache). Data in the schema is partitioned among the Oracle IMDB Cache nodes based on account POID_DB (database number). Accounts and all associated objects reside in the same node. Each node owns the specific data partition. If one node is down, clients can continue to access accounts in the other nodes, and, because database operations are performed at the nodes where the data resides, this reduces the workload on the BRM database, thus improving the overall system efficiency.
In a BRM system with high availability and logical partitions, each IMDB Cache DM handles operations for a specific logical partition and database schema. All logical partitions are assigned a database number. The following naming scheme is used to denote the logical partition for a specific database schema:

0.M.0.N

where

- M specifies the logical partition.
- N specifies the database schema.

**Note:** The first logical partition is 0, not 1. For example, for database schema 0.0.0.1 (schema #1), the first partition will be 0.0.0.1 and the second partition will be 0.1.0.1. For database schema 0.0.0.2 (schema #2), the first partition will be 0.0.0.2 and the second partition will be 0.1.0.2.

Figure 20–1 shows an IMDB Cache Grid architecture for a single-schema database with two logical partitions and how you can configure additional database schemas and logical partitions as your subscriber base grows.
How Accounts Are Assigned to the Logical Partition

Each database schema and logical partition is assigned a status and a priority. The status determines whether the database schema and the logical partition are available.
for account creation. The priorities are used during account creation to determine in which schema and partition to create the account.

BRM finds an open database schema with the highest priority and assigns accounts to an open logical partition with highest priority for that database schema. If all database schemas and logical partitions have the same priority, BRM chooses an open database schema and logical partition at random in which to create the account.

For hierarchical, brand, or group accounts, all members are created in the same database schema and generally in the same logical partition as the parent. However, it is possible that members could be located in different logical partitions based on the logical partition priority.

The database schema and logical partition status can be set to open, closed, or unavailable:

- Open status: Database schemas and logical partitions with open status are available for account creation.

- Closed status: Database schemas and logical partitions with closed status are not used for account creation under most circumstances. Accounts are created in a closed database schema or logical partition only if an account’s parent, branded, or sponsoring account belongs to that database schema or logical partition or if all database schemas and logical partitions are closed. If all database schemas and logical partitions are closed, BRM chooses a closed database schema and logical partition at random in which to create accounts and continues to create accounts in that database schema until a database schema becomes open. To limit the number of accounts created in a database schema or logical partition, you can manually change the status to closed, or you can have BRM automatically change it to closed when the database schema or logical partition reaches a predefined limit.

- Unavailable status: Database schemas and logical partitions with unavailable status are not used for account creation unless the database schema or logical partition contains an account’s parent, sponsoring, or branded account.

To change the status or priority for a database schema or logical partition, edit the STATUS and PRIORITY entries in the $BRM_Home/apps/multi_db/config_dist.conf$ file and then use the $load_config_dist$ utility to load the distribution information into the primary database.

**How the CM Locates Accounts in the Logical Partition**

For a single-schema database with logical partitions, CM can perform a search on all grid members to locate the account and its services.

For a multi-schema database with logical partitions, BRM stores the logical partition and database schema information about the account in the /uniqueness object in the BRM database. When the CM receives an AAA request, it sends a search request to the IMDB Cache DM to locate the account. IMDB Cache DM forwards the request to the BRM database. The BRM database performs a search on the /uniqueness object and returns the logical partition and database schema information corresponding to the location where the account resides.

**About Setting Up the CM for Logical Partitioning**

When you set up a IMDB Cache DM-enabled system, you must set entries in the CM configuration file ($pin.conf$) for IMDB Cache and IMDB Cache DM.

You must set the $timesten$ entry to 1, if your BRM system is IMDB Cache DM enabled. This indicates to the CM that BRM system is configured with IMDB Cache.
Set the `logical_partition` entry to 1, if your BRM system is configured with logical partitions. When the CM receives requests from the client applications, if `logical_partition` is enabled, it sends the requests to the appropriate cache grid node.

To specify the nodes in the cache grid, set the `dm_pointer` entry. Each `dm_pointer` entry specifies a node consisting of a primary and secondary IMDB Cache DM pair with the corresponding host name and listen port.

You can set DM attributes (`dm_attributes`) for each IMDB Cache DM to which you connect. You set the `dm_attributes` entry using the following options:

- **scoped:** Use this option only when you use branding. It restricts database access for separate brands. You can disable this option if branding is not used. See "Adding new client applications" in the BRM 7.4 documentation.
- **assign_account_obj:** Assigns an owner account reference when the object is created. If you use branding, all objects are associated with an account. You can disable this option if branding is not used.
- **searchable:** Limits access to certain logical partitions in a multi-schema environment. The CM can search all nodes in a single cache grid or perform a global search on all nodes in all the cache grids. This option lets the CM know which nodes are searchable.

Certain object type searches may return duplicate results when the search is performed on multiple logical partitions; for example, a search on `/config` objects. In order to prevent duplicate search results, the `single_node_search_object_list` entry is used. It specifies the default list of object types for which search is provided only in one (current) node. In other words, for the specified object types the schema search (search across nodes) is not provided. If you want to exclude the search for other object types, add the object type to the list.

The following is an example of the CM configuration entries:

```plaintext
- cm timesten 1
- cm logical_partition 1
- cm dm_pointer 0.0.0.1 ip 156.151.2.168 33950 ip 168.35.37.128 12960
- cm dm_attributes 0.0.0.1 assign_account_obj, scoped, searchable
- cm dm_pointer 0.1.0.1 ip 156.151.2.168 33920 ip 168.35.37.128 12940
- cm dm_attributes 0.1.0.1 assign_account_obj, scoped, searchable
- cm single_node_search_object_list
  /event,/config,/device,/product,/deal,/discount,/plan,/zonemap,/group,/rate,
  /rate_plan,/tmp_unprocessed_events,/rate_plan_selector,/rate_change,/string,/batch,/search,/uniqueness,/profile
```

**How IMDB Cache DM Connects to IMDB Cache**

IMDB Cache DM uses a direct driver connection to connect to IMDB Cache. For a direct driver connection, you must configure IMDB Cache DM and IMDB Cache on the same server machine.

See "Connecting IMDB Cache DM to the Data Store" for more information.

For more information about the direct driver connection mode, see Oracle In-Memory Database Introduction.

**How IMDB Cache DM Handles Both IMDB Cache and BRM Database Transactions**

Certain operations in the BRM system require updates to reference objects in IMDB Cache as well as updates to objects in the BRM database in a single transaction. **Dual**
connection allows IMDB Cache DM to maintain separate simultaneous connections to IMDB Cache and the BRM database. For example, to create an account, IMDB Cache DM uses dual connection to create subscriber data in IMDB Cache and to update configuration data stored in the BRM database.

However, for high availability, BRM uses IMDB Cache TWOSAFE replication configuration. In this configuration, IMDB Cache DM cannot perform operations on two different databases in a single transaction. To resolve this, IMDB Cache DM uses dual connection and dual transaction. Dual transaction allows IMDB Cache DM to maintain separate transactions to IMDB Cache and the BRM database. Using dual transaction, IMDB Cache DM opens a second database transaction to update data that exists only in the BRM database.

**Important:** IMDB Cache DM’s dual transaction commits one database transaction at a time. It does not support a two-phase commit. If one transaction succeeds and the other transaction fails, it is not possible to roll back the transaction that completed, which may cause unpredictable behavior.

For more information about TWOSAFE replication and PassThrough queries, see the Oracle TimesTen In-Memory Database TimesTen to TimesTen Replication Guide.

### About the AAA Flow in a BRM System with IMDB Cache Manager

In an IMDB Cache-enabled BRM system, the authentication, authorization, and accounting (AAA) flow uses the /uniqueness table for account and service lookups to authorize accounts for prepaid services. The /uniqueness object contains the data required for BRM to find the account and service objects corresponding to the given login.

The PCM_OP_ACT_FIND opcode uses the /uniqueness object to obtain /account and /service data for a given account login. The Services Framework AAA opcodes use the data in the /uniqueness table to authorize and reauthorize prepaid sessions as well as end prepaid accounting sessions. The Customer FM standard opcodes update the /uniqueness table when a new customer account is created or when a customer purchases a new service.
21

Creating and Initializing the BRM Cache Group Schema

This chapter describes the Oracle Communications Billing and Revenue Management (BRM) \texttt{pin\_tt\_schema\_gen} utility, how to configure it, and how to use it.

About Creating and Initializing BRM Cache Group Schema

You create and initialize the BRM cache group schema by using the \texttt{pin\_tt\_schema\_gen} utility. This utility performs the following tasks:

- Generates the \texttt{tt\_schema.sql} file used to create the BRM cache group schema
- Generates the \texttt{tt\_load.sql} file used to load the BRM objects in the cache groups
- Generates the \texttt{tt\_drop.sql} file used to drop the BRM cache group schema
- Creates missing indexes and not null constraints on the BRM database

You use the Oracle IMDB Cache tt\_sql tool to run the \texttt{tt\_schema.sql} script on the Oracle IMDB Cache database to create the BRM cache groups schema. After running \texttt{tt\_schema.sql}, run the \texttt{tt\_load.sql} script to preload the corresponding BRM objects from the BRM database into the BRM cache groups.

\textbf{Note:} In a high-availability system with Oracle Clusterware, Oracle Clusterware automatically duplicates the complete data store from the active to the standby node.

However, in a high-availability system without Oracle Clusterware, you need to run \texttt{tt\_schema.sql} on both the active and the standby IMDB Cache nodes to create the schema for the BRM cache groups. You run \texttt{tt\_load.sql} only on the active node, and the Oracle IMDB Cache replication agent replicates the data to the standby node.

To drop the BRM cache groups schema and the tables created in the cache groups, run the \texttt{tt\_drop.sql} script. This will delete all the BRM objects from the cache. Data in the cache propagated to the BRM database prior to running \texttt{tt\_drop.sql} persists in the BRM database.

Creating and Initializing Your Cache Group Schema

To create and initialize your schema:

1. Configuring the \texttt{pin\_tt\_schema\_gen.values} File.
2. Generating Your Schema and Load SQL Scripts.

Configuring the pin_tt_schema_gen.values File

The pin_tt_schema_gen utility connects to the BRM database and uses the existing BRM database schema to generate the tt_schema.sql, tt_load.sql, and tt_drop.sql scripts based on the specifications in the pin_tt_schema_gen.values file.

When you configure the pin_tt_schema_gen.values file, you specify values such as:

- The database name, user name, and password for the BRM database
- The database number where to load the data
- The cache group definitions
- The cache group aging policy
- The transient tables

Table 21–1 describes the entries in the pin_tt_schema_gen.values file.

Table 21–1  pin_tt_schema_gen.values File Entries

<table>
<thead>
<tr>
<th>Entry</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAIN_DB['alias']</td>
<td>Specifies the database alias name for the BRM database defined in the tnsnames.ora file.</td>
</tr>
<tr>
<td>MAIN_DB['user']</td>
<td>Specifies the user name for the BRM database.</td>
</tr>
<tr>
<td>MAIN_DB['password']</td>
<td>Specifies the password for the BRM database user.</td>
</tr>
<tr>
<td>fileName</td>
<td>Specifies the name of the generated SQL file that can be used to create the schema definitions for the BRM cache groups. The default is tt_schema.sql.</td>
</tr>
<tr>
<td>droppingDatastoreSqlFile</td>
<td>Specifies the name of the generated SQL file that can be used to drop the schema definitions for the BRM cache groups. The default is tt_drop.sql.</td>
</tr>
<tr>
<td>loadingCachegroupSqlFile</td>
<td>Specifies the name of the generated SQL file that can be used to load the data into the BRM cache groups. The default is tt_load.sql.</td>
</tr>
<tr>
<td>logfile</td>
<td>Specifies the name of the log file that contains the error logs for the pin_tt_schema_gen utility. The default is pin_tt_schema_gen.log.</td>
</tr>
<tr>
<td>primary_db_user_name</td>
<td>Specifies the name of the primary database user. This is required if you are using a multischema configuration.</td>
</tr>
<tr>
<td>logical_partitioning_enabled</td>
<td>Specifies whether logical partitioning is enabled in IMDB Cache. The default is No.</td>
</tr>
<tr>
<td>cache_group_info</td>
<td>Specifies cache group information. See “Defining Cache Group Information” for more information.</td>
</tr>
<tr>
<td>cache_group_info_specific</td>
<td>Specifies information for special cases where you need to create the cache group definition for each table or class. For details on this entry, see &quot;Defining Cache Group Information for Special Cases&quot;.</td>
</tr>
</tbody>
</table>
Creating and Initializing Your Cache Group Schema

Creating and Initializing the BRM Cache Group Schema

The `pin_tt_schema_gen.values` file includes syntax information and specifications for the default BRM cache groups. Use the information in the following sections to supplement the information in `pin_tt_schema_gen.values`.

### Defining Cache Group Information

You define each cache group by using the `%cache_group_info` entry. This entry uses the following syntax:

```
%cache_group_info = {cache_group_name => {
  'CLASSES' => [class1, class2, ...... classN ],
  'CG_PROPERTIES' => [cache_group_type, baseTableName, optionalAging, primary_key,
    'USE_GLOBAL_CACHE_GROUP'],
  'FK_DEF' => ['DEFAULT' => foreignKey, referredBaseTable, fieldOfReferredBaseTable',
    'baseTableName_N' => foreignKey, referredBaseTable, fieldOfReferredBaseTable'] }, ....);
```

**Table 21–2** defines the parameters in the above syntax.

---

<table>
<thead>
<tr>
<th>Entry</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>on_delete_cascade_def</td>
<td>Specifies that when rows containing referenced key values are deleted from a parent table, rows in child tables with dependent foreign keys are also deleted. For details about this entry, see &quot;Setting On Delete Cascade&quot;.</td>
</tr>
<tr>
<td>lru_config_def</td>
<td>Specifies the least-recently-used (LRU) aging policy for the cache group. The default is (<code>'30', '50', '1'</code>). For details about this entry, see &quot;Defining the Cache Group Aging Policy&quot;.</td>
</tr>
<tr>
<td>local_tables_class_def</td>
<td>Specifies transient tables. For details about this entry, see &quot;Defining Transient Local Tables&quot;.</td>
</tr>
<tr>
<td>use_timesten_datatypes</td>
<td>Specifies whether you want to change the NUM field type to TT_INTEGER and TT_BIGINT. The default is Yes. For details about this entry, see &quot;Supporting IMDB Cache Data Types&quot;.</td>
</tr>
<tr>
<td>tt_load_cache_group_def</td>
<td>Specifies the cache groups in which to load data. For details about this entry, see &quot;Generating Load SQL&quot;.</td>
</tr>
<tr>
<td>db_no_for_load_sql</td>
<td>Specifies the logical partition database number. This entry can be used if your IMDB Cache has multiple logical partitions. This allows you to generate separate SQL files that can be used to load the data for each logical partition or database number. <strong>Note:</strong> You must set <code>logical_partitioning_enabled</code> to Yes to use this entry. For details about this entry, see &quot;Defining the Logical Partition Database Number&quot;.</td>
</tr>
<tr>
<td>specific_sql_query</td>
<td>Specifies additional SQL statements to be added in the generated schema definition SQL file.</td>
</tr>
</tbody>
</table>
Creating and Initializing Your Cache Group Schema

Defining Cache Group Information for Special Cases

You define cache group information for each table of class by using the `%cache_group_info_specific` entry. This entry uses the following syntax:

```
%cache_group_info_specific = { table_name => [ cache_group_name, primary_key, cache_group_type, 'AGING_OFF', 'USE_PRIMARY_DB', 'USE_GLOBAL_CACHE_GROUP' ], .......... }
```

Table 21–3 describes the parameters in the above syntax.

### Table 21–2 Parameters for the `%cache_group_info` Entry

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cache_group_name</td>
<td>Specifies the name of the cache group.</td>
</tr>
<tr>
<td>'CLASSES'</td>
<td>Specifies the BRM storable classes in the cache group, separated by commas. For example: 'CLASSES' =&gt; [ '/account', '/balance_group', '/bill', '/billinfo' ],</td>
</tr>
<tr>
<td>'CG_PROPERTIES'</td>
<td>Specifies the properties of the cache group where:</td>
</tr>
<tr>
<td></td>
<td>- cache_group_type specifies the cache group type.</td>
</tr>
<tr>
<td></td>
<td>- baseTableName specifies the root table in the cache group.</td>
</tr>
<tr>
<td></td>
<td>- (Optional) optionalAging is used to specify whether aging is disabled for the cache group. Aging is enabled by default.</td>
</tr>
<tr>
<td></td>
<td>- primaryKey specifies the root table primary key.</td>
</tr>
<tr>
<td></td>
<td>- USE_PRIMARY_DB</td>
</tr>
<tr>
<td></td>
<td>- USE_GLOBAL_CACHE_GROUP is specified when logical partitioning is enabled.</td>
</tr>
<tr>
<td></td>
<td>For example: 'CG_PROPERTIES' =&gt; [ 'ASYNCHRONOUS Writethrough' # Cache Group Type, 'ACCOUNT_T', # RootTable, 'POID_ID0', # Primary Key, 'USE_GLOBAL_CACHE_GROUP' ],</td>
</tr>
<tr>
<td>'FK_DEF'</td>
<td>(Optional) Used to define the foreign key for the subsequent base table of the subsequent class where:</td>
</tr>
<tr>
<td></td>
<td>- foreignKey</td>
</tr>
<tr>
<td></td>
<td>- referredBaseTable</td>
</tr>
<tr>
<td></td>
<td>- fieldOfReferredBaseTable</td>
</tr>
<tr>
<td></td>
<td>For example: 'FK_DEF' =&gt; [ 'DEFAULT =&gt; ACCOUNT_OBJ_ID0,ACCOUNT_T,POID_ID0', # Default for other root table of subsequent classes.</td>
</tr>
<tr>
<td></td>
<td>'JOURNAL_T =&gt; ITEM_OBJ_ID0,ITEM_T,POID_ID0'</td>
</tr>
</tbody>
</table>

### Table 21–3 Parameters for the `%cache_group_info` Entry

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table_name</td>
<td>Specifies the name of the cache table.</td>
</tr>
</tbody>
</table>

21-4  BRM System Administrator's Guide
Creating and Initializing Your Cache Group Schema

### Setting On Delete Cascade

You specify the table on which to apply ON DELETE CASCADE by using the `%on_delete_cascade_def` entry. This entry uses the following syntax:

```plaintext
%on_delete_cascade_def = ('cache_group_name' => ['table1','table2',...,'tableN'],................);
```

where:
- `cache_group_name` specifies the name of the cache group schema.
- `tableN` is the table name.

For example:

```
%on_delete_cascade_def = ('ACCOUNT_CG' => ['JOURNAL_T']);
```

### Defining the Cache Group Aging Policy

By default, IMDB Cache Manager checks the memory space usage of the IMDB Cache database at one-minute intervals. When the database memory usage exceeds 50%, IMDB Cache Manager purges the least-recently-used (LRU) objects from the cache groups until the memory space usage is below 30%.

You define the IMDB Cache database aging policy by using the `@lru_config_def` entry. This entry uses the following syntax:

```plaintext
@lru_config_def = ('LowUsageThreshold', 'HighUsageThreshold', 'AgingCycle');
```

where:
- `LowUsageThreshold` is the memory space usage in IMDB Cache below which the aging policy is deactivated. The default is .30.
- `HighUsageThreshold` is the memory space usage in IMDB Cache above which the aging policy is activated. The default is .50.
- `AgingCycle` is the frequency (in minutes) that the memory space usage is checked. The default is 1.

For detail information on implementing aging on a cache group, see Oracle In-Memory Database Cache User’s Guide.

---

### Table 21–3 (Cont.) Parameters for the `%cache_group_info` Entry

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>cache_group_name</code></td>
<td>Specifies the name of the cache group.</td>
</tr>
<tr>
<td><code>primary_key</code></td>
<td>The primary key for the specified table.</td>
</tr>
<tr>
<td><code>cache_group_type</code></td>
<td>Specifies the cache group type.</td>
</tr>
</tbody>
</table>
Defining Transient Local Tables

Transient tables are created as local tables in the Oracle IMDB Cache database. You define which storable classes to create as local tables in the IMDB Cache database by using the `@local_tables_class_def` entry. This entry uses the following syntax:

```
@local_tables_def = ('class1', 'class2', ...... , 'classN');
```

where `classN` is a BRM storable class.

The default entry is set to the following:

```
@local_tables_class_def = ('/reservation','/reservation/active','/reservation_list','/active_session','/active_session/telco','/active_session/telco/gprs','/active_session/telco/gprs/master','/active_session/telco/gprs/subsession','/active_session/telco/gsm');
```

Supporting IMDB Cache Data Types

You can reduce the amount of data stored in IMDB Cache by converting all BRM fields with `NUMBER(38)` data type to a `TT_INTEGER` or `TT_BIGINT` data type. The IMDB Cache integer data types `TT_INTEGER` and `TT_BIGINT` require less memory than the BRM integer data type `NUMBER(38)`.

- `NUMBER(38)` requires 22 bytes of storage.
- `TT_INTEGER` requires 4 bytes of storage.
- `TT_BIGINT` requires 8 bytes of storage.

To convert all fields with a `NUMBER(38)` data type to a `TT_INTEGER` or `TT_BIGINT` data type, set the `$use_timesten_datatypes` entry to YES:

```
$use_timesten_datatypes = "YES";
```

When this entry is set to `NO` or missing, fields with a `NUMBER(38)` data type are left unchanged.

Generating Load SQL

You define the load criteria for generating the `tt_load.sql` script by using the `%tt_load_cache_group_def` entry. This entry uses the following syntax:

```
%tt_load_cache_group_def = ('Cache_group_name', => 'no_of_rows_to_commit, condition, no_of_threads')
```

where:

- `Cache_group_name` is the name of the cache group.
- `no_of_rows_to_commit` is the number of rows to insert into the cache group before committing the work. It must be a non-negative integer. If `no_of_rows_to_commit` is 0, the entire statement is executed as one transaction.
- (Optional) `condition` is the search condition used to retrieve the target rows from the BRM database.
- (Optional) `no_of_threads` is the number of loading threads that run concurrently for parallel loading. One thread performs the bulk fetch from the BRM database and the remaining threads perform the inserts into the Oracle IMDB Cache database. Each thread uses its own connection or transaction.
The minimum value for `no_of_threads` is 2. The maximum value is 10. If you specify a value greater than 10, Oracle IMDB Cache database assumes the value 10.

For a high-availability system, you must configure `pin_tt_schema_gen.values` to generate `tt_load.sql` for the specific Oracle IMDB Cache database node and then run `tt_load.sql` against the respective node.

**Defining the Logical Partition Database Number**

You specify the logical partition database number on which to run the `tt_load.sql` script by using the `$dm_no_for_load_sql` entry. This entry uses the following syntax:

```
$db_no_for_load_sql = 'db_number';
```

where `db_number` is the logical partition database number.

For example, suppose there are two database schemas with two logical partitions for each schema as follows:

- Schema 1 has logical partitions 0.0.0.1 and 0.1.0.1
- Schema 2 has logical partitions 0.0.0.2 and 0.1.0.2

On the machine with logical partition 0.0.0.1, set `$db_no_for_load_sql` to "0.0.0.1". Then generate `tt_load.sql` using `pin_tt_schema_gen -l`.

On the machine with logical partition 0.1.0.1, set `db_no_for_load_sql` to "0.1.0.1". On the machine with logical partition 0.0.0.2, set `$db_no_for_load_sql` to "0.0.0.2". On the machine with logical partition 0.1.0.2, set `$db_no_for_load_sql` to "0.1.0.2".

---

**Note:** For logical partitioning support, set `logical_partitioning_enabled` to YES.

---

In a high-availability system with Oracle Clusterware, Oracle Clusterware automatically duplicates the complete data store from the active node to the standby node.

In a high-availability system without Oracle Clusterware, you must

- Run `tt_schema.sql` on both the active and the standby IMDB Cache nodes to create the schema for the BRM cache groups.
- Run `tt_load.sql` only on the active node. The Oracle IMDB Cache replication agent replicates the data to the standby node.

See "Using the IMDB Cache Grid to Partition Data" for more information about logical partitioning.

**Generating Your Schema and Load SQL Scripts**

You can generate schema and load SQL scripts for a basic BRM system or a BRM system with logical partitioning:

**Generating Scripts for A Basic BRM System**

To generate the schema and load SQL scripts:

1. If you have not already done so, modify the `pin_tt_schema_gen.values` file. See "Configuring the pin_tt_schema_gen.values File".
2. Run the following command:
Creating and Initializing Your Cache Group Schema

source BRM_Home/source.me.csh

3. Run the pin_tt_schema.gen utility with the -a parameter:
   ./pin_tt_schema_gen -a

---

**Note:** If you do not specify the values for MAIN_DB['user'] and MAIN_DB['password'] in the pin_tt_schema_gen.values file, the pin_tt_schema_gen utility prompts you to enter these values.

---

This updates the BRM database with unique indexes and non-null constraints and generates the following files:
- tt_schema.sql
- tt_load.sql
- tt_drop.sql

Generating Scripts for A BRM System with Logical Partitioning

To generate your scripts for logical partitions, perform the following steps for each database schema on your system:

1. Open the pin_tt_schema_gen.values file in a text editor.
2. Set the $logical_partitioning_enabled entry to Yes.
3. Set the $db_no_for_load_sql entry to the appropriate value for the first logical partition in your database schema.
4. Save and close the file.
5. Run the following command:
   ```bash
   source BRM_Home/source.me.csh
   ```
6. Run the pin_tt_schema_gen utility with the -a parameter:
   ```bash
   ./pin_tt_schema_gen -a
   ```
   This generates your tt_schema.sql, tt_drop.sql, and tt_load_0.M.0.N.sql scripts, where M specifies the partition number and N specifies the database schema number.
7. Open the pin_tt_schema_gen.values file in a text editor.
8. Set the $db_no_for_load_sql entry to the appropriate value for the second logical partition in your database schema.
9. Save and close the file.
10. Run the pin_tt_schema_gen utility with the -l parameter:
    ```bash
        ./pin_tt_schema_gen -l
    ```
    This generates the tt_load_0.M.0.N.sql script for loading BRM data into your second logical partition.
Customizing BRM for In-Memory Database Cache DM

This chapter provides information for customizing Oracle Communications Billing and Revenue Management (BRM) for In-Memory Database (IMDB) Cache Data Manager (DM).

About Customizing the Cache Groups

The following sections describe how to customize the default BRM cache groups and how to add custom cache groups to the BRM cache groups schema.

When you modify an existing cache group definition, you need to unload the data, drop the existing cache group, create the cache group with the new definition, and reload the data. Therefore, this operation requires service downtime. See Oracle In-Memory Database Cache User’s Guide for more information about cache group operations.

Customizing the Default BRM Cache Groups

You can modify the default BRM cache groups by adding custom storable objects or by modifying existing storable objects.

To customize the default BRM cache groups:

1. Create the custom storable class or modify the existing storable class. See "Creating Custom Storable Classes".

2. Edit the pin_tt_schema_gen.values file (BRM_Home/bin/pin_tt_schema_gen.values) and modify the existing cache group definitions as needed. See "Configuring the pin_tt_schema_gen.values File".

   **Note:** All custom storable classes for subscriber data must be added to the existing subscriber cache group. See "About Extending Tables in the Default BRM Cache Groups" for more information.

3. Run the pin_tt_schema_gen utility with the -t parameter to generate the tt_schema.sql file to include the custom tables and fields.

4. Using the TimesTen ttlsql, update the BRM cache groups schema in the IMDB Cache as follows:
   
   - Stop the Oracle IMDB Cache replication agent.
   - Stop the cache agent.
c. Drop the cache group schema.

d. Run the tt_schema.sql file to generate the schema with the new definitions.

e. Start the cache agent.

f. Start the replication agent.

5. Reload the cache group, if needed.

6. Stop and restart IMDB Cache DM.

**About Extending Tables in the Default BRM Cache Groups**

All custom storable classes for subscriber data must be added to the existing subscriber cache group definition to support resource sharing relationships. For example, if you define a custom service storable class such as `/service/voip`, the corresponding custom tables should be added in the subscriber cache group schema. Custom storable classes for other data types should be added to custom cache group definitions. See "Creating a Custom Cache Group".

All custom storable classes for transient objects should to be added to the TimesTen local tables.

**Creating a Custom Cache Group**

You create custom cache groups manually by defining the custom cache group information in the `pin_tt_schema_gen.values` file, generating the schema SQL file, and then creating the schema in Oracle IMDB Cache. You can create the custom cache group using existing BRM storable objects or custom storable objects.

To create custom cache groups in the Oracle IMDB Cache:

1. Create the custom storable class. See "Creating Custom Storable Classes".

2. Edit the `pin_tt_schema_gen.values` file (`BRM_Home/bin/pin_tt_schema_gen.values`) and add a new cache group definition. See "Creating and Initializing the BRM Cache Group Schema".

3. Run the `pin_tt_schema_gen` utility with the `-t` parameter to generate the tt_schema.sql script.

4. Using the TimesTen ttIsql tool, update the BRM cache groups schema in the Oracle IMDB Cache as follows:
   a. Stop the Oracle IMDB Cache replication agent.

b. Stop the cache agent.

c. Drop the cache group schema.

d. Run the tt_schema.sql file to generate the schema for the custom cache groups.

e. Start the cache agent.

f. Start the replication agent.

5. Load the cache group, if needed.

6. Stop and restart IMDB Cache DM.

See the Oracle In-Memory Database Cache User’s Guide for more information on the different types of cache groups and how to define them.
Creating Custom Storable Classes

Use Developer Center to create custom storable classes and to modify existing storable classes in the BRM database.

For specific instructions on how to create or modify storable classes, see “Creating custom fields and storable classes” in Developer’s Guide.

Assigning Custom Objects a Residency Value

When you define a custom storable class, you must also set the residency type for the class. The residency type specifies where the storable object resides in the BRM system. Object residency types are predefined in the BRM data dictionary. To determine the residency type value for your custom storable class, you can run a SQL query on the BRM database to get the residency type values. For example, to get all the objects with residency type values 1, 5, or 7, you can run the query:

```
Select name, residency_type from dd_objects_t where residency_type in (1,5,7);
```

To set the residency type value for the custom storable class, you can run a SQL query on the BRM database to set the value. For example, to set the value for a custom service class, you can run the query:

```
update dd_objects_t set residency_type=1 where name like '/service%' and residency_type !=5 and residency_type !=7;
```

After setting the residency type value for the custom storable class, restart IMDB Cache DM and Connection Manager.

About Extending Event Type Storable Classes

When you define a custom event storable class, it inherits the attributes of the parent event storable class. Event attributes can be:

- TimesTen resident expanded object
- BRM resident expanded object

When you extend the base event class (event), you can define the subclass by setting the residency type.
PIN_GET_SCHEMA_NO

This macro returns the database schema number from the database number provided in the parameter. A database number is represented in the format 0.logical_partition_number.0.database_schema_number. The PIN_GET_SCHEMA_NO macro extracts database_schema_number from the database number.

Syntax

PIN_GET_SCHEMA_NO (Database_number);

Parameters

*Database_number*
A 64-bit integer that represents a database number.

Return Values

Returns the database schema number.

Error Handling

This macro does not return any error.
Part VI describes how to partition the Oracle Communications Billing and Revenue Management (BRM) database, how to purge data, and how to generate virtual columns on event tables. It contains the following chapters:

- Partitioning Database Tables
- Changing from a Nonpartitioned to a Partitioned Database
- About Purging Data
- Generating Virtual Columns on Event Tables
Partitioning Database Tables

This document explains how to organize your Oracle Communications Billing and Revenue Management (BRM) database by using partitioned tables.

Before partitioning any tables, you should be familiar with the following:

- Basic BRM concepts. See "Introducing BRM" in BRM Concepts.
- BRM system architecture. See "BRM System Architecture" in BRM Concepts.
- Oracle documentation and Oracle database administration skills, including PL/SQL and SQL*Plus.
- Perl.
- UNIX commands and the UNIX operating system.
- The pin_setup.values file for configuring the BRM server. See "Editing the pin_setup.values File to Enable Partitioning for Non-Event Tables" in BRM Installation Guide.

About Partitioning

Partitioning is an optional feature that splits tables and indexes into smaller, more manageable units. When you no longer need the data stored in a partition, you can purge it from your database by deleting the partition (see "About Purging Data").

To use this feature, you need the following:

- An Oracle database. Partitioning is available only for Oracle databases.
- Oracle Partitioning. You license this third-party software from Oracle.

To enable partitioning for BRM, do one of the following:

- When installing BRM, choose to partition your database tables. See "Installing BRM" in BRM Installation Guide.

Note: You can partition non-event tables during the BRM installation process only. See "Editing the pin_setup.values File to Enable Partitioning for Non-Event Tables" in BRM Installation Guide.

- After installing BRM, upgrade your nonpartitioned database to a partitioned database. See "Changing from a Nonpartitioned to a Partitioned Database".

In either case, you should ensure that the SENABLE_PARTITION parameter is set to yes ($ENABLE_PARTITION = yes) in the pin_setup.values file. The above operations should set this parameter for you. In addition, the tables for the /event class
are automatically enabled when you run `pin_setup` if the `$ENABLE_PARTITION` parameter is set to `yes`.

If you wish to partition non-event tables, assign a list of the classes (not including `/event`) that you wish to partition to the `@CLASSES_TO_BE_PARTITIONED` parameter in the `pin_setup.values` file before you run the `pin_setup` utility.

For example:

```
@CLASSES_TO_BE_PARTITIONED = ("/journal:local","/account:local");
```

**Note:** You should only list base classes.

The :local and :global suffixes designate two types of indexes that are available for non-event partitions (all event object tables are local):

- **:global indexes**
  - Advantage: Search operations become fast.
  - Disadvantage: BRM services must be shut down before adding or dropping partitions because these indexes must be rebuilt.

- **:local indexes**
  - Advantage: Maintenance can be done without shutting down BRM.
  - Disadvantage: Performance may suffer. Search operations will be more time consuming because searches must hit every local index.

When you use partitioning, objects are stored in a partition according to criteria set when the partitions are created:

- Event objects that should never be purged (*non-purgeable events*) are stored in `partition_historic`. See "About the Default Partitioning Scheme".

- All other objects are stored in date-based, purgeable partitions according to the date they were created. For example, if a partition includes a date range from January 15 to February 15, all objects that have a creation date (PIN_FLD_CREATED_T) in that range are included.

  You create separate partitions for real-time events and delayed events. Real-time events are created by real-time rating; delayed events are created by pipeline batch rating. You can purge real-time events, delayed events, or both.

  **Note:** Delayed partitioning is not supported for non-event classes.

- If there is no date-based partition, objects are stored in `partition_last`. See "About Objects Stored in partition_last and partition_last_pin".

For partitioned event tables, you can use the default purgeable partitions that `pin_setup` creates. If you do not want to use the default partitions, you can create custom purgeable partitions with the `partition_utils` utility. For non-event tables, you must explicitly create custom purgeable partitions.

See "About Partitioning Schemes" and "Customizing Partition Limitations".
To create partitions and purge data, use the `partition_utils` utility. See "About Managing Partitions".

**Note:** Rule-based optimization is not available for partitioned tables and indexes. For more information, see "Using Rule-Based Optimization versus Cost-Based Optimization" in *BRM Installation Guide*.

---

### Partitioning Overview

The flow chart in Figure 23–1 provides an overview of how to set up and use partitioning. All of the steps before you customize or add partitions must be done when you install BRM.

**Important:** To enable partitioning when you install BRM, you configure and run `pin_setup`.

For an overview of partitioning enabled by upgrading, see “About Upgrading from a Nonpartitioned to a Partitioned Database”.

---
About Partitioning Schemes

Partitions are based on dates; your partitioning scheme determines how the
date-based time periods are divided. Partitions can be daily, weekly, or monthly. You
can specify multiples of each time period; for example, you can create partitions based
on five days or on three weeks.

Determining the appropriate partition size depends on many factors. Generally
speaking, partitions should not exceed 80,000,000 to 120,000,000 entries for standard
BRM configurations. For more specific information, see the Oracle documentation or
contact Oracle technical support.

You can set up partitioning schemes based on different time periods in advance; for
example, you can specify three monthly partitions followed by three weekly partitions.
However, every table of the same base class must have the same partitioning scheme;
you cannot create different partitions for different tables with the same base class.
About Partitioning Schemes

**Important:** When you add partitions, you define a start date for the first partition. The start date cannot be earlier than the day after tomorrow. For example, if the current date is January 1, the earliest start date for the new partition is January 3.

If you use pipeline batch rating, the Rated Event (RE) Loader loads delayed events into the BRM database. You can enable partitioning for delayed events and create partitions for the delayed events. See "About Partitions for Delayed Events".

**Note:** All indexes on event tables are local. When you partition an event table, the associated index is identically partitioned by Oracle.

### About Nonpurgeable Events

By default, certain classes of events cannot be purged from the database. All event objects belonging to these classes are automatically stored in `partition_historic`. You cannot remove this partition.

For the default list of events stored in `partition_historic`, see "Event Objects Stored in partition_historic".

To customize this list, see "Customizing the List of Events Stored in partition_historic".

### About Objects Stored in `partition_last` and `partition_last_pin`

The `partition_last` partition holds objects that occur after the time period covered by the regular partitions if no purgeable partitions are available for such objects. You cannot remove this partition for real-time event tables or non-event tables.

**Caution:**

- This partition is not intended to store objects; it is a temporary container for them. To keep objects out of these partitions, make sure there are partitions to hold new events. See "Adding Partitions".

- To purge objects stored in spillover partitions, stop all delayed-event loading and add partitions and use the `-f` (force) parameter. Adding partitions in this way moves the data from the spillover partitions to the added partitions, but it takes much longer than adding partitions normally.

If your partitions are enabled for delayed events, `partition_last` is used for delayed events; `partition_last_pin` is used for real-time events.

### About the Default Partitioning Scheme

If you enable partitioning when you install BRM and choose to create the default partitions, your real-time events are stored in the following partitions:

- `partition_historic` (event tables only)
- `partition_last`
- 12 monthly partitions
About Partitioning Schemes

- +1 partition for objects created on the start date

Each monthly partition stores objects created on the date of the month you install BRM through the previous date of the next month. For example, if you install BRM on January 15, the partitions cover these time periods: January 15-February 14, February 15-March 14, and so on.

**Note:** The naming convention for partitions is P_R_MMDDYYYY (real time) and P_D_MMDDYYYY (delayed) based on the start date.

If You Did Not Create Default Partitions

If you enable partitioning when you install BRM but choose not to create the default partitions, your real-time object tables are divided into fewer partitions: **partition_historic** (stores all non-purgeable events) and **partition_last** (stores all purgeable objects). This is sufficient for a test or development system in which purging by partition is not required. For a production system, however, you must add more purgeable partitions. See "Adding Partitions".

Upgrade Partitioning Scheme

If you enable partitioning by upgrading your database after you install BRM, your partitioned real-time tables are divided into three partitions:

- **partition_migrate**: Holds all event objects created before the nonpartitioning-to-partitioning upgrade.
- **partition_historic**: Holds nonpurgeable events created after the nonpartitioning-to-partitioning upgrade. Nonpurgeable events should not be purged from the database. See Event objects stored in **partition_historic** (Oracle only).
- **partition_last**: A spillover partition that is not intended to store event objects you want to purge or preserve. If you do not add purgeable partitions to your event tables before BRM resumes generating events, purgeable events created after the upgrade are stored in this partition.

About Partitions for Delayed Events

To create partitions for delayed events, you use the **partition_utils** utility to do the following:

- Enable delayed-event partitions. You specify which events you want partitions for.
- Add partitions for delayed events. You add delayed partitions the same way you add real-time partitions.

**Note:** When you enable partitions, the **partition_utils** utility automatically runs the **update** operation. This aligns the delayed partitions with any partitions already defined in the EVENT_T table. Therefore, you do not need to add partitions for delayed events after enabling them.

When you add partitions for delayed events, the event tables are divided into the default partitions plus the following partitions:
- **Partitions for delayed events**: The time periods covered by these partitions do not need to match the periods covered by the partitions for real-time event objects. For example, you can use daily partitions for real-time events and weekly partitions for delayed events.

- **partition_last_pin**: When delayed-event partitions are added to tables, this partition is added for real-time event objects; **partition_last** becomes the spillover partition for delayed events for which there is no other partition.

  **Caution**: Spillover partitions are not intended to store event objects you want to purge or preserve. To keep objects out of these partitions, you need to add partitions for upcoming dates (see "Adding Partitions"). To purge events stored in spillover partitions, you must add partitions and use the -f (force) parameter. When you do that, you must stop all BRM processes and stop all delayed-event loading. Adding partitions moves the data from the spillover partitions, but it takes much longer than normally adding partitions.

Delayed-event storage is based on the date that the delayed events are loaded into the BRM database.

If you enable delayed-event partitions for an event class, but then decide you do not need partitions for that delayed-event class, you can use `sqlplus` to remove delayed-event partitions for that event table, then remove the **partition_last_pin** partition for that event table. See "Disabling Delayed-Event Partitioning".

**Overview of Partition Schemes**

The following tables show the possible partition schemes.

**Table 23–1  No Partitions**

<table>
<thead>
<tr>
<th>Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object table</td>
</tr>
</tbody>
</table>

If you do not enable partitions when installing BRM, every object table has no partitions as shown in Table 23–1.

**Table 23–2  Real-Time Partitioning Enabled, No Default Partitions**

<table>
<thead>
<tr>
<th>Partition</th>
<th>Partition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partition_historic</td>
<td>partition_last</td>
</tr>
<tr>
<td>(event tables only)</td>
<td></td>
</tr>
</tbody>
</table>

An object table designated for partitioning is partitioned as shown in Table 23–2 if you enable partitions but do not create the 12 default partitions.

**Caution**: Do not use the partition scheme shown in Table 23–2 in a production system because all non-historic real-time objects are stored in **partition_last**. You need to add real-time partitions.
An object table designated for partitioning is partitioned as shown in Table 23–3 if you enable partitions and accept the 12 default partitions.

### Table 23–3  Real-Time Partitioning Enabled, 12 Default Partitions

<table>
<thead>
<tr>
<th>Partition</th>
<th>Partition</th>
<th>Partition</th>
<th>Partition</th>
<th>Partition</th>
<th>Partition</th>
<th>Partition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partition_historic (event tables only)</td>
<td>Real Time Partition 1</td>
<td>Real Time Partition 2</td>
<td>…</td>
<td>Real Time Partition 11</td>
<td>Real Time Partition 12</td>
<td>partition_last</td>
</tr>
</tbody>
</table>

An object table designated for partitioning is partitioned as shown in Table 23–4 if you enable partitions and create custom partitions.

### Table 23–4  Real-Time Partitioning Enabled, Custom Default Partitions

<table>
<thead>
<tr>
<th>Partition</th>
<th>Partition</th>
<th>Partition</th>
<th>Partition</th>
<th>Partition</th>
<th>Partition</th>
<th>Partition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partition_historic (event tables only)</td>
<td>Real Time Partition 1</td>
<td>Real Time Partition 2</td>
<td>Real Time Partition 3</td>
<td>…</td>
<td>Real Time Partition N</td>
<td>partition_last</td>
</tr>
</tbody>
</table>

An object table designated for partitioning is partitioned as shown in Table 23–5 if you enable partitions without the 12 default partitions and then enable delayed-event partitions.

### Table 23–5  Real-Time Partitioning Enabled, Delayed Partitioning Enabled, No Default Partitions

<table>
<thead>
<tr>
<th>Partition</th>
<th>Partition</th>
<th>Partition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partition_historic</td>
<td>partition_last_pin</td>
<td>partition_last</td>
</tr>
</tbody>
</table>

The event table is partitioned as shown in Table 23–5 if you enable partitions without the 12 default partitions and then enable delayed-event partitions.

---

**Caution:** Do not use this partition scheme shown in Table 23–5 in a production system because all non-historic events are stored in partition_last and partition_last_pin. You need to add real-time and delayed partitions.

The event table is partitioned as shown in Table 23–6 if you enable partitions, create real-time partitions, and enable delayed-event partitions.

### Table 23–6  Real-Time Partitioning Enabled, Real-Time Partitions Exist, Delayed Partitioning Enabled

<table>
<thead>
<tr>
<th>Partition</th>
<th>Partition</th>
<th>Partition</th>
<th>Partition</th>
<th>Partition</th>
<th>Partition</th>
<th>Partition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partition_historic</td>
<td>RT P_1</td>
<td>RT P_2</td>
<td>RT P_3</td>
<td>…</td>
<td>RT P_N</td>
<td>partition_last_pin</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>partition_last</td>
</tr>
</tbody>
</table>

The event table is partitioned as shown in Table 23–6 if you enable partitions, create real-time partitions, and enable delayed-event partitions.

---

**Caution:** Do not use this partition scheme in a production system because all non-historic delayed events are stored in partition_last. You need to add delayed partitions.
About Partitioning Schemes

The event table is partitioned as shown in Table 23–7 if you enable partitions, do not create real-time partitions, and enable and create delayed partitions.

Caution: Do not use this partition scheme shown in Table 23–7 in a production system because all non-historic real-time events are stored in \texttt{partition\_last\_pin}. You need to add real-time partitions.

Table 23–8 Real-Time Partitioning Enabled, Real-Time Partitions Exist, Delayed Partitioning Enabled, Delayed Partitions Exits

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{partition_historic}</td>
<td>RT P_1</td>
<td>RT P_2</td>
<td>...</td>
<td>RT P_N</td>
<td>\texttt{partition_last_pin}</td>
<td>D P_1</td>
<td>D P_2</td>
<td>...</td>
<td>D P_N</td>
</tr>
</tbody>
</table>

The event table is partitioned as shown in Table 23–8 if you enable partitions, create real-time partitions, and enable and create delayed partitions.

About Managing Partitions

To manage partitions, you perform these tasks:

- Customize partitions before you use a production system. See "Adding Partitions".
- Enable delayed partitions. You can specify which delayed events you want to create partitions for. See "Enabling Delayed-Event Partitions" and "Adding Partitions".
- Update partitions when you customize classes with partitioned database tables, or add classes by adding components. See "Updating Partitions".
- Purge objects by removing partitions. See "Purging Objects by Removing Partitions".
- Purge events by purging old events without removing partitions. See "Purging Event Objects without Removing Partitions".
- Add real-time partitions or delayed-event partitions to store new objects. See "Adding Partitions".
- Customize the events stored in \texttt{partition\_historic}. See "Customizing the List of Events Stored in partition\_historic".

Before your BRM production system begins generating objects that you wish to store in partitions, you should add partitions to your tables if any of these situations is true:

- You removed the 12 default monthly event partitions created during installation (event tables only).
- You chose not to create the 12 default monthly event partitions when you enabled partitioning during installation.
- You partitioned your tables by upgrading your database.
You enabled partitioning for any non-event tables.

In addition, after your partitioned production system is up and running, you should add purgeable partitions whenever the time periods covered by the existing purgeable partitions are about to pass.

---

**Caution:**

- The `partition_last` partition is not intended to store objects; it is a temporary container for them. To keep objects out of these partitions, you need to add partitions for upcoming dates (see "Adding Partitions").

- To purge objects stored in spillover partitions, stop all delayed-event loading and add partitions and use the `-f` (force) parameter. Adding partitions in this way moves the data from the spillover partitions to the added partitions, but it takes much longer than adding partitions normally.

---

**About Purging Objects**

When you purge objects, you can either:

- Remove the partitions that contain objects.
- Purge event objects, but maintain the event partitions.

**About Removing Partitions to Purge Objects**

When you purge objects by removing their partitions, you specify a start date and an end date to identify the objects to be removed. For example, you can purge objects created from January 15 to March 15.

---

**Important:** The dates you specify for purging objects might not align with the partition dates. If this is the case, only partitions that are completely within the date range are removed.

If a partition included in the start and end date includes an object associated with an open item, the partition is not removed.

---

Figure 23–2 shows monthly partitions, each starting on the first of the month. If you specify to purge objects created between January 15 and April 15, only objects from February 1 through April 1 are purged, because those are the only objects in the date range within complete partitions.
**About Running the partition_utils Utility**

Most partition management tasks are accomplished by using the `partition_utils` utility.

---

**Important:** Purging partitions is considered safer than removing them. For more information, see "Purging Event Objects without Removing Partitions".

**Tip:** Before purging objects, run the `partition_utils` utility with the `-p` parameter to write an SQL statement that shows the partitions that will be removed. See "Running the partition_utils Utility in Test Mode".

---

**About Purging Events While Maintaining Partitions**

You can delete old events without deleting partitions by specifying the last event date you want to keep. If any old events are associated with active items, those particular events are not deleted. See "Purging Event Objects without Removing Partitions".

---

**Tip:** Before purging objects, run the `partition_utils` utility with the `-p` parameter to write an SQL statement that shows the partitions that will be removed. See "Running the partition_utils Utility in Test Mode".

---

**Important:** Purging partitions is considered safer than removing them.

---

**About Running the partition_utils Utility**

Most partition management tasks are accomplished by using the `partition_utils` utility.

---

**Caution:** After you use your most current version of the `partition_utils` utility, do not use the previous versions. Doing so can corrupt your database. (However, partitions created by using the previous version are supported.)
You can run only one instance of this utility at a time. If you try to run more than one instance, the utility does not run and returns an error.

---

**Caution:** After you start the utility, do not interrupt it. It might take several minutes to complete, depending on the size of your database. If the utility is interrupted for any reason, you should use the `partition_utils restart` operation to continue the previous operation. See "Restarting partition_utils".

---

The `partition_utils` utility creates the following tables. Do not delete them or use the table names for any other purpose:

- PIN_TEMP_DD_INFO_T
- PIN_TEMP_TBL_NAMES_T
- PIN_TEMP_OUT_OF_SYNC_T
- PIN_TEMP_TAB_PARTITIONS_T
- PIN_TEMP_HIGH_VAL_T
- PIN_TEMP_DETAILS_T
- PIN_TEMP_PURGE_STATISTICS_T
- PIN_TEMP_PURGE_PARTITIONS_T
- PIN_TEMP_PURGE_LOGIC_T
- PIN_TEMP_PURGE_POIDS_T

**Running the partition_utils Utility in Test Mode**

When you add, remove, or enable partitions, you can use the `-p` parameter to write an SQL statement of the operation to the `partition_utils.log` file without performing any action on the database. The `partition_utils.log` file is in the same directory as the `partition_utils` utility.

---

**Caution:** Do not copy the SQL statements and run them against the database. This action is not supported.

---

The naming convention for real-time partitions is:

P_R_MMDDYYYY

For example, the following partition name specifies that the last date used for objects is June 29, 2004.

P_R_06292004

The naming convention for delayed-event partitions is:

P_D_MMDDYYYY

For example, the following partition name specifies that the last date used for objects is June 29, 2004.

P_D_06292004
Configuring a Database Connection

Before you use the partition_utils utility, configure the database connection parameters in the partition_utils.values file in BRM_Home/apps/partition_utils. For example:

```plaintext
$MAIN_DB{'vendor'} = "oracle";
$MAIN_DB{'alias'} = "pindb.myserver.com";
$MAIN_DB{'user'} = "pin";
$MAIN_DB{'password'} = "pin";
```

For more information, see the comments in the partition_utils file.

Improving Performance When Using partition_utils

For all partition_utils operations, you can run processes in parallel to improve performance. Edit the NUM_OF_PROCESSES parameter in the partition_utils.values file in BRM_Home/apps/partition_utils to run processes in parallel. This parameter controls how many partition_utils processes may run in parallel, except when executing the enable operation.

The valid range for NUM_OF_PROCESSES is from 1 to 10. The default is 2.

Restarting partition_utils

If partition_utils is interrupted for any reason, it is best to use the partition_utils restart operation to continue the last operation. This will prevent your database partitions from becoming unaligned and corrupted.

The syntax is:

```plaintext
partition_utils -o restart [-b]
```

Use the -b (bypass) parameter to bypass the last operation and clean the status of it.

For more information, see partition_utils.

Adding Partitions

To add partitions, run the partition_utils utility with the add operation.

---

**Important:**

- Partitioning must be enabled in your system before you can add partitions. If partitioning isn’t enabled, see "Changing from a Nonpartitioned to a Partitioned Database".
- Delayed-event partitions must be enabled before you can add delayed-event partitions. See "Enabling Delayed-Event Partitions".
- To add a partition for a non-event class, partitioning must have been enabled for that class before pin_setup was run during the BRM installation process. For more information, see "Editing the pin_setup.values File to Enable Partitioning for Non-Event Tables" in BRM Installation Guide.
- When adding partitions, an additional partition for the time up to start_date will be created by default if no partition for that time exists.
The syntax is:

```
partition_utils -o add -t realtime|delayed -s start_date
    -u month|week|day -q quantity [-c storable_class] [-w width]
    [-f] [-p]
```

- Use the `-t` (type) parameter to add real-time or delayed partitions.

  **Note:** Non-event partitions are always real-time. You may only add delayed-event object partitions. To do so, you must enable partitioning for each event type you want to partition. See "Enabling Delayed-Event Partitions".

- Use the `-s` (start date) parameter to specify the start date for the first of the new partitions. The format is `MMDDYYYY`.

  The start date must be the day after tomorrow or later; you cannot create partitions starting on the current day or the next day. For example, if the current date is January 1, the earliest start date for the new partition is January 3.

  **Note:** When adding partitions, an additional partition for the time up to `start_date` will be created by default if no partition for that time exists.

By default, the start date must also be earlier than six months from today. You can change these defaults by editing the `partition_utils.values` file in `BRM_Home/apps/partition_utils`. See "Customizing Partition Limitations".

If you try to create a partition with a start date within the current partition, the utility reports an error and provides the earliest date that you can use for the new partition.

  **Note:** To override this limitation, use the `-f` parameter.

- Use the `-u` (unit) parameter to specify the time unit for the partition.

- Use the `-q` (quantity) parameter to specify the number of partitions to add. Enter an integer greater than 0.

  By default, you can add the following numbers of partitions:
  - No more than 60 daily partitions
  - No more than 15 weekly partitions
  - No more than 6 monthly partitions

  You can change these defaults by editing the `partition_utils.values` file in `BRM_Home/apps/partition_utils`. See "Customizing Partition Limitations".

- Use the `-c` (storable class) parameter to specify the class of objects to be stored in the partition.

- Use the `-w` (width) parameter to specify the number of units in a partition; for example, a partition that is 3 days wide, 2 weeks wide, or 1 month wide. Enter an integer greater than 0. The default is 1.

  By default, the maximum widths you can use are:
Adding Partitions

Partitioning Database Tables

You can change these defaults by editing the **partition_utils.values** file in `BRM_Home/apps/partition_utils`. See "Customizing Partition Limitations".

The default storable base class for **partition_utils** is the `/event` base class.

- Use the `-f` (force) parameter to create a partition with a start date that falls within the time period of an active partition. Before you use this parameter:
  - For real-time partitions, stop all BRM server processes before adding partitions.
  - For delayed-event partitions, stop all delayed-event loading before adding partitions.

**Important:** If you use the `-f` parameter to create partitions within the time period of the current partition, and you do not stop BRM processes or loading, the BRM server might write data to a partition that is in the process of being divided. This can result in a server error, such as a write operation error.

**Note:** The `-f` parameter works differently when you remove partitions. In that case, it forces the removal of objects associated with open items.

- Use the `-p` (print) parameter to run the utility in test mode and generate an SQL statement. See "Running the partition_utils Utility in Test Mode".

For more information, see "partition_utils".

The following examples show how to add partitions.

**Note:** Adding partitions will add the requested number of partitions along with an additional partition on the given start date unless a partition labeled with that date exists already.

The following command adds five real-time event partitions, starting on July 27, 2004, and one partition for events up to July 27, 2004. Each partition includes events for a two-week period:

```
partition_utils -o add -t realtime -s 07272004 -u week -w 2 -q 5
```

The following command adds the `-f` parameter. This creates partitions even if they include dates in the current partition:

```
partition_utils -o add -t realtime -s 07272004 -u week -w 2 -q 5 -f
```

The following command adds six daily event partitions starting on July 27, 2004, and one partition for events up to July 27, 2004. Because the `-w` parameter is not used, each parameter is one day long:

```
partition_utils -o add -t delayed -s 07272004 -u day -q 6
```
Enabling Delayed-Event Partitions

The following command adds 12 monthly real-time partitions for /journal class starting on July 27, 2004, and one partition for /journal up to July 27, 2004:

```
partition_utils -c /journal -o add -t realtime -s 07272004 -u month -q 12
```

The following command creates five weekly partitions for /journal class starting on July 27, 2004, and one partition for /journal up to July 27, 2004:

```
partition_utils -c /journal -o add -t realtime -s 07272004 -u week -q 5
```

Enabling Delayed-Event Partitions

To add delayed-event partitions, run the `partition_utils` utility with the `enable` operation. You must enable delayed-event partitioning before you can add delayed-event partitions.

**Important:** After you enable delayed-event partitions, add partitions for delayed events before you use the system for production. Otherwise, delayed events are stored in `partition_last`. See “About Objects Stored in partition_last and partition_last_pin”.

**Note:** You can enable delayed partitions only for event objects. All non-event partitions are real-time partitions.

The syntax is:

```
partition_utils -o enable -t delayed [-c storable_class] [-p]
```

- Use the `-c` (storable class) parameter to specify the event class to partition. Because delayed-event partitions cannot be used for other classes, this specific version of `-c` does not specify a different base class.
- Use the `-p` (print) parameter to run the utility in test mode and generate an SQL statement. See "Running the partition_utils Utility in Test Mode".

**Note:** When you enable delayed partitions, the `partition_utils` utility automatically runs the `update` operation to synchronize partitioning schemes across all tables. See "Updating Partitions".

For more information, see "partition_utils".

The following command enables delayed-event partitioning for the /event/delayed/session/telco/gsm class:

```
partition_utils -o enable -t delayed -c /event/delayed/session/telco/gsm
```

You can enable partitioning for all subclasses of an event by using the percent sign (%) as a wildcard:

```
partition_utils -o enable -t delayed -c /event/session/
```

**Note:** This operation supports only delayed events. If you specify anything other than `-t delayed`, the utility returns an error.
Disabling Delayed-Event Partitioning

If you enable delayed-event partitions for an event class, but then decide you do not need partitions for that delayed-event class, you can use `sqlplus` to remove delayed-event partitions for that event table and then remove the `partition_last` partition for that event table.

---

**Important:** If you remove partitions for a delayed-event table, remove all the delayed partitions before removing the `partition_last` partition.

---

For example, to disable partitioning for the `EVENT_DLYD_SESSION_TELCO_GSM_T` table, do the following:

1. Use this `sqlplus` statement to ensure that there is no data in `partition_last`:
   ```sql
   SQL> select count(*) from event_dlyd_session_telco_gsm_t partition PARTITION_LAST
   ```
   If there is data in `partition_last`, add partitions to move the data.

   **Caution:** You must use the `-f` parameter. See "Adding Partitions".

2. Use this `sqlplus` statement to list the delayed partitions in the table:
   ```sql
   SQL> select partition_name from user_tab_partitions where table_name = UPPER(event_dlyd_session_telco_gsm_t) and partition_name like 'P_D_%';
   ```

3. Use this `sqlplus` statement to remove the listed partitions individually:
   ```sql
   SQL> alter table event_dlyd_session_telco_gsm_t drop partition partition_name
   ```

4. Use this `sqlplus` statement to remove `partition_last`:
   ```sql
   SQL> alter table event_dlyd_session_telco_gsm_t drop partition PARTITION_LAST
   ```

5. Use this `sqlplus` statement to verify that there are no partitions in the table:
   ```sql
   SQL> select count(*) from user_tab_partitions where table_name = UPPER(event_dlyd_session_telco_gsm_t) and partition_name like 'P_D_%';
   ```
   This should return zero rows.

Updating Partitions

The `update` operation allows you to align table partitioning schemes for all subclass tables with the current partitioning scheme for their base class table. Updating partitions enforces these rules:

- From the day after tomorrow, all tables with real-time objects will have the same real-time partitioning scheme as their base table (EVENT_T for /event base class tables, ITEM_T for /item base class tables, etc.).
- From the day after tomorrow, all tables with delayed events will have the same delayed-event partitioning scheme as the EVENT_T table.

You need to update partitions in the following cases:

- When you add custom partitioned classes.
Purging Objects by Removing Partitions

- When you extend an existing class with an array or substruct that adds a partitioned table.
- When you add a component that includes new partitioned classes.

**Note:** The update operation runs once automatically when you enable delayed partitions.

To align the partition schemes of new or changed object tables with the partition schemes of their base classes, run the `partition_utils` utility with the **update** operation.

The syntax is:

```
partition_utils -o update [-c storable_class] [-p]
```

- Use the `-c` (storable class) parameter to specify the class of objects to be updated.
- Use the `-p` (print) parameter to run the utility in test mode and generate an SQL statement. See "Running the `partition_utils` Utility in Test Mode".

For more information, see "partition_utils".

---

### Purging Objects by Removing Partitions

The general way to purge partitioned data is to remove one or more partitions. This frees up database space. By default, partitions that contain objects associated with open items are not removed when you remove partitions.

**Caution:** You can use the `-f` (force) parameter to remove partitions even if the objects are associated with open items, but this parameter must be used with care.

**Important:** Before removing partitions that contain objects associated with open items, verify that this does not contradict your business practices.

**Tip:** Before removing partitions, run the `partition_utils` utility with the `-p` parameter to write an SQL statement that shows the partitions that will be removed. See "Running the `partition_utils` Utility in Test Mode".

To remove partitions, run the `partitions_utils` utility with the **remove** operation. The syntax is:

```
partition_utils -o remove -s start_date -e end_date [-c storable_class] [-t realtime|delayed] [-f] [-p]
```

- Use the `-s` (start date) parameter to specify the start of the date range for the partitions you want to remove. The format is `MMDDYYYY`.
- Use the `-e` (end date) parameter to specify the end of the date range for the objects you want to purge. The format is `MMDDYYYY`.
All partitions that are entirely within these dates are removed. See "About Purging Objects".

By default, you can use this operation to remove only those partitions that are older than 45 days. You can change this limitation by editing the partition_utils.values file in BRM_Home/apps/partition_utils. See "Customizing Partition Limitations".

- Use the -c (storable base class) parameter to specify the objects you are purging by base class. The default is /event.

---

**Important:** To remove a non-event table partition, you must use the -f parameter. Operations using this parameter cannot be undone and will remove objects that are being used. Use with caution.

---

- Use the -t (type) parameter to remove only real-time events or delayed events for delayed-event partitions. The default is to remove both real-time and delayed objects.

- Use the -f parameter to remove partitions even if they contain objects associated with open items. By default, partitions that contain events associated with open items are not removed.

---

**Note:** The -f parameter works differently when you add partitions. In that case, it forces the splitting of partitions even when they fall within the date range of the current partition.

---

- Use the -p (print) parameter to run the utility in test mode and generate an SQL statement. See "Running the partition_utils Utility in Test Mode".

For more information, see "partition_utils".

The following command removes the partitions with real-time events from July 20, 2004, to September 20, 2004:

```
partition_utils -o remove -s 07202004 -e 09202004 -t realtime
```

This command removes real-time partitions for item table entries from July 20, 2004, to September 20, 2004:

```
partition_utils -o remove -s 07202004 -e 09202004 -c /item -t realtime -f
```

**Purging Event Objects without Removing Partitions**

You may purge event partitions without removing the partitions.

---

**Tip:** Before purging events, run the partition_utils utility with the -p parameter to write an SQL statement that shows the partitions that will be purged. See "Running the partition_utils Utility in Test Mode".

---

To purge events without removing partitions, run the partitions_utils utility with the purge operation. The syntax is:

```
partition_utils -o purge -e end_date [-t realtime|delayed] [-p]
```
Finding the Maximum POID for a Date

Use the `-e` (end date) parameter to specify the last date for the events you want to keep. The format is `MMDDYYYY`. All events older than that date that are associated with closed items are deleted. See "About Purging Objects".

Use the `-t` (type) parameter to purge only real-time or delayed events. The default is to purge both real-time and delayed events.

Use the `-p` (print) parameter to run the utility in test mode and generate an SQL statement. See "Running the partition_utils Utility in Test Mode".

For more information, see "partition_utils".

The following command purges real-time events before July 20, 2004:

```
partition_utils -o purge -e 07202004 -t realtime
```

Finding the Maximum POID for a Date

You can use `partition_utils -o maxpoid` to find the maximum POID in a partition for a given date. You might need this if you have scripts to automatically manage partitions.

The syntax is:

```
partition_utils -o maxpoid -s date -t realtime|delayed
```

The following command finds the maximum POID for a real-time partition:

```
partition_utils -o maxpoid -s 02012005 -t realtime
```

For more information, see "partition_utils".

Customizing Partition Limitations

The `partition_utils` utility specifies default limitations to prevent the creation of an unrealistic number of partitions. You can change these limitations by editing the `partition_utils.values` file in `BRM_Home/apps/partition_utils`.

The default limitation values are:

- Partition start date: later than tomorrow, earlier than 6 months from now.
- Maximum number of partitions:
  - 60 daily
  - 15 weekly
  - 6 monthly
- Maximum widths:
  - 5 days
  - 3 weeks
  - 2 months
- Removed partitions must be at least 45 days old.
- Minimum required percent of purgeable data.
- Partitions will be purged only when purgeable POIDs are greater than 70% or the total purgeable POIDs in the EVENT_BAL_IMPACT_T table.

The valid range for this setting is from 60 to 100.
Records are deleted only if there is a specified number in a chunk. The default is 1000 records.

The valid range is from 500 to 5000.

When purging, the DELETE_IN_PLACE method is used if the number of purgeable events are greater than 5% of the total events, or if there are more than 10,000 purgeable records.

You can change the number of purgeable records required. The valid range is from 1000 to 20000.

**Customizing the List of Events Stored in partition_historic**

If you do not need to save all the event types stored by default in partition_historic or if you need to save additional classes of events, use the Storable Class Editor in Developer Center to customize the list of nonpurgeable event classes.

Use the **Non-purgeable** class property to specify if an event is purgeable.

---

**Caution:** To avoid storing any event objects you do not want to purge in purgeable partitions, you must customize event classes stored in partition_historic before running BRM and generating any events.

---

For more information, see "About Purging Data".

---

**Note:** The **Non-purgeable** property for /event classes has no effect unless your event tables are partitioned.
Changing from a Nonpartitioned to a Partitioned Database

If you did not enable event table partitioning when you installed Oracle Communications Billing and Revenue Management (BRM), follow the procedures in this document to upgrade from a nonpartitioned BRM database to a partitioned BRM database.

**Note:** Partitioning is available for Oracle databases only.

This document covers Oracle on UNIX (HP-UX, AIX, Linux, and Solaris) platform.

Before partitioning your event tables, read the following:

- Partitioning Database Tables
- About Purging Data

**About Upgrading from a Nonpartitioned to a Partitioned Database**

The nonpartitioning-to-partitioning upgrade splits event tables in an Oracle database into the following partitions:

- **partition_migrate**: Holds all event objects created before the nonpartitioning-to-partitioning upgrade.

- **partition_historic**: Holds nonpurgeable events created after the nonpartitioning-to-partitioning upgrade. Nonpurgeable events should not be purged from the database. See "Event Objects Stored in partition_historic".

- **partition_last**: A spillover partition that is not intended to store event objects you want to purge or preserve. If you do not add purgeable partitions to your event tables before BRM resumes generating events, purgeable events created after the upgrade are stored in this partition.

**Note:** For information on how partitioning is enabled when you install BRM, see "Partitioning Overview".

**Software Required for Event Table Partitioning**

Before performing the nonpartitioning-to-partitioning upgrade, you must install:
■ Third-Party software, which includes the PERL libraries and JRE required for installing BRM components. See "Installing the Third-Party Software" in BRM Installation Guide.

■ BRM.

■ Oracle or Oracle 9i Release 2 with Oracle Partitioning.

■ Perl 5.004_01.

Upgrading to a Partitioned Database

To upgrade from a nonpartitioned database to a partitioned database, perform these tasks:

1. Increasing Disk Space for Event Tables and Indexes
2. Installing the Upgrade Package
3. (Optional) Reconfiguring the Parameters in partition.cfg
4. Merging the pin_setup.values File
5. Backing Up Your BRM Database
6. Running the Upgrade Scripts
7. Adding Purgeable Partitions to Event Tables
8. Restarting BRM

Increasing Disk Space for Event Tables and Indexes

Before adding partitions to your event tables, you must increase the disk space allocated for the event tables and indexes in your BRM database.

Tip: Oracle recommends that you add enough space for 6 to 12 months of event data.

Installing the Upgrade Package

To install the nonpartitioning-to-partitioning upgrade scripts and files, follow the procedure:

Note: If you have already installed the upgrade package, features in that package which are already installed cannot be reinstalled without first uninstalling them. To reinstall a feature, uninstall it and then install it again.

To install the upgrade package:

1. Download the software to a temporary directory (temp_dir).
2. Make sure no users are logged on to BRM.
3. Go to the directory where the Third-Party package is installed and source the `source.me` file.

   **Caution:** You must source the `source.me` file to proceed with installation, otherwise “suitable JVM not found” and other error messages appear.

   Bash shell:
   ```bash
   source source.me.sh
   ```

   C shell:
   ```csh
   source source.me.csh
   ```

4. Go to `temp_dir` and enter this command:
   ```
   7.4_PartitionUpg_platform_32_opt.bin
   ```

   **Note:** You can use the `-console` parameter to run the installation in command-line mode. To use the graphical user interface (GUI) for installation, make sure you have X Windows installed in your computer.

5. Follow the instructions displayed during installation. The default installation directory for the upgrade package is `opt/portal/7.4`.

   **Note:** The installation program does not prompt you for the installation directory if BRM or the upgrade package is already installed on the machine and automatically installs the package at the `BRM_Home` location.

6. Go to the directory where you installed the upgrade package and source the `source.me` file:

   Bash shell:
   ```bash
   source source.me.sh
   ```

   C shell:
   ```csh
   source source.me.csh
   ```
(Optional) Reconfiguring the Parameters in partition.cfg

The nonpartitioning-to-partitioning upgrade configuration file, partition.cfg, controls the parameters of your upgrade. If necessary, change these parameters to meet your business requirements.

To reconfigure the parameters in the partition.cfg file:

1. Log in as user pin, go to BRM_Home/upgrade/partition, and open the partition.cfg file in a text editor such as vi:

   ```
   % su - pin
   % cd BRM_Home/upgrade/partition
   % vi partition.cfg
   ```

2. Change the default parameters as necessary. For information on each parameter, see the comments in the partition.cfg file.

Merging the pin_setup.values File

Copy any customizations from your backed-up pin_setup.values file into the BRM_Home/setup/pin_setup.values file.

Backing Up Your BRM Database

Make a complete offline backup of your BRM database, and make sure the backup is completely valid and usable. For more information on performing full database backups, see your database software documentation.

In addition to the backup, use the Oracle export utility to export all BRM tables. This helps you restore individual tables if necessary.

Running the Upgrade Scripts

Upgrading from a nonpartitioned database to a partitioned database should take about 30 minutes. The size of your event tables does not affect the speed of this upgrade.

To partition your event tables, run the partitioning.pl script.

Check the log and pinlog files in the directory specified by the UPGRADE_LOG_DIR parameter in your partition.cfg file (by default, BRM_Home/upgrade/partition/sqllog). These log files show how long each script took to execute and list any errors that occurred.

---

Important: If any errors are reported, fix them, and run the script again.

---

Adding Purgeable Partitions to Event Tables

Before your BRM system resumes generating event data, use the partition_utils script to add purgeable partitions to your event tables. For more information, see "Adding Partitions".

Restarting BRM

Start all BRM processes. For more information, see "Starting and Stopping the BRM System".
About the Upgrade Scripts and Files

Table 24–1 lists the scripts and files used to upgrade your nonpartitioned BRM database to a partitioned BRM database. These scripts and files are in the BRM_HOME/upgrade/partition folder.

**Important:** To upgrade custom database objects, you might need to create additional SQL scripts. To plug custom scripts into the upgrade, add appropriate SQL file entries to the nonpartitioning-to-partitioning upgrade configuration file, partition.cfg. For more information, see "(Optional) Reconfiguring the Parameters in partition.cfg".

<table>
<thead>
<tr>
<th>Script or file</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crt_pinlog.sql</td>
<td>SQL script that creates the pinlog files.</td>
</tr>
<tr>
<td>partition.cfg</td>
<td>Configuration file in which you must enter details about the Oracle database configuration before you run the upgrade scripts. All the upgrade Perl scripts parse this file to get the database connection parameters.</td>
</tr>
<tr>
<td>partitioning.pl</td>
<td>Master Perl script for the upgrade process. This script calls other SQL scripts to perform the upgrade.</td>
</tr>
<tr>
<td>pin_upg_common.sql</td>
<td>SQL script that creates the common routines needed for the upgrade.</td>
</tr>
<tr>
<td>tables_tobe_partitioned.sql</td>
<td>SQL scripts that implement event table partitioning.</td>
</tr>
<tr>
<td>make_indexes_partition_ready.sql</td>
<td></td>
</tr>
<tr>
<td>event_partitioning.sql</td>
<td></td>
</tr>
<tr>
<td>upg_oracle_functions.pl</td>
<td>Perl script that performs many miscellaneous upgrade tasks related to the Oracle database.</td>
</tr>
</tbody>
</table>
This document provides information about purging obsolete event objects and expired account sub-balances that consume large amounts of space in your Oracle Communications Billing and Revenue Management (BRM) database. It discusses the impact of purging major event objects and how this affects BRM applications.

Before purging data, you should be familiar with the following:

- Basic BRM concepts. See "Introducing BRM" in BRM Concepts.
- BRM system architecture. See "BRM System Architecture" in BRM Concepts.
- Database administration.

### About Purging Event Objects

You can purge event objects that are no longer required for daily business operations from your BRM database as follows:

- If your event tables are partitioned, you can use the `partition_utils` utility to purge event objects. See "Partitioning Database Tables".
- To purge event objects from any type of database, you can use custom purging scripts. For more information, contact Oracle.

Purging event objects enables you to do the following:

- Delete obsolete data from your database.
- Reduce storage space in your database.
- Reduce the number of event objects in your database, making it easier to upgrade to later releases of BRM.

### Categories of BRM Event Objects

Event objects are grouped into the following categories:

- Event Objects That Have a Balance Impact
- Event Objects That Do Not Have a Balance Impact
- Event Objects Stored in `partition_historic`

### Event Objects That Have a Balance Impact

Event objects that have a balance impact are required for the following functions:

- Billing
■ Accounts receivable (A/R) operations
■ Tracking session charges

The balance impacts in these event objects are stored in the `EVENTS_BAL_IMPACTS_T` table. These objects are typically created by usage, cycle, and purchase events.

---

**Caution:** Purge these event objects only after you finish all billing, A/R, and session event processing for the current billing cycle.

---

**Event Objects That Do Not Have a Balance Impact**

Event objects that do not have a balance impact are not needed by BRM for rating or billing. Your business, however, might need them for auditing. Auditing objects and unused event objects are in this category.

If you purge event objects that do not generate a balance impact, you cannot view auditing information for these event objects. For example, if you purge `/event/customer/login` objects and then want to check when a user logged in, you will not be able to find the user’s login event with Event Browser.

---

**Important:** Before purging event objects that do not have a balance impact, verify that the objects aren’t required by any custom code.

---

**Event Objects Stored in `partition_historic`**

If your event tables are partitioned, event objects that are required for the product to behave correctly or for auditing purposes are stored in the `partition_historic` table.

---

**Important:** If you must purge event objects in your implementation, before your purge them make sure you understand the impact of purging those objects. See "Impact of Purging Event Objects”.

---

By default, the following types of event objects are stored in `partition_historic` when you install BRM:

■ `/event/billing/cdc_update`
■ `/event/billing/cdc_update`
■ `/event/billing/cycle/discount`
■ `/event/billing/cycle/discount/mostcalled`
■ `/event/billing/cycle/rollover`
■ `/event/billing/cycle/rollover/monthly`
■ `/event/billing/cycle/rollover_transfer`
■ `/event/billing/deal`
■ `/event/billing/deal/purchase`
■ `/event/billing/deal/cancel`
■ `/event/billing/discount/action`
■ `/event/billing/discount/action/cancel`
About Purging Event Objects

You can customize this list to meet your business requirements. For more information on partition_historic, see "About Partitioning".

Impact of Purging Event Objects

Before purging event objects, you must know this information:

- Impact of purging the event objects
- How long to keep the event objects

The following tables describe the impact of purging event objects that contain data for billing, A/R, opening and closing sessions, and auditing.

---

**Note:** Only major classes of events and events stored by default in the partition_historic tables are listed in this document. For a complete list of event classes, including information on when they are created, see "Storable Class Definitions" in BRM Developer’s Reference.

---

Billing Event Objects

Table 25–1 lists the impacted Billing Event Objects.
### Table 25–1  Billing Event Objects

<table>
<thead>
<tr>
<th>Event object</th>
<th>Impact of purging/when to purge</th>
</tr>
</thead>
<tbody>
<tr>
<td>/event/billing/cdc_update</td>
<td>If you purge these events, the number of contract days for resources will be lost. You cannot apply discounts based on the cumulative number of contract days for resources.</td>
</tr>
<tr>
<td>/event/billing/cdcd_update</td>
<td>If you purge these events, the number of contract days for discount resources will be lost. You cannot apply discounts based on the cumulative number of contract days for discount resources.</td>
</tr>
</tbody>
</table>
| /event/billing/charge and all of its subclasses | If you purge these event objects:  
- You cannot use the `resubmit` parameter with the `pin_recover` utility to resubmit failed credit card transactions.  
- The `pin_clean` utility cannot report VERIFY checkpoints.  
If you have run the `pin_recover` and `pin_clean` utilities and successfully recovered all failed credit card transactions, you can purge these event objects. |
| /event/billing/cycle and all of its subclasses | If you purge these event objects:  
- Invoices cannot display details about subscription charges applied to accounts.  
- Data in default reports generated by the `pin_ledger_report` utility will not tally correctly because the reports display charges accrued by these objects. For information about the utility, see BRM Collecting General Ledger Data.  
- You cannot perform A/R activities such as adjustments, write-offs, disputes, and settlements for subscription charges logged by these events.  
If you generate invoices and G/L reports, keep these event objects through the current billing cycle. |
| /event/billing/cycle/discount/mostcalled | If you purge these events, all the information about the total charges, duration, and the number of calls for the most called numbers will be lost. You cannot apply the most-called number discounts.  
See also the impact of purging /event/billing/cycle and all of its subclasses. |
| /event/billing/cycle/rollover/monthly | Purging these objects will have the following impact:  
- Rollover correction during final billing will not work if the rollover events for the current cycle have been purged.  
- Rerating will not work correctly if the rollover events in the rerating period have been purged.  
- The controlled rollover of free resources during plan transition will not work if the rollover events for the current cycle have been purged.  
See also the impact of purging /event/billing/cycle and all of its subclasses. |
| /event/billing/cycle/rollover_transfer | If you purge these events, information about the balance impacts of the original rollover event will be lost.  
See also the impact of purging /event/billing/cycle and all of its subclasses.  
You may want to keep these objects for auditing purposes. |
| /event/billing/deal | Abstract class, not persisted in the database. |
### About Purging Event Objects

#### Table 25–1 (Cont.) Billing Event Objects

<table>
<thead>
<tr>
<th>Event object</th>
<th>Impact of purging/when to purge</th>
</tr>
</thead>
</table>
| /event/billing/deal/cancel | If you purge these events:  
  - Rerating, which needs to replay these events, will not work correctly. However, you can purge old events that occurred before your rerating time frame.  
  - Best pricing will not work correctly because rerating is called during best pricing.  
  - You cannot see the corresponding event history in Customer Center. You can, however, see event history contained in other types of deal event objects that were not purged. Keep these event objects as long as you want to display deal cancellation history. |
| /event/billing/deal/purchase | If you purge these events:  
  - Rerating will not work correctly. However, you can purge old events that occurred before your rerating time frame.  
  - Best pricing will not work correctly because rerating is called during best pricing.  
  You cannot see deal purchase history. Keep these event objects as long as you want to display the history of deal purchases. |
| /event/billing/debit | If you purge these event objects:  
  - The Bill Details panel in Customer Center cannot display debit event objects associated with a particular item.  
  - Invoices cannot display event details for debit or credit items completed during the current billing cycle.  
  - If you generate any custom reports that use debit events, the reports will be incorrect.  
  - If you configure the `pin_ledger_report` utility to report debits and credits, the reported numbers will be incorrect. (For information about the utility, see BRM Collecting General Ledger Data.)  
  Keep these event objects as long as you want to do the following:  
  - Display debit events.  
  - Display event details in an invoice.  
  - Create custom reports that use debit events. |
| /event/billing/discount/action/cancel | The PCM_OP_SUBSCRIPTION_READ_ACCT_PRODUCTS and PCM_OP_SUBSCRIPTION_GET_HISTORY opcodes refer to these events to show the history of the discount.  
  If these event objects are purged, you cannot see the corresponding discount history. You can, however, see the discount history contained in other types of event objects that are not purged. If these are the only events for the discount, then no history will be shown. |
| /event/billing/discount/action/modify | The PCM_OP_SUBSCRIPTION_READ_ACCT_PRODUCTS and PCM_OP_SUBSCRIPTION_GET_HISTORY opcodes refer to these events to show the history of the discount.  
  If these event objects are purged, you cannot see the corresponding discount history. You can, however, see discount history contained in other types of event objects that were not purged.  
  Keep these event objects if you need to display the history of discount attribute and status modification. |
About Purging Event Objects

### Table 25–1 (Cont.) Billing Event Objects

<table>
<thead>
<tr>
<th>Event object</th>
<th>Impact of purging/when to purge</th>
</tr>
</thead>
<tbody>
<tr>
<td>/event/billing/discount/action/modify/status</td>
<td>If these event objects are purged, you cannot see the corresponding discount history. You can, however, see discount history contained in other types of event objects that were not purged. Keep these event objects if you need to display the history of discount attribute and status modification.</td>
</tr>
<tr>
<td>/event/billing/discount/action/purchase</td>
<td>If you purge these event objects, you cannot see the corresponding discount history. Keep these event objects as long as you want to display the history of discount purchases.</td>
</tr>
</tbody>
</table>
| /event/billing/discount/action/set_validity | If you purge these events:  
- Rating will not consider the validity period setting because the discount validity period settings prior to first usage and the new calculated validity period after first usage will be lost.  
- Rerating will not work correctly. However, you can purge old events that occurred before your rerating time frame. |
| /event/billing/lcupdate | If this event is purged, discount amount based on the number of active subscriptions will not be applied. |
| /event/billing/limit | Do not purge these event objects if you have any custom applications that use these objects. |
| /event/billing/mfuc_update | If this event is purged, monthly fee and usage discount will not be applied at billing time. You may also want to keep these objects for auditing purposes. |
| /event/billing/ordered_balgrp | Abstract class, not persisted in the database. |
| /event/billing/ordered_balgrp/create | If you purge these events, the history of the creation of the object is lost. You may also want to keep these objects for auditing purposes. |
| /event/billing/ordered_balgrp/delete | If you purge these events, the history of the deletion of the object is lost. You may also want to keep these objects for auditing purposes. |
| /event/billing/ordered_balgrp/modify | If you purge these events, the history of the modifications of the object is lost. You may also want to keep these objects for auditing purposes. |
| /event/billing/product | If you purge these event objects:  
- Reports generated by the `pin_ledger_report` utility cannot display purchase and cancellation fees, and data shown in these reports will be incorrect.  
- Invoices cannot display purchase or cancellation fees.  
Keep these event objects if either of the following is true:  
- You use Customer Center to view current or historical information for canceled and item products.  
- Your custom applications use these event objects. |
Table 25–1 (Cont.) Billing Event Objects

<table>
<thead>
<tr>
<th>Event object</th>
<th>Impact of purging/when to purge</th>
</tr>
</thead>
</table>
| /event/billing/product/action/cancel | If you purge these events:  
- Rerating, which needs to replay these events, will not work correctly. However, you can purge old events that occurred before your rerating time frame.  
- Best pricing will not work correctly because rerating is called during best pricing,  
- You cannot see the corresponding event history in Customer Center. You can, however, see event history contained in other types of deal event objects that were not purged. Keep these event objects as long as you want to display deal cancellation history. |
| /event/billing/product/action/modify | If you purge these events:  
- Rerating will not work correctly. However, you can purge old events that occurred before your rerating time frame.  
- Best pricing will not work correctly because rerating is called during best pricing,  
- You cannot see the corresponding product history. You can, however, see product history contained in other types of product event objects that were not purged. Keep these event objects if you need to display the history of product attribute and status modification. |
| /event/billing/product/action/modify/status | If you purge these events:  
- Rerating will not work correctly. However, you can purge old events that occurred before your rerating time frame.  
- You cannot see the corresponding product history. Keep these event objects as long as you want to display the history of product purchases. |
| /event/billing/product/action/purchase | If you purge these events:  
- Rerating will not work correctly. However, you can purge old events that occurred before your rerating time frame.  
- You cannot see the corresponding product history. |
| /event/billing/product/action/set_validity | If you purge these events:  
- Rating will not consider the validity period setting because the product’s validity period settings prior to first usage and the new calculated validity period after first usage will be lost.  
Also during the back-out process, rerating uses these events to remove the validity setting.  
**Note:** You can purge old events that occurred before your rerating time frame. |
| /event/billing/product/fee/cycle/cycle_forward_annual  /event/billing/product/fee/cycle/cycle_forward_bimonthly  /event/billing/product/fee/cycle/cycle_forward_monthly | If you purge the cycle_forward_* events for a particular period, features or operations that involve cancellation will not work correctly. For example, if all the cycle_forward_* events from 01/01/2008 to 06/01/2008 are purged, then operations, such as product or discount cancellations, service inactivations, plan or deal transitions, and change of options, which internally call the cancellations of the product or discount during that time period will not give correct results.  
During product and discount cancellations, BRM searches for all the original charged events and refunds the charges for each event individually. If these events are purged, then charges for these events will not be refunded. |
| /event/billing/validate/cc | You may want to keep these objects for auditing purposes, for example, to keep track of credit card validations. |
### Accounts Receivable Event Objects

Table 25–2 lists the Accounts Receivable Event Objects

<table>
<thead>
<tr>
<th>Event object</th>
<th>Impact of purging/when to purge</th>
</tr>
</thead>
<tbody>
<tr>
<td>/event/billing/adjustment/account</td>
<td>If you purge these event objects, you lose some G/L information.</td>
</tr>
<tr>
<td>/event/billing/adjustment/item</td>
<td>The pin_ledger_report utility uses these event objects to generate the G/L report for billed_earned and billed revenue types. (For information about the utility, see BRM Collecting General Ledger Data.) Keep these event objects for the specified G/L posting period.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>/event/billing/adjustment/event</td>
<td>If you purge these event objects, you lose some G/L information.</td>
</tr>
<tr>
<td></td>
<td>The pin_ledger_report utility uses these event objects to generate the G/L report for billed_earned and billed revenue types.</td>
</tr>
<tr>
<td></td>
<td>Keep these event objects for the specified G/L posting period.</td>
</tr>
<tr>
<td>/event/billing/adjustment/tax_event</td>
<td>If you purge these event objects, you lose some G/L information.</td>
</tr>
<tr>
<td></td>
<td>The pin_ledger_report utility uses these event objects to generate the G/L report for billed_earned and billed revenue types.</td>
</tr>
<tr>
<td></td>
<td>Keep these event objects for the specified G/L posting period.</td>
</tr>
<tr>
<td>/event/billing/batch/payment</td>
<td>If you purge these event objects:</td>
</tr>
<tr>
<td></td>
<td>■ You cannot use the pin_recover utility with the resubmit parameter to resend failed transactions.</td>
</tr>
<tr>
<td></td>
<td>■ You cannot use the pin_clean utility.</td>
</tr>
<tr>
<td></td>
<td>You can purge the original batch payment event objects after you successfully resubmit any failed batch transactions.</td>
</tr>
<tr>
<td>/event/billing/batch/reversal</td>
<td>You can purge these event objects if you do not have any custom applications using them.</td>
</tr>
<tr>
<td>/event/billing/dispute/item</td>
<td>If you purge these event objects, you lose some past G/L, dispute, and settlement information.</td>
</tr>
<tr>
<td>/event/billing/settlement/item</td>
<td>The pin_ledger_report uses these event objects to generate the G/L report for billed_earned and billed revenue types. For information about the utility, see BRM Collecting General Ledger Data.</td>
</tr>
<tr>
<td></td>
<td>Keep these event objects for the specified G/L posting period.</td>
</tr>
<tr>
<td>/event/billing/item/transfer</td>
<td>If you purge these event objects:</td>
</tr>
<tr>
<td></td>
<td>■ You cannot audit the related bill items.</td>
</tr>
<tr>
<td></td>
<td>■ A/R Tool might not show the actions performed on the related bill items.</td>
</tr>
<tr>
<td></td>
<td>■ Payments cannot be reversed. You must preserve these event objects if you intend to reverse payments at a future date. To find the items a payment was applied to, the PCM_OP_BILL_REVERSE_PAYMENT opcode searches for all the transfer events in which PIN_FLD_ITEM_OBJECT is equal to the payment item’s POID.</td>
</tr>
<tr>
<td></td>
<td>Keep transfer event objects as long as you keep their related billable items.</td>
</tr>
</tbody>
</table>
### Delayed Event Objects

Table 25–3 lists the Delayed Event Objects.

<table>
<thead>
<tr>
<th>Event object</th>
<th>Impact of purging/when to purge</th>
</tr>
</thead>
<tbody>
<tr>
<td>/event/billing/payment and all of its subclasses</td>
<td>If you purge these event objects:</td>
</tr>
<tr>
<td></td>
<td>- You cannot perform payment reversals.</td>
</tr>
<tr>
<td></td>
<td>- If you generate G/L reports with the pin_ledger_report utility, the reports do not show that</td>
</tr>
<tr>
<td></td>
<td>payments recorded in the purged events have been made. By default, G/L reports show</td>
</tr>
<tr>
<td></td>
<td>payment data.</td>
</tr>
<tr>
<td></td>
<td>- Invoices for the current billing cycle cannot show event details for payments.</td>
</tr>
<tr>
<td></td>
<td>- Payment Tool cannot display payment information.</td>
</tr>
<tr>
<td></td>
<td>- Custom reports that use payment objects will be incorrect.</td>
</tr>
<tr>
<td></td>
<td>If you have already reversed payments or your company is bound by corporate policies prohibiting</td>
</tr>
<tr>
<td></td>
<td>payment reversal after a specified period of time, you can purge these events.</td>
</tr>
<tr>
<td>/event/billing/reversal and all of its subclasses</td>
<td>If you purge these event objects:</td>
</tr>
<tr>
<td></td>
<td>- If the pin_ledger_report utility is configured to report reversals, G/L reports will be</td>
</tr>
<tr>
<td></td>
<td>incorrect for periods whose events were purged. For information about the utility, see</td>
</tr>
<tr>
<td></td>
<td>BRM Collecting General Ledger Data.</td>
</tr>
<tr>
<td></td>
<td>- Invoices for the current billing cycle cannot display event details for the purged payment</td>
</tr>
<tr>
<td></td>
<td>reversals.</td>
</tr>
<tr>
<td></td>
<td>- Any custom reports that use the purged reversal objects will be incorrect. They cannot</td>
</tr>
<tr>
<td></td>
<td>display details of event reversals.</td>
</tr>
<tr>
<td></td>
<td>Your business requirements determine how long you keep these event objects.</td>
</tr>
<tr>
<td>/event/billing/writeoff/account</td>
<td>If you purge these event objects, you lose some G/L and tax information.</td>
</tr>
<tr>
<td>/event/billing/writeoff/bill</td>
<td>The pin_ledger_report utility uses these event objects to generate the G/L report for billed</td>
</tr>
<tr>
<td>/event/billing/writeoff/item</td>
<td>earned and billed revenue types.</td>
</tr>
<tr>
<td>/event/billing/writeoff/tax_account</td>
<td>Keep these event objects for the specified G/L posting period.</td>
</tr>
<tr>
<td>/event/billing/writeoff/tax_bill</td>
<td></td>
</tr>
<tr>
<td>/event/billing/writeoff/tax_item</td>
<td></td>
</tr>
</tbody>
</table>

### Group Event Objects

Table 25–4 lists the Group Event Objects.
### Table 25–4  Group Event Objects

<table>
<thead>
<tr>
<th>Event object</th>
<th>Impact of purging/when to purge</th>
</tr>
</thead>
<tbody>
<tr>
<td>/event/group/member</td>
<td>Purge these event objects if you do not need to track the addition or deletion of member accounts.</td>
</tr>
<tr>
<td>/event/group/parent</td>
<td>Purge these event objects if you do not need to track the creation of parent accounts.</td>
</tr>
</tbody>
</table>

### Sharing Group Event Objects

Table 25–5 lists the Sharing Group Event Objects.

### Table 25–5  Sharing Group Event Objects

<table>
<thead>
<tr>
<th>Event object</th>
<th>Impact of purging/when to purge</th>
</tr>
</thead>
<tbody>
<tr>
<td>/event/group/sharing/charges/create</td>
<td>Purging these objects does not affect the behavior of the product. You may want to keep these objects for auditing purposes, because if you purge these events, the history of creation, deletion, and modification of the /group/sharing/charges object is lost.</td>
</tr>
<tr>
<td>/event/group/sharing/charges/delete</td>
<td>Purging these objects does not affect the behavior of the product. You may want to keep these objects for auditing purposes, because if you purge these events, the history of creation, deletion, and modification of the /group/sharing/charges object is lost.</td>
</tr>
<tr>
<td>/event/group/sharing/charges/modify</td>
<td>Purging these objects does not affect the behavior of the product. You may want to keep these objects for auditing purposes, because if you purge these events, the history of creation, deletion, and modification of the /group/sharing/charges object is lost.</td>
</tr>
<tr>
<td>/event/group/sharing/discounts/create</td>
<td>Purging these objects does not affect the behavior of the product. You may want to keep these objects for auditing purposes, because if you purge these events, the history of creation, deletion, and modification of the /group/sharing/charges object is lost.</td>
</tr>
<tr>
<td>/event/group/sharing/discounts/delete</td>
<td>Purging these objects does not affect the behavior of the product. You may want to keep these objects for auditing purposes, because if you purge these events, the history of creation, deletion, and modification of the /group/sharing/charges object is lost.</td>
</tr>
<tr>
<td>/event/group/sharing/discounts/modify</td>
<td>Purging these objects does not affect the behavior of the product. You may want to keep these objects for auditing purposes, because if you purge these events, the history of creation, deletion, and modification of the /group/sharing/charges object is lost.</td>
</tr>
<tr>
<td>/event/group/sharing/profiles/create</td>
<td>Purging these objects does not affect the behavior of the product. You may want to keep these objects for auditing purposes, because if you purge these events, the history of creation, deletion, and modification of the /group/sharing/charges object is lost.</td>
</tr>
<tr>
<td>/event/group/sharing/profiles/delete</td>
<td>Purging these objects does not affect the behavior of the product. You may want to keep these objects for auditing purposes, because if you purge these events, the history of creation, deletion, and modification of the /group/sharing/charges object is lost.</td>
</tr>
<tr>
<td>/event/group/sharing/profiles/modify</td>
<td>Purging these objects does not affect the behavior of the product. You may want to keep these objects for auditing purposes, because if you purge these events, the history of creation, deletion, and modification of the /group/sharing/charges object is lost.</td>
</tr>
</tbody>
</table>

### Session Event Objects

Table 25–6 lists the Session Event Objects.
Table 25–6  Session Event Objects

<table>
<thead>
<tr>
<th>Event object</th>
<th>Impact of purging/when to purge</th>
</tr>
</thead>
<tbody>
<tr>
<td>/event/session and all of its subclasses</td>
<td>If you purge these event objects:</td>
</tr>
<tr>
<td></td>
<td>■ And if you do not purge the corresponding item objects, the item objects will be inaccurate.</td>
</tr>
<tr>
<td></td>
<td>■ Different event objects point to session event objects (session_obj_* in EVENT_T) for audit purposes. You lose auditing information if you purge session event objects.</td>
</tr>
<tr>
<td></td>
<td>■ You cannot make adjustments or settle disputes on bills containing the corresponding items.</td>
</tr>
<tr>
<td></td>
<td>■ You cannot use invoicing, taxation, bill adjustments, or G/L functionality.</td>
</tr>
<tr>
<td></td>
<td>■ If you purge /event/session/dialup objects, batch accounting operations might be performed more than once, which could result in customers being overcharged.</td>
</tr>
</tbody>
</table>

Note: For dialup events, the CM pin.conf file contains an entry called trans_id_window. Purging event objects older than now - trans_id_window does not affect whether batch accounting occurs more than once.

Most session event objects have balance impacts and are required for billing, A/R, and opening and closing session activity. Therefore, these objects should be kept for at least one accounting cycle.

Table 25–7  Auditing Event Objects

<table>
<thead>
<tr>
<th>Event object</th>
<th>Impact of purging/when to purge</th>
</tr>
</thead>
<tbody>
<tr>
<td>/event/audit/price</td>
<td>Purging these objects does not affect the behavior of the product.</td>
</tr>
<tr>
<td>/event/audit/price/deal</td>
<td>These event objects are generated by BRM auditing features. Keep these event objects as long as you need the auditing information they contain.</td>
</tr>
<tr>
<td>/event/audit/price/discount</td>
<td></td>
</tr>
<tr>
<td>/event/audit/price/plan</td>
<td></td>
</tr>
<tr>
<td>/event/audit/price/product</td>
<td></td>
</tr>
<tr>
<td>/event/audit/price/rate</td>
<td></td>
</tr>
<tr>
<td>/event/audit/price/rate_plan</td>
<td></td>
</tr>
<tr>
<td>/event/audit/price/rate_plan_selector</td>
<td></td>
</tr>
</tbody>
</table>

Auditing Event Objects

Table 25–7 lists the Auditing Event Objects.

About Purging Account Sub-Balances

You can purge expired account sub-balances that are no longer required for daily business operations from your BRM database. Expired sub-balances are the validity-based balances whose valid end dates are in the past.

Caution: When you delete sub-balances from the database, events that impacted those sub-balances cannot be rerated. Make sure you no longer need the expired sub-balances before deleting them.

You delete expired sub-balances by running the pin_sub_balance_cleanup utility. Run this utility from the BRM_Home/apps/pin_subscription directory, which contains the
appropriate configuration file. Use the following command to run the `pin_sub_balance_cleanup` utility:

```
pin_sub_balance_cleanup -n | -d
```

For more information, see "pin_sub_balance_cleanup".
This chapter describes how to generate virtual columns in the Oracle Communications Billing and Revenue Management (BRM) database. In the current release, virtual columns are generated only for event tables (for the /event class and subclasses).

About Generating Virtual Columns on Event Tables

Oracle Database 11g allows you to create virtual columns on tables. A virtual column is similar to a normal table column but it is defined by an expression. The result of evaluation of this expression becomes the value of the column. A virtual column contains a function upon other table columns. Virtual columns are not physically stored in the table (they are derived from data in the other columns of the table) and their values are computed at run time when you query the data. Being able to create virtual columns is enabled by default in Oracle Database 11g. See the Oracle Database documentation for detailed information about virtual columns.

Implementations of BRM have shown that a high percentage of the BRM database storage space can be used by the event tables. BRM can use virtual columns in a way that results in space savings for event records. To use virtual columns in the BRM database, you convert event classes (/event and its subclasses) in the BRM schema to use virtual columns. You convert event classes to use virtual columns by running the pin_virtual_gen utility (see “Generating Virtual Columns on Event Tables” for instructions). The savings in database storage applies to event data that the system creates after the virtual columns are generated (not to existing event data). Virtual column functionality is transparent to the BRM application.

BRM creates virtual columns on the POID field_name_type columns of event tables in the BRM database. The POID field_name_type columns are the columns that are SQL mappings for the PIN_FLDT_POID type (the POID data type). After the virtual columns are enabled, a new column is added named field_name_type_id that stores the ID mapping for the value in the POID field_name_type column.

For example, these fields within the event classes are candidates for conversion to virtual columns:

- **PIN_FLD_POID**
  
  The poid_type column is converted to a virtual column.

- **PIN_FLD_ACCOUNT_OBJ**
  
  The account_obj_type column is converted to a virtual column.

- **PIN_FLD_BRAND_OBJ**
  
  The brand_obj_type column is converted to a virtual column.
After the event classes have been enabled to have virtual columns, any new event subclass (for example, /event/billing/discount/new) will be virtual-column enabled. Specifically, all PIN_FLDT_POID field_name_type columns within the new event subclass will use virtual columns. Virtual columns cannot be enabled only for a subclass (the base class must be virtual-column enabled).

**Note:** If you have a non-partitioned schema, and you want to enable partitions, you must enable the partitions before enabling virtual columns. After you enable virtual columns on a non-partitioned schema, you cannot enable partitioning.

See "Generating Virtual Columns on Event Tables" for instructions on converting event classes in the BRM schema to use virtual columns.

For additional information on how using virtual columns in your BRM database may impact your system, see the following documentation:

- See the discussion on creating custom applications in BRM Developer’s Guide for information on how to make your custom applications support virtual columns.
- See the discussion on Rated Event (RE) Loader in BRM System Administrator’s Guide for information on how to set up RE Loader when using a virtual column-enabled system.
- See the discussion on the Conversion Manager pin_cmt utility in BRM System Administrator’s Guide for information on moving legacy data when using a virtual column-enabled system.

### Generating Virtual Columns on Event Tables

This section describes how to generate virtual columns in the BRM database for event tables (for the /event class and subclasses).

You use the pin_virtual_gen utility to convert a standard BRM database into one with virtual columns. The utility generates virtual columns on event tables for all event classes.

For information about using virtual columns in BRM, see "About Generating Virtual Columns on Event Tables".

For information about pin_virtual_gen syntax, see "pin_virtual_gen".

Before you can generate virtual columns, do the following:

- Install BRM 7.4 Patch Set 16.
  
  See BRM Patch Set 16 Installation Guide.
- Plan for downtime of your BRM system.
  
  The duration of time for running the utility that generates virtual columns is a downtime for the BRM system.
- (Optional) If you have a non-partitioned event table and you want to enable partitioning, install the 7.4_PartitionUpg_platform_32_opt.bin package and enable partitioning for the event tables.
Important: If you have a non-partitioned schema, and you want to enable partitions, you must enable the partitions before enabling virtual columns. After you enable virtual columns on a non-partitioned schema, you cannot enable partitioning.

(Optional) The pin_virtual_gen utility uses an Infranet.properties file which is preconfigured with the required values to run it. For information on setting the log level and number of threads for the utility, see 'pin_virtual_gen'.

To generate virtual columns on event tables:

1. Go to BRM_Home/apps/pin_virtual_columns.
2. While your BRM system is processing, run the following command.
   
   pin_virtual_gen -gentasks verify_types -execute

   The utility checks if you have POID custom type names for your custom storable classes.

   This can be a long-running process.

3. Do one of the following:
   - If the utility returns a message that it found no invalid object names, stop the BRM system.
   - If the utility returns a message that it found object names that are missing from the data dictionary, do the following:
     a. Stop the BRM system.
     b. Run the following command:

        pin_virtual_gen -gentasks create_types -execute

        This command stores the POID custom type names of custom storable class types in the data dictionary of the BRM schema (required for a virtual column-enabled system).

4. Run the following command:

   pin_virtual_gen -gentasks create -execute

   This command converts the tables within the /event storable class and its subclasses to use virtual columns.

   If the pin_virtual_gen utility is interrupted while running this command, run the following command.

   pin_virtual_gen -readtasks create -execute

5. Start the BRM system.

For more information about using the pin_virtual_gen utility, see "pin_virtual_gen" and "Viewing Tasks for Generating Virtual Columns".

Viewing Tasks for Generating Virtual Columns

When you run the pin_virtual_gen utility, various tasks (jobs) are run to generate the virtual columns. The statuses of the tasks are maintained in the database so if the utility is interrupted, you can restart it without any issue.
Each task has a task ID. You can view the tasks before or after they are run (before or after they have generated virtual columns). You can view task details of all tasks or only tasks within a task ID range.

You may want to view tasks at the following times:

- **Before you generate virtual columns**, when you want to see what is going to happen to your database. By viewing the SQL statements of the tasks, you can see which tables will have virtual columns added to them and which tables will be renamed.

  See “Viewing and then Running Virtual-Column Tasks”.

- **After you generate virtual columns**, when you want to see the tasks that completed. For example, if the `pin_virtual_gen` utility is interrupted while running, you might want to view the tasks that ran before the interruption.

  See “Viewing Virtual-Column Task Details”.

For more information, see “pin_virtual_gen”.

### Viewing and then Running Virtual-Column Tasks

To view virtual-column tasks and then run them:

1. Go to `BRM_Home/apps/pin_virtual_columns`.
2. Run the following command:
   
   ```
   pin_virtual_gen -gentasks create
   ```

   This command generates the tasks and stores them in the database without executing them.

3. Do one of the following:

   - To view task details of all tasks, run the following command:
     ```
     pin_virtual_gen -showtasks
     ```

   - To view task details for tasks within a task ID range, run the following command:
     ```
     pin_virtual_gen -showtasks [minID maxID]
     ```

     where you want to see tasks that have an ID greater than `minID` and less than `maxID`.

4. After viewing the tasks, run them by entering the following command:
   ```
   pin_virtual_gen -readtasks create -execute
   ```

   This command reads the tasks from the database and runs them.

---

**Note:** If there is an error before the job completes after running either the `gentasks` or `readtasks` commands (for example, if there is a power outage), run the following command:

```
pin_virtual_gen -readtasks create -execute
```

The `readtasks` command reads the statuses of the various tasks recorded in the database and runs the appropriate tasks.
Viewing Virtual-Column Task Details
To view the details of all virtual-column tasks, run the following command:

```
pin_virtual_gen -showtasks 0 -
```

To view the details of virtual-column tasks by task ID, run the following command:

```
pin_virtual_gen -showtasks [minID maxID]
```

where you want to see tasks that have an ID greater than \( minID \) and less than \( maxID \).
If you omit \( minID \) and \( maxID \), the details of all tasks are displayed.

Exporting a BRM Schema with Virtual Columns

After you generate virtual columns, if you need to export your BRM schema, you must remove the virtual columns from the schema before the export. In addition, if you need to restore the exported schema, you need to add the virtual columns back after the import.

---

**Note:** Ensure that you use the Oracle Database export and import utilities that support virtual columns. Refer to the Oracle Database 11g documentation for information.

---

To export a BRM schema with virtual columns:

1. Stop the BRM system.
2. Go to `BRM_Home/apps/pin_virtual_columns` and run the following command:
   ```
   pin_virtual_gen -gentasks pre_export -execute
   ```
   This command removes the virtual columns.
3. Export the schema using the Oracle Database export utility (for example, run the `expdp` command).
   The BRM schema is now exported into a dump file and has no virtual columns.
4. From `BRM_Home/apps/pin_virtual_columns`, run the following command:
   ```
   pin_virtual_gen -gentasks post_export -execute
   ```
   This command restores the virtual columns (the BRM schema virtual columns are again enabled).
5. Start the BRM system.
6. If you need to revert back to your previous database (restore the database by importing the schema from the dump file back to disk), do the following:
   a. Stop the BRM system.
   b. Import the schema using the Oracle Database import utility (for example, run the `impdp` command).
      The imported schema does not have the virtual columns because you removed them when you exported the schema to the dump file.
   c. From `BRM_Home/apps/pin_virtual_columns`, run the following command:
      ```
      pin_virtual_gen -gentasks post_export -execute
      ```
This command restores the virtual columns in your restored database (the BRM schema virtual columns are again enabled).

d. Start the BRM system.
Part VII describes how to manage an Oracle Communications Billing and Revenue Management (BRM) multidatabase system. It contains the following chapters:

- Managing a Multidatabase or Multischema System
- Multidatabase Utilities
This document provides guidelines to help you set up and manage a Oracle Communications Billing and Revenue Management (BRM) multidatabase or multischema system, as well as information about how to set up the database, create custom tables for replication, synchronize data, and change process time intervals.

Before you read this document, you should be familiar with how BRM works. See "BRM System Architecture" in BRM Concepts.

For more information about monitoring and maintaining a BRM system, see "Monitoring and Maintaining Your BRM System".

**Important:** Multidatabase functionality is an optional component, not part of base BRM. For information on installing a multidatabase system, see: "Installing a Multidatabase System" in BRM Installation Guide.

### About Multidatabase Systems

MultiDB Manager is an optional feature that enables multiple complete databases in a single BRM installation. A multidatabase system lets you distribute your customer accounts among several databases, allowing increased storage capacity. For an overview, see "A BRM Multidatabase Production System" in BRM Installation Guide.

You can run billing and invoicing utilities against each BRM database. For more information, see "Setting Up Billing to Run in a Multidatabase Environment" in BRM Configuring and Running Billing and "Configuring pin.conf for Multiple Databases" in BRM Designing and Generating Invoices.

The main administrative tasks for a multidatabase system are:

- Converting a Single-Database System to a Multidatabase System
- Preparing to Manage a Multidatabase System
- Adding Databases to a Multidatabase System
- Setting Database Status
- Setting Database Priorities
- Changing Refresh Intervals for BRM Groups
- Forcing a Database Refresh
- Creating Custom Tables for Replication
Converting a Single-Database System to a Multidatabase System

To convert a single-database system to a multidatabase system, see "Installing a Multidatabase System" in BRM Installation Guide.

Preparing to Manage a Multidatabase System

Before working with your multidatabase system, verify that the PERL5LIB environment variable is a system variable and contains $BRM_HOME/perl/lib/5.8.0.

Adding Databases to a Multidatabase System

To add one or more databases to a multidatabase system:

1. On the new secondary database machine, install Oracle and then create your BRM database. For more information, see "Configuring Oracle for BRM" in BRM Installation Guide.

Using Pipeline Manager with Multiple Databases

When you use MultiDB Manager, you run an instance of Pipeline Manager for each BRM database. Two types of data need to be managed:

- **Account data**: When account data changes in a database, the Account Synchronization Data Manager (DM) sends the data to Pipeline Manager. The DAT_Listener module map file specifies which Pipeline Manager instance the data is sent to.

- **CDR data**: Before a call details record (CDR) file is processed, the Pipeline Manager FCT_AccountRouter module separates CDRs by BRM databases and sends the CDRs to the appropriate Pipeline Manager instance.

Figure 27-1 shows how FCT_AccountRouter manages incoming CDRs:

---

**Figure 27-1** FCT_AccountRouter CDR Management

Account router retrieves uniqueness data from BRM
2. Verify that all primary and secondary databases are interconnected. For more information, see "Verifying That the Installation Machines Connect to Both Databases" in BRM Installation Guide.


5. Verify that your secondary installation machine can connect to all databases.

6. Configure the pin_multidb.conf file on the primary installation machine.
   See "Configuring the pin_multidb.conf File on the Primary Installation Machine".

7. Run the pin_multidb.pl script on the primary installation machine. See "Running the pin_multidb.pl Script on the Primary Installation Machine" in BRM Installation Guide.

### Configuring the pin_multidb.conf File on the Primary Installation Machine

Enter your custom configuration parameters for your new secondary database in the pin_multidb.conf file.

To configure your new secondary database, perform the following on the primary installation machine:

1. Open the BRM_Home/setup/scripts/pin_multidb.conf file in a text editor.

2. Modify the following parameters to indicate which database instances you are adding to the system. For example, if your system contains three secondary databases and you are adding two more databases, set $PIN_MD_SECONDARY_START_INST to 3 and $PIN_MD_SECONDARY_STOP_INST to 4. This tells the multidatabase installation script to configure only database instances 3 and 4 and to ignore database instances 0, 1, and 2.

   $PIN_MD_SECONDARY_START_INST = '3';
   $PIN_MD_SECONDARY_END_INST = '4';

3. Modify the following configuration entries for each new database, making sure you:
   - Create a set of $PIN_MD_SECONDARY* entries for each database you are adding to the system.
   - Update the array number to the appropriate value.
   - Update the secondary database number.
   - Update the secondary database alias.
   - Update the host name of the secondary DM machine.

   Important:
   - All databases in a multidatabase system must use the same user name and database character set.
   - The database must also have the same default domain name in the Oracle_Home/network/admin/sqlnet.ora file as the other databases.
Setting Database Status

Database status determines whether a database is available for account creation. Databases can be set to open, closed, or unavailable:

- **Open status**: Open databases are available for account creation.
- **Closed status**: Closed databases are not used for account creation under most circumstances. Accounts are created in a closed database only if an account’s parent, branded, or sponsoring account belongs to that database or if all databases are closed. If all databases are closed, the multidatabase software chooses a closed database at random in which to create accounts and continues to create accounts in that database until a database becomes open. To limit the number of accounts created on a database, you can manually change the database’s status to closed, or you can have the multidatabase software automatically change it to closed when the database reaches a predefined limit.
- **Unavailable status**: Unavailable databases are not used for account creation unless the database contains an account’s parent, sponsoring, or branded account.

To change a database’s status, edit the STATUS entries in the `config_dist.conf` file and then use the `load_config_dist` utility to load the distribution information into the primary database.

**Caution**: The `load_config_dist` utility overwrites existing distributions. If you are updating distributions, you cannot load new distributions only. You must load complete sets of distributions each time you run the `load_config_dist` utility.

To change a database’s status, perform the following on the primary installation machine:

1. Go to the `BRM_Home/apps/multi_db` directory and open the `config_dist.conf` file.
2. Change the values in the STATUS entries:
Setting Database Priorities

Note: If your system contains multiple secondary databases, create a new set of entries for each additional secondary database.

```
DB_NO = "0.0.0.1" ;             # 1st database config. block
PRIORITY = 1 ;
MAX_ACCOUNT_SIZE = 100000 ;
STATUS = "OPEN" ;

DB_NO = "0.0.0.2" ;             # 2nd database config. block
PRIORITY = 3;
MAX_ACCOUNT_SIZE = 50000 ;
STATUS = "OPEN" ;
```

3. Save and close the file.

4. Make sure the pin_config_distribution utility is not running.

5. From the BRM_Home/apps/multi_db directory, run the load_config_dist utility.

6. Stop and restart all Connection Managers. See "Starting and Stopping the BRM System".

Tip: To check how full your databases are, see "Monitoring Database Space" in BRM Installation Guide.

Setting Database Priorities

Database priority determines when customer accounts are created on a particular database, relative to other databases. The multidatabase software assigns accounts to an open database with the highest priority. If all databases have the same priority, the multidatabase software chooses an open database at random in which to create the account. This distributes accounts evenly across all databases.

To customize how databases are selected, see "Creating Accounts in a Multidatabase System" in BRM Managing Customers.

Important: To limit the number of accounts on your primary database, you can set your primary database at a lower priority than the secondary databases. Accounts will be created in the secondary databases when possible.

To change database priorities, edit the PRIORITY entries in the config_dist.conf file and then use the load_config_dist utility to load the distribution information into the primary database.

Caution: The load_config_dist utility overwrites all distributions that are already in the database. If you are updating distributions or adding new ones, beware that you cannot load only the new and changed distributions.

To change database priorities, perform the following on the primary installation machine:
1. Go to the `BRM_Home/apps/multi_db` directory and open the `config_dist.conf` file.

2. Edit the `PRIORITY` entries. In the following example, BRM creates accounts on database 0.0.0.2 because it has the highest priority setting of all open databases.

   Note: If your system contains multiple secondary databases, create a new set of entries for each additional secondary database.

   ```
   DB_NO = "0.0.0.1" ;  # 1st database config. block
   PRIORITY = 1 ;
   MAX_ACCOUNT_SIZE = 100000 ;
   STATUS = "OPEN" ;

   DB_NO = "0.0.0.2" ;  # 2nd database config. block
   PRIORITY = 3 ;
   MAX_ACCOUNT_SIZE = 50000 ;
   STATUS = "OPEN" ;

   DB_NO = "0.0.0.3" ;  # 3rd database config. block
   PRIORITY = 5 ;
   MAX_ACCOUNT_SIZE = 50000 ;
   STATUS = "CLOSED" ;
   ```

3. Save and close the file.

4. Make sure the `pin_config_distribution` utility is not running.

5. From a command prompt, go to the `BRM_Home/apps/multi_db` directory and run the `load_config_dist` utility.

   Note: The `load_config_dist` utility requires a configuration file. See "Creating Configuration Files for BRM Utilities".

6. Stop and restart all Connection Managers. See "Starting and Stopping the BRM System".

Tip: To check how full your databases are, see "Monitoring Database Space" in `BRM Installation Guide`.

Changing Refresh Intervals for BRM Groups

The refresh interval controls how often a configuration object group is replicated from the primary database to the secondary databases. Refresh intervals are originally set during multidatabase installation through the `pin_multidb.conf` file.

Changing the Refresh Interval

To change the refresh interval for a configuration object group, perform the following on the primary installation machine:

1. Go to the `BRM_Home/setup/scripts` directory and open the `pin_multidb.conf` file.

2. Change the values in the Refresh Groups section:
To re-create refresh groups in the secondary database with new frequencies, go to the `BRM_Home/setup/scripts` directory and run `pin_multidb.pl -R`.

To create a specific refresh group, substitute the name of the group. To create all refresh groups, use `all`. For example:

```bash
cd BRM_Home/setup/scripts
perl pin_multidb.pl -R all
```

For information, see "pin_multidb".

4. Stop and restart the primary and secondary DMs and all CMs. See "Starting and Stopping the BRM System".

### Forcing a Database Refresh

You can force the multidatabase software to refresh a particular configuration object group immediately rather than wait for it to refresh according to the interval setting. You might do this, for example, if you made changes to the price object.

To refresh a particular configuration object group immediately, perform the following on the primary installation machine:

1. Go to the `BRM_Home/setup/scripts` directory and run `pin_multidb.pl -r`:

   ```bash
cd BRM_Home/setup/scripts
perl pin_multidb.pl -r group
```

   __Note:__ For information, see "pin_multidb".

2. Stop and restart the primary and secondary DMs and the CM. See "Starting and Stopping the BRM System".

### Creating Custom Tables for Replication

You can create custom tables or modify existing tables after your multidatabase system is installed and configured. To replicate these tables to your secondary databases or make the tables available to all databases in your system, you must perform one of the following:

- Creating Custom Tables You Want Replicated to the Secondary Databases
- Creating Custom Tables That Are Available to All Databases
Creating Custom Tables You Want Replicated to the Secondary Databases

To replicate custom tables, perform the following on the primary installation machine:

1. Create your custom table in the primary database by connecting to the multidatabase CM.
2. Name your table using the following naming convention:

   - CONFIG*
   - PLAN*
   - PRODUCT*
   - RATE*
   - UNIQUEN*
   - STRINGS*
   - CHANNEL*
   - SEARCH*
   - ZONE*_T
   - TOD*
   - FOLD*

3. Add an index on the primary key of each custom table that you want replicated.
4. Perform step 2 of the `pin_multidb.pl -i` script:
   a. Go to the `BRM_Home/setup/scripts` directory and run `pin_multidb.pl -i`:
      ```
      cd BRM_Home/setup/scripts
      perl pin_multidb.pl -i
      ```
   b. At the following prompt, enter `yes` to begin configuration:
      ```
      Do you want to start the configuration now? (y/n): Y
      ```
   c. At the following prompt, enter 2 to initialize the primary database:
      ```
      Please enter the starting step (0-8). If you don’t know, enter 0: 2
      ```
   d. Exit the `pin_multidb.pl` script.
5. Run `pin_multidb.pl -R` to re-create all refresh groups in the secondary databases:
   ```
   % cd BRM_Home/setup/scripts
   % perl pin_multidb.pl -R group
   ```
   where `group` is the name of the group your table belongs to. For information, see "pin_multidb".

Creating Custom Tables That Are Available to All Databases

To make your data dictionary and corresponding tables available to all secondary databases, but not replicate the tables:
1. Create your custom table in the primary database by connecting to the multidatabase CM.

2. Make sure the table DOES NOT use the following naming convention:
   - CONFIG*
   - PLAN*
   - PRODUCT*
   - RATE*
   - UNIQUEN*
   - STRINGS*
   - CHANNEL*
   - SEARCH*
   - ZONE*_T
   - TOD*
   - FOLD*

### Synchronizing Database Data Dictionaries

During normal multidatabase operations, the data dictionaries of the primary and secondary databases are updated automatically if you make the changes using the Storable Class Editor.

For information about adding, deleting, and modifying storable classes in the data dictionary, see "Creating Custom Fields and Storable Classes" in *BRM Developer’s Guide*.

If there is a failure in making the changes to any of the secondary databases, the secondary database data dictionary is in an inconsistent state with respect to the primary database. The failure is reported.

To synchronize the secondary database data dictionary with the primary database data dictionary, perform the following on the primary installation machine:

1. Go to the `BRM_Home/apps/multi_db` directory and open the `pin.conf` file.
2. Verify all entries for the primary database.
3. Go to the `BRM_Home/bin` directory and run the `multi_db_synch_dd` script.
   
The application reports if an error occurs.

### Synchronizing the Database /uniqueness Objects

During normal multidatabase operations, the /uniqueness objects in the primary and secondary databases are updated automatically. BRM uses this object in a multidatabase environment to locate subscribers. It contains a cache of services and must stay synchronized with the service cache in the primary database.

---

**Note:** To determine if the /uniqueness object in a secondary database is out of synchronization, use `sqlplus` to compare the entries in the `uniqueness_t` database table with those in the `service_t` database table. There should be a one-to-one relationship.
If the database tables are not synchronized, run the `load_pin_uniqueness` utility on the secondary databases to update the `/uniqueness` object with the current service data. This utility overwrites existing `/uniqueness` objects in the database.

---

**Note:** This utility needs a configuration (pin.conf) file in the directory from which you run the utility. For information about creating configuration files for BRM utilities, see "Creating Configuration Files for BRM Utilities".

---

1. Go to the `BRM_Home/apps/multi_db` directory.
2. Make sure the `pin_multidb` utility is not running.
   
   The `pin_multidb` utility calls the `load_pin_uniqueness` utility when you configure a multidatabase environment; therefore, you must stop the `pin_multidb` utility before you run the `load_pin_uniqueness` utility.
3. Use the following command to run the `load_pin_uniqueness` utility:
   
   ```
   load_pin_uniqueness
   ```
4. Stop and restart the Connection Manager (CM). See "Starting and Stopping the BRM System".
5. Verify that the `/uniqueness` object was loaded by displaying the `/uniqueness` object by using the Object Browser, or by using the `robj` command with the `testnap` utility.

---

**Changing the Interval for Checking New Accounts**

When the multidatabase system is installed, a time interval is set to check for new accounts in the login table and to verify that the accounts were fully created.

---

**Note:** If you use dblinks for managing the uniqueness table in a multidatabase environment, the following procedure is not required. For more information on dblinks, see "Using database links to manage the uniqueness table."

---

To change that interval:

1. Save a copy of `BRM_Home/apps/multi_db/pin.conf` and open the original file in a text editor.
2. Set the desired frequency value for the `pin_confirm_logins` utility, in minutes:
   
   ```
   - pin_confirm_logins frequency 10
   ```
   
   See the documentation in the `pin.conf` file for information about the frequency value.
3. Save and close the file.
4. Restart `pin_confirm_logins`. 
Changing the Interval for Updating the Distribution Table

A time interval for updating the distribution table is originally set when the multidatabase system is installed. You can change the interval by running the `pin_config_distribution` utility on a different schedule. This utility governs the number of accounts in each database.

1. Save a copy of `BRM_Home/apps/multi_db/pin.conf` and open the original file in a text editor.
2. Set the desired frequency value for the `pin_config_distribution` utility, in minutes:
   ```
   pin_config_distribution frequency 10
   ```
   See the documentation in the `pin.conf` file for information about the frequency value.
3. Save and close the file.
4. Restart `pin_config_distribution`.
5. Verify that the utility was successful by displaying the `/config/distribution` object by using the Object Browser, or by using the `robj` command with the `testnap` utility. (See "Reading an Object and Writing Its Contents to a File" in BRM Developer’s Guide.)

Reinstating Permissions across Schemas after a Database Refresh in Multischema Systems

In a multischema system, the permissions granted to access the other schema tables are revoked every time you replicate a configuration object group from the primary schema to the secondary schema by running the following command:

```perl
perl pin_multidb.pl [-R all | group]
```

Each time you run refresh groups in a multischema system, you must manually grant permissions to enable cross-schema transactions.

To reinstate the permissions granted to access the other schema tables do the following:

1. Connect to the primary schema and run the following command:
   ```sql
   SQL> call grant_permissions ('secondarySchema');
   ```
   where `secondarySchema` is the name of the secondary schema.
2. Connect to the secondary schema and run the following command:
   ```sql
   SQL> call grant_permissions ('primarySchema');
   ```
   where `primarySchema` is the name of the primary schema.
This chapter provides reference information for Oracle Communications Billing and Revenue Management (BRM) multidatabase utilities.
load_config_dist

Use this utility to load the configuration/distribution table for database priorities into the Oracle Communications Billing and Revenue Management (BRM) multidatabase system. You define the database priorities in the `BRM_Home/apps/multi_db/config_dist.conf` file.

For more information, see "Setting Database Priorities".

---

**Note:** You cannot load separate `/config/distribution` objects for each brand. All brands use the same object.

---

**Caution:**

- You must stop the `pin_config_distribution` utility before you run the `load_config_dist` utility.
- The `load_config_dist` utility overwrites the existing database priorities table. If you are updating database priorities, you cannot load new ones only. You must load the complete database priorities table each time you run the `load_config_dist` utility.

---

**Important:** To connect to the Infranet database, the `load_config_dist` utility needs a configuration file in the directory from which you run the utility. See "Creating Configuration Files for BRM Utilities".

---

**Location**

`BRM_Home/bin`

**Syntax**

`load_config_dist`

**Parameters**

There are no input parameters for the `load_config_dist` utility.

**Results**

The progress of the program is displayed on the screen.

---

**Important:** You must restart the Connection Manager to make new resources available. See "Starting and Stopping the BRM System".
load_pin_uniqueness

Use this utility to update the /uniqueness object in a Oracle Communications Billing and Revenue Management (BRM) multidatabase system when the object is not synchronized with the /service object in the database.

For more information, see "Synchronizing the Database /uniqueness Objects".

Caution:
- The pin_multidb utility calls this utility to create a /uniqueness object from a /service object in a multidatabase environment; therefore, stop the pin_multidb utility before you run the load_pin_uniqueness utility.
- The load_pin_uniqueness utility overwrites existing /uniqueness objects. If you are updating /uniqueness objects, you cannot load new /uniqueness objects only. You must load complete sets of /uniqueness objects each time you run the load_pin_uniqueness utility.

Important: To connect to the Infranet database, the load_pin_uniqueness utility needs a configuration file in the directory from which you run the utility. See "Creating Configuration Files for BRM Utilities".

Location

BRM_Home/bin

Syntax

load_pin_uniqueness

Parameters

There are no input parameters for the load_pin_uniqueness utility.

Results

The progress of the program is displayed on the screen.

Important: You must restart the Connection Manager to make new resources available. See "Starting and Stopping the BRM System".
pin_config_distribution

Use this utility to update the multidatabase configuration object (the /config/distribution object). You can update the configuration at a regular interval by running the utility with the cron command.

To run pin_config_distribution directly, edit BRM_Home/apps/multi_db/pin.conf and run the command from that directory. See "Managing a Multidatabase System".

---

**Important:** For multidatabase systems, you must run the utility separately against each database in your system. See "Running Non-MTA Utilities on Multidatabase Systems".

---

**Location**

BRM_Home/bin

**Syntax**

pin_config_distribution

**Parameters**

There are no input parameters for the pin_config_distribution utility.

**Results**

The progress of the program is displayed on the screen.
pin_confirm_logins

Use this utility to update the uniqueness object and the main Oracle Communications Billing and Revenue Management (BRM) database with entries that have been confirmed and to delete logins that are not found.

Note: If you use dblinks for managing the uniqueness table in a multidatabase environment, this utility is not required. For more information on dblinks, see "Using Database Links to Manage the Uniqueness Table" in BRM Installation Guide.

Define the frequency for running this utility in the BRM_Home/apps/multiDB/pin_multidb.conf file.

The pin_multidb script with the -f parameter starts pin_confirm_logins, but you can run the command directly if it is terminated for some reason, or if you need to restart the Connection Manager after reconfiguration.

Location

BRM_Home/apps/multiDB

Syntax

pin_confirm_logins

Parameters

There are no input parameters for the pin_confirm_logins utility.

Results

The progress of the program is displayed on the screen.
**pin_mta_monitor**

Use this utility to monitor and regulate the thread activity of a BRM multithreaded application (MTA) without interrupting the application. You must start the MTA before you run the `pin_mta_monitor` utility.

---

**Note:** To connect to the BRM database, `pin_mta_monitor` uses the configuration file of the MTA that it monitors. For example, if you use `pin_mta_monitor` to track the thread activity of `pin_inv_export`, `pin_mta_monitor` uses the `pin.conf` file of `pin_inv_export`. See "Configuring Your Multithreaded Application" in `BRM Developer’s Guide`.

You must run `pin_mta_monitor` from the same directory from which you run the MTA.

---

**Location**

`BRM_Home/bin`

**Syntax**

```
pin_mta_monitor [mta_application]
```

**(mon)> command**

**Parameters**

This utility runs in an interactive mode. In this mode, you enter single-letter commands to perform individual actions. To run `pin_mta_monitor` for an MTA, you must be at the monitor `((mon)>)` prompt.

Use the following to enter into the monitor prompt.

```
pin_mta_monitor [mta_application]
```

where, `mta_application` is the MTA that `pin_mta_monitor` tracks.

The `((mon)>)` prompt appears.

**Commands**

Table 28–1 lists the commands that you can enter at the monitor prompt.

**Table 28–1  pin_mta_monitor Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>P</code></td>
<td>Starts monitoring <code>mta_application</code>. The utility prints the thread activity for <code>mta_application</code> to <code>stdout</code>. See the discussion on multithreaded applications in <code>BRM System Administrator’s Guide</code>.</td>
</tr>
</tbody>
</table>
Results

This utility logs messages to `stdout`. 

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
</table>
| t [+] number | Regulates the thread load on your MTA utility, where number specifies the number by which you want to increase or decrease the number of threads that `mta_application` uses.  
  - t [+] number adds the specified number of threads to the thread pool of `mta_application`  
  - t [-] number removes the specified number of threads from the thread pool of `mta_application` |
| ? | Displays the `pin_mta_monitor` utility's commands. |
| q | Stops monitoring `mta_application` and exits the utility. |
pin_multidb

Use this script to configure a Oracle Communications Billing and Revenue Management (BRM) multidatabase system. You use this script for the initial configuration of a new multiple database system and to configure additional secondary databases when necessary.

Important

- Edit the pin_multidb.conf file before you run pin_multidb. The pin_multidb.conf file is in BRM_Home/apps/multiDB.
- Run the pin_multidb script on the machine that hosts the Connection Manager (CM) and Data Manager (DM) of the multidatabase system.

For more information about pin_multidb, see "Managing a Multidatabase System".
For more information about installing a multidatabase BRM system, see "Installing a Multidatabase System" in BRM Installation Guide.

Location

BRM_Home/setup/scripts

Syntax

pin_multidb [-i] [-f] [-R all | group] [-r group] [-h]

Parameters

---

**Important:** To fully configure a multidatabase system, run the utility with -i, and then run it again with -f.

---

-i

Initializes the primary and secondary databases.

-f

Finalizes the multidatabase installation.

-R all | group

-R all: Re-creates all refresh groups.

-R group: Re-creates the specified refresh group, where group can be any of the values listed in Table 28–2:

<table>
<thead>
<tr>
<th>Group</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONFIG</td>
<td>Configuration objects</td>
</tr>
<tr>
<td>PRICE</td>
<td>Pricing objects</td>
</tr>
<tr>
<td>UNIQUENESS</td>
<td>Uniqueness objects</td>
</tr>
<tr>
<td>GENERAL</td>
<td>General objects</td>
</tr>
</tbody>
</table>

The refresh frequencies are specified in the pin_multidb.conf file.
**-r group**
Forces a refresh of the specified group of objects, where group can be any of the values listed in Table 28–3:

<table>
<thead>
<tr>
<th>Group</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONFIG</td>
<td>Configuration objects</td>
</tr>
<tr>
<td>PRICE</td>
<td>Pricing objects</td>
</tr>
<tr>
<td>UNIQUENESS</td>
<td>Uniqueness objects</td>
</tr>
<tr>
<td>GENERAL</td>
<td>General objects</td>
</tr>
</tbody>
</table>

The refresh frequencies are specified in the `pin_multidb.conf` file.

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

**Results**
If the `pin_multidb` script doesn’t notify you that it was successful, check that the data in both databases are the same or look in the utility log file (`pin_multidb.log`) to find any errors. The log file is either in the directory from which the utility was started or in a directory specified in the configuration file.
Part VIII describes how to migrate accounts from one database to another in an Oracle Communications Business and Revenue Management (BRM) system. It contains the following chapters:

- Understanding Account Migration Manager
- Installing and Configuring Account Migration Manager
- Migrating Accounts with the Pipeline Manager Running
- Using Account Migration Manager
- Modifying Applications to Work with AMM
- Modifying the Account Migration Manager
- AMT ER Diagram
- Account Migration Manager Utilities
Understanding Account Migration Manager

This document provides an overview of the Oracle Communications Billing and Revenue Management (BRM) Account Migration Manager (AMM), including how AMM interfaces with your BRM databases.

Before using AMM, you should be familiar with the following topics in BRM Concepts:

- BRM concepts. See "Introducing BRM".
- BRM system architecture. See "BRM System Architecture".

About AMM

AMM is an application that migrates accounts and all of their associated objects from one BRM database to another in a multidatabase system. You use AMM to balance loads between existing databases or off-load accounts to a new database.

When to Redistribute Database Loads

You should consider redistributing database loads when:

- One database contains significantly more or fewer accounts than other databases. For example, when you add a database to an existing multidatabase system.
- The number of events per account is significantly greater in one database than in other databases.
- The time it takes to complete a billing run becomes erratic.

Scheduling Account Migration

You achieve optimal migration performance and the smallest impact to your operations if you schedule migrations for maintenance windows. If you need to migrate a large number of accounts but your maintenance window is only a couple of hours, you can perform migrations over a period of several days. The AMM software processes jobs in stages, allowing you to pause and resume account migration without affecting database integrity.

For information on scheduling migration, see "Automating Account Migration".

About Migrating Accounts When the Pipeline Manager is Online

You can configure your system to migrate accounts when the Pipeline Manager is running. In this configuration, the Pipeline Manager temporarily stops call details record (CDR) processing for all accounts that are undergoing migration. For
information, see "Migrating Accounts with the Pipeline Manager Running".

By default, AMM does not support migration when your pipelines are running. You specify whether AMM can migrate accounts while the Pipeline Manager is online by using the controller_N_event_generation parameter in the AMM Infranet.properties file. See "Connecting AMM to Your Databases".

Caution: If you disable this option, you must shut down the Pipeline Manager before you migrate accounts. Otherwise, your pipelines may send account information to the wrong BRM database, causing incorrect account balances and revenue leakage.

How AMM Works

The AMM software migrates accounts from the source database to the destination database based on information you provide in an account search configuration file. Using this file, you specify a group of accounts to migrate based on specific criteria, such as account creation date or account status. The group of accounts and associated objects that meet this criteria forms a job.

Jobs are processed by the AMM software through a queue system, with each job processed in the order received. To improve migration performance, jobs are subdivided into batches, which contain a configurable number of accounts.

Batches are assigned to a configurable number of threads, which process the batches in parallel. Each batch is migrated in a single distributed transaction, during which time activity is prevented on the accounts in the batch. Depending on the success of the batch migration, changes to the databases are either committed or rolled back.

Deleting Migrated Objects

For performance reasons, the AMM software does not automatically remove the migrated objects from the source database. Instead, it flags these migrated objects as invalid to prevent BRM applications from accessing them. You can use the AMM software to manually purge invalid objects from the database at any time.

About Migrating Hierarchical, Sponsorship, and Resource Sharing Accounts

You can configure AMM to migrate hierarchical, sponsorship, and resource sharing groups from one database to another. In this configuration, AMM:

- Searches for accounts in two phases. See "About Searching for Member and Non-Group Member Accounts".
- Organizes accounts that meet the search criteria by account group. See "About Account Groups".
- Migrates entire account groups. See "About Migrating Account Groups".

By default, group migration is disabled. You specify whether to migrate account groups by using the migration_mode entry in the account search file (BRM_Home/apps/amt/account_search.cfg). For information, see "Creating the Account Search Configuration File".
About Searching for Member and Non-Group Member Accounts

When you enable group migration, AMM searches for accounts in two phases:

- **In the first phase, AMM searches for non-group member accounts only.** That is, AMM excludes all remittance, sponsorship, hierarchical, branded, and resource sharing accounts from the account search. Accounts meeting the search criteria are divided into batches, and each batch is flagged as containing non-group member accounts only.

- **In the second phase, AMM searches for accounts belonging to hierarchical, sponsorship, and resource sharing groups only.** If an account meets the search criteria, AMM finds all other account members that are related to the account. These accounts are organized into an account group. See "About Account Groups".

Each account group is divided into batches, which are assigned an account group ID and flagged as containing account group members.

---

**Caution:** When you enable group migration, you must perform extra verification steps to prevent accounts from being severed from their associated account group. See "Checking Account Group Details".

---

**Note:** AMM cannot migrate remittance or brand accounts. For information, see "Brand and Remittance Accounts Not Migrated".

---

All accounts meeting the search criteria: both group member and non-group member accounts, still form one job.

### About Account Groups

An account group consists of all account members that are related to a specific account. When AMM finds an account that meets the search criteria, it finds the parent account and all other child accounts in the group. If one of the accounts is also a member of another group, it finds all members of the other group as well.

*Figure 29–1 One Account Group*
For example, in Figure 29–1, account A meets your search criteria. Because account A is a child in a hierarchy group, AMM finds the parent account and all child accounts in that hierarchy group. Because one hierarchy account member is also a member of a sponsorship group, AMM finds all accounts in the sponsorship group as well. In this example, the account group consists of all accounts in the hierarchy group and the sponsorship group.

About Migrating Account Groups

AMM migrates account group member and non-group member batches in different ways. AMM migrates:

- **Batches containing non-group members in one transaction.** During migration, AMM disables all accounts that belong to a single batch.
  - If the batch migrates successfully, AMM commits the changes to all databases and enables the accounts in the destination database.
  - If the batch fails to migrate, AMM rolls back the changes and re-enables the accounts in the source database.

- **Batches containing account group members by account group ID.** AMM disables all accounts that belong to an account group before migration begins. Once migration starts, AMM monitors whether all batches for the account group migrate successfully.
  - If all batches migrate successfully, AMM commits the changes to all databases and enables the accounts in the destination database.
  - If even one batch in the group fails, AMM leaves all account group members disabled in both source and destination databases. You must fix the error and use AMM to reprocess the job before your BRM system can access the accounts.

AMM Process Overview

Figure 29–2 shows the AMM process overview.

**Figure 29–2  AMM Process Overview**

How AMM migrates accounts:

1. In the account search configuration file, you specify which accounts to migrate and the source and destination databases. See "About the Account Search"
Configuration File”.

2. The pin_amt utility processes the account search configuration file, divides jobs into batches, and places jobs in the queue. See “About the pin_amt Utility”.

3. The AMM Controller allocates batches to threads and passes batches to the AMM Mover. See “About the AMM Controller”.

4. The AMM Mover migrates the batch of accounts. See "About the AMM Mover”.

About the Account Search Configuration File

The account search configuration file specifies the source and destination databases, which accounts to migrate, and the size of each batch. For information on how to create this file, see “Creating the Account Search Configuration File”.

You can also migrate accounts based on custom criteria. See "Creating Custom Account Search Criteria”.

About the pin_amt Utility

The pin_amt utility is a standalone utility that generates account migration jobs for the AMM Controller to process. This utility can perform all of its functions, such as finding accounts to migrate and submitting and enabling jobs, whether the AMM Controller is online or offline.

You use the pin_amt utility to:

■ Start, stop, resume, and monitor the AMM Controller.
■ Find all accounts in the source database that meet your search criteria.
■ Enable account migration jobs in the queue.
■ Delete jobs from the job management tables.
■ Purge invalid objects from the source database.

When you submit an account search file, the pin_amt utility:

1. Searches the source database for all accounts that meet the criteria in the account search configuration file, excluding all remittance accounts and all accounts that are part of branded, hierarchical, sponsorship, and resource sharing groups.

2. Divides the list of account POIDs into batches.

3. Populates the job management tables in the primary, source, and destination databases with the list of account POIDs to migrate. See "About AMM Job Management Tables”.

4. Determines whether group migration is enabled.
   ■ If group migration is enabled, pin_amt proceeds with steps 5 through 9.
   ■ If group migration is disabled, the account search is complete. The AMM Controller can begin processing the job when the job is enabled in the queue. See "About the AMM Controller”.

5. Searches all hierarchy, sponsorship, and resource sharing groups in the source database for accounts that meet the search criteria.

6. When an account meets the search criteria, pin_amt finds all other accounts related to the account and organizes them into an account group.
7. Determines whether the size of the account group exceeds the maximum. If it does, AMM excludes the account group from the job.

---

**Note:** You specify the maximum size of an account group by using the account search file. For information, see "Creating the Account Search Configuration File".

---

8. Divides all members of one account group into batches. All batches in the group are assigned the same group ID and flagged as containing account group members.

9. Populates the job management tables in the primary, source, and destination databases with the list of account POIDs to migrate. See "About AMM Job Management Tables".

### About the AMM Controller

The AMM Controller is a server process that checks the queue for jobs to process. By default, your system contains one AMM Controller, which processes one job at a time. For information about using multiple AMM Controllers, see "About Using Multiple AMM Controllers".

The AMM Controller processes group member and non-group member batches in different ways:

- How AMM Controller Processes Batches with Non-Group Members
- How AMM Controller Processes Batches with Account Group Members

#### How AMM Controller Processes Batches with Non-Group Members

When processing batches that contain non-group members, the AMM Controller:

1. Assigns batches to threads, which run in parallel. Each thread processes one batch at a time. If there are more batches than threads, each thread must process multiple batches in sequence. You use a configuration file to configure the number of AMM Controller threads. See "Connecting AMM to Your Databases".

2. Changes the batch status to IN_PROGRESS. See "About Batch Status Flags".

3. Passes the job ID, batch number, and database link name to the AMM Mover on the destination database for processing. See "About the AMM Mover".

4. Determines if the AMM Mover successfully migrated accounts.
   - If migration is successful, the AMM Controller commits the changes to all databases and changes the batch status to FINISHED. See "About Batch Status Flags".
   - If migration fails, the AMM Controller rolls back the changes and updates the batch status to FAILED. See "About Batch Status Flags".

#### How AMM Controller Processes Batches with Account Group Members

When processing batches that contain account group members, the AMM Controller:

1. Changes the account group status to GROUP_DISABLING. See "About Group Status Flags".

2. Locks the appropriate base table records in the source database, so applications cannot access group member accounts during migration.
3. Marks all account group members in the source database as invalid.

4. Determines whether all account group members were disabled in the source database.
   - If all accounts are successfully disabled, the AMM Controller changes the account group status to GROUP_READY. See "About Group Status Flags".
   - If any accounts failed to disable, the AMM Controller changes the account group status to FAILED. See "About Group Status Flags".

5. Changes the account group status to GROUP_IN_PROGRESS.

6. Passes individual batches in the account group to the AMM Mover for processing.
   a. Assigns an individual batch in the group to a thread.
   b. Changes the batch status to IN_PROGRESS.
   c. Passes the job ID, batch number, and database link name to the AMM Mover. See "About the AMM Mover".
   d. Determines if the AMM Mover successfully migrated the batch and sets the batch status to FINISHED or FAILED.

7. Determines whether all batches in the account group migrated successfully.
   - If all batches migrated successfully, the AMM Controller enables all account group members in the destination database and changes the account group status to GROUP_FINISHED. See "About Group Status Flags".
   - If any batch failed to migrate, the AMM Controller changes the account group status to GROUP_FAILED. See "About Group Status Flags".

---

**Important:** When an account group fails to migrate, all of its accounts remain disabled in the source and destination databases. You must fix the error and migrate the job again before your BRM system can access the accounts.

---

**About the AMM Mover**

The AMM Mover is the process that actually moves accounts from one database to another. Each database contains at least one AMM Mover.

---

**Note:** The AMM Mover performs the following functions regardless of whether a batch contains group-member or non-group member accounts.

When the AMM Mover receives a batch, it:

1. Locks the appropriate base table records in the source database, so applications cannot access accounts during migration.

2. Migrates the objects for an account batch from the source database to the destination database in a single distributed transaction. See "About Distributed Transactions".

3. Marks all migrated objects in the source database as invalid.

4. Updates the account POIDs in the uniqueness table to reflect the account’s new location. For example, if an account is migrated from database 0.0.0.1 to database
About Distributed Transactions

The AMM software migrates each batch of accounts as a single distributed transaction by using a series of database links that were created at installation. This means that changes can be made to the primary, source and destination databases and then committed or rolled back in one transaction, ensuring the integrity of the databases.

Figure 29–3 shows the database links for a multidatabase system with three databases, one AMM Controller, and two threads:

**Figure 29–3 AMM Database Connections**

Account Migration Restrictions

When you run the AMM software, be aware of the following restrictions:

- Account Activity Prevented during Account Migration
- Do Not Rerate Events during Account Migration
- Do Not Alter Account Group Members
- Migration Prevented during Account Activity
- Unique POIDs Required across All Databases
- Some Client Applications May Fail during Account Migration
- AMM Does Not Support Some BRM Components
- Using BRM Reports after Migration
Account Activity Prevented during Account Migration

To prevent applications from accessing or modifying accounts that are being migrated, AMM locks the accounts in Oracle. Only one batch of accounts per thread is locked at a time and only while the accounts are being physically migrated.

---

**Note:** When migrating account groups, AMM locks all accounts in the account group before migration begins.

---

If an application attempts to access a locked account, the Oracle Data Manager (DM) returns a PIN_ERR_INVALID_OBJECT error.

Do Not Rerate Events during Account Migration

Because the AMM software may suspend some events that you want to rerate, you must not rerate pipeline events during account migration.

Do Not Alter Account Group Members

AMM checks account group relationships only when you first create a job, and does not re-verify relationships during the migration process. Therefore, once an account group is included in a migration job, you:

- **Must not** add members to the account group
- **Must not** modify relationships between account group members

If you need to alter an account group after it is included in a job but before the job completes migration, you must:

1. Delete the migration job.
2. Modify the account group.
3. Re-create the account migration job.

Migration Prevented during Account Activity

AMM does not migrate accounts while they are being accessed or modified by BRM or another application. For best performance, stop all account activity before you migrate accounts.

If you cannot restrict all access to the accounts in your databases, AMM can still process account migration jobs. However, AMM does not migrate any batch that contains active accounts. You can check for failed batches and resubmit them for migration once account activity stops.

Brand and Remittance Accounts Not Migrated

You can migrate accounts based on a variety of criteria, including account status, account creation date, and product name. However, the AMM software always excludes brand and remittance account from account migration jobs.

Unique POIDs Required across All Databases

The AMM software can only migrate accounts that have a unique POID. Starting with Infranet Release 6.2 ServicePack1, the multidatabase software automatically creates unique POIDs across all databases in your system.
If your databases contain accounts created both prior to and after you installed Release 6.2 ServicePak1, AMM only migrates those accounts created after you installed 6.2 ServicePak1. For information on how to migrate accounts created prior to 6.2 ServicePak1, contact your Oracle BRM representative.

**Important:** If your BRM system uses a custom POID generation scheme, make sure the sequence number generation algorithm creates unique POIDs across all of your databases.

### Some Client Applications May Fail during Account Migration

BRM client applications, such as Customer Center, Payment Tool, and Self-Care Manager, may generate error messages and fail to commit changes to the database during account migration. For example, if a CSR opens an account in Customer Center just prior to the account being migrated to another database, Customer Center generates an “Unable to process account” and “ERR_INVALID_OBJECT” error message when the CSR attempts to save any changes to the account.

In this case, the CSR must restart Customer Center and access the account again so it retrieves the account’s new location.

**Important:** If your system contains custom client applications that connect to a BRM database and search accounts based on POID, you must modify your application. See "Modifying Custom Client Applications for AMM".

### AMM Does Not Support Some BRM Components

Currently, you cannot use AMM to migrate accounts if your BRM system includes any of the following optional components:

- LDAP Manager
- RADIUS Manager

### Using BRM Reports after Migration

To use BRM Reports after you migrate accounts, you must use BRM Reports Release 6.2 ServicePak1 or later. If you use an earlier version of BRM Reports with AMM, your reports will retrieve and process duplicate data from your source and destination databases.

For example, if an account object is migrated from database 0.0.0.1 to database 0.0.0.2, earlier versions of BRM Reports retrieve the account object from both databases, while BRM Reports 6.2 ServicePak1 and later retrieve the account object only from database 0.0.0.2.

For information on how to modify custom reports to work with AMM, see "Modifying Custom BRM Reports for AMM".
About Using Multiple AMM Controllers

Caution: Implementing multiple AMM Controllers is for advanced users only. Use multiple AMM Controllers only if you understand the impact to migration performance.

Using multiple AMM Controllers allows you to process multiple account migration jobs in parallel. However, you receive performance improvements only if:

- Your system contains more than three databases.
- No two migration jobs use the same database at the same time.

When multiple jobs use the same database, as shown Figure 29–4, migration performance degrades significantly.

Figure 29–4 Concurrent Database Use Performance Degradation

Job 1 and Job 2 use database C at the same time, which decreases performance.

For more information, contact your Oracle BRM representative.

Account Migration Performance

Account migration is resource intensive and can overload your BRM system.

Signs that your system is overloaded during account migration:

- Batch processing times steadily increase, without returning to their initial processing times.
- The AMM software is processing fewer than five accounts per second.
- You receive a distributed transaction time-out error (Oracle error 2049).
- There are a high number of waits for undo segment extension and latch free operations.

If your system exhibits any of these signs, you need to tune your Oracle database. For guidelines, see "Tuning Your Database for Optimal Account Migration Performance" or contact your Oracle BRM representative.

About AMM Job Management Tables

Job management tables are created on your databases during installation and are populated with information about each migration job.
The AMM installer creates the tables listed in Table 29–1 on your databases:

### Table 29–1 AMM Job Management Tables

<table>
<thead>
<tr>
<th>Table name</th>
<th>Database</th>
<th>Description</th>
<th>When populated</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMT_ACCOUNT_BATCH_TABLE_T</td>
<td>Primary only</td>
<td>Stores the list of tables containing data to migrate for a particular batch.</td>
<td>Populated by the AMM Mover when it migrates a particular batch.</td>
</tr>
<tr>
<td>AMT_METADATA_T</td>
<td>All databases</td>
<td>AMM data dictionary. This lists all default BRM tables as well as any custom tables you created. If you add any tables after you install AMM, you must refresh the AMM data dictionary. See &quot;Configuring AMM for New Custom Tables&quot;.</td>
<td></td>
</tr>
<tr>
<td>AMT_POID_TYPE_MAP_T</td>
<td>All databases</td>
<td>Maps the POID type to the table name. This table is static.</td>
<td>During installation and when you run <code>pin_amt_install.pl -m</code> to refresh the AMM data dictionary.</td>
</tr>
</tbody>
</table>

### About Job Status Flags

The AMM software sets jobs to a status listed in Table 29–2. You can see a job’s status by running a `list_jobs` report. For information, see "Monitoring Job Status".

### Table 29–2 Job Status Flags

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISABLED</td>
<td>The job has been submitted but not enabled.</td>
</tr>
<tr>
<td>NOT_PROCESSED</td>
<td>The account migration job is enabled and waiting in the queue to be processed.</td>
</tr>
<tr>
<td>PRE_MIGRATION_WAITING</td>
<td>AMM is notifying the account-router Pipeline Manager to suspend events for all accounts in the job.</td>
</tr>
<tr>
<td>PRE_MIGRATION</td>
<td>The account-router Pipeline Manager acknowledged that it is suspending events for all accounts in the job. AMM is waiting a specified amount of time before starting migration.</td>
</tr>
<tr>
<td>READY</td>
<td>The job is ready to be processed.</td>
</tr>
<tr>
<td>IN_PROGRESS</td>
<td>The job is being processed by the AMM Controller.</td>
</tr>
<tr>
<td>FAILED</td>
<td>The job has been aborted.</td>
</tr>
<tr>
<td>BAL_MIGRATED</td>
<td>The account discount balance migrated successfully.</td>
</tr>
<tr>
<td>POST_MIGRATION_WAITING</td>
<td>AMM is notifying the account-router, source, and destination instances of the Pipeline Manager that the job migrated successfully.</td>
</tr>
<tr>
<td>POST_MIGRATION</td>
<td>The account-router, source, and destination instances of the Pipeline Manager acknowledged that they updated all account information in their caches.</td>
</tr>
<tr>
<td>PRE_JOB_RECYCLE</td>
<td>AMM is notifying the account-router Pipeline Manager to resume processing events for accounts in the job.</td>
</tr>
<tr>
<td>JOB_RECYCLE</td>
<td>The account-router Pipeline Manager acknowledged that it is ready to begin reprocessing events for accounts in the job.</td>
</tr>
<tr>
<td>FINISHED</td>
<td>The job has completed successfully.</td>
</tr>
</tbody>
</table>
About Batch Status Flags

AMM sets account batches to a status listed in Table 29–3. You can check a batch’s status by running a `job_details` report. For information, see "Checking Job Details".

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOT_PROCESSED</td>
<td>The account batch has not yet been migrated.</td>
</tr>
<tr>
<td>IN_PROGRESS</td>
<td>The account batch is currently being migrated.</td>
</tr>
<tr>
<td>FAILED</td>
<td>The account batch failed to migrate. All changes to the databases have been rolled back.</td>
</tr>
<tr>
<td>FINISHED</td>
<td>The account batch migrated successfully. All changes to the databases have been committed.</td>
</tr>
</tbody>
</table>

Table 29–4  Group Status Flags

AMM sets account groups to a status listed in Table 29–4. You can check an account group’s status by running a `group_details` report. For information, see "Checking Account Group Details".

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOT_PROCESSED</td>
<td>The account group has not yet been migrated.</td>
</tr>
<tr>
<td>GROUP_DISABLING</td>
<td>All account group members are being disabled in the source database. That is, AMM is marking all account group members as invalid to prevent applications from accessing those accounts.</td>
</tr>
<tr>
<td>FAILED</td>
<td>AMM did not disable all account group members in the source database.</td>
</tr>
<tr>
<td>GROUP_READY</td>
<td>All account group members were successfully disabled in the source database. AMM can begin processing batches.</td>
</tr>
<tr>
<td>GROUP_IN_PROGRESS</td>
<td>The account group is currently being migrated.</td>
</tr>
<tr>
<td>GROUP_FAILED</td>
<td>The account group failed to migrate to the destination database.</td>
</tr>
<tr>
<td>GROUP_FINISHED</td>
<td>The account group migrated successfully.</td>
</tr>
</tbody>
</table>
This document explains how to install and configure the Oracle Billing and Revenue Management (BRM) Account Migration Manager (AMM) software.

Before you read this document, you should be familiar with BRM, multidatabase, and AMM concepts and architecture. See the following topics:

- "Introducing BRM" in BRM Concepts
- "BRM System Architecture" in BRM Concepts
- "Putting Together Your BRM System" in BRM Installation Guide
- Understanding Account Migration Manager

System Requirements

AMM is available for Oracle databases and the HP-UX IA64, Linux, AIX, and Solaris operating systems. For information on disk space requirements for the HP-UX IA64, AIX, Linux, and Solaris operating systems, see "Disk Space Requirements" in BRM Installation Guide.

Software Requirements

Before installing AMM, you must install:

- Oracle9i or Oracle 10g or Oracle 11g database software. You must also install the following Oracle9i or 10g or 11g components:
  - JServer
  - Oracle9i JVM version 1.5.0_06
  - PL/SQL
  - SQL*Plus
- Java Development Kit (JDK).
- Third-Party software, which includes the PERL libraries and JRE required for installing BRM components. See "Installing the Third-Party Software" in BRM Installation Guide.
- BRM. See "Installing BRM" in BRM Installation Guide.
- MultiDB Manager. See "Installing a Multidatabase System" in BRM Installation Guide.
If you plan to migrate accounts when the Pipeline Manager is running, you must also install the following BRM software:

- **Pipeline Manager.** See "Installing Pipeline Manager" in BRM Installation Guide.
- **Account Synchronization Data Manager (DM).** See "Installing and Configuring the Account Synchronization DM" in BRM Installation Guide.
- **Suspense Manager.** See "Installing Suspense Manager" in BRM Configuring Pipeline Rating and Discounting.
- **Rated Event (RE) Loader.** See "Installing Rated Event Loader" in BRM Configuring Pipeline Rating and Discounting.

---

**Important:** These components include changes that enable you to migrate accounts while your pipelines are running. You must install the latest versions of these components to migrate accounts.

---

**Installing AMM**

The instructions in this section assume that you have four machines in your multidatabase environment as shown in Figure 30–1.

*Figure 30–1 Installing AMM*

Installing AMM requires the following general steps:

1. **Configuring All Primary and Secondary Database Machines**
2. **Installing the AMM Software on the Primary Installation Machine**
3. **Configuring your Oracle DM to Check for Invalid Objects**
4. **Connecting AMM to Your Databases**

**Configuring All Primary and Secondary Database Machines**

Before you can begin installation, you must first configure your primary and secondary databases for account migration.

Perform the following steps on *each* database machine in your multidatabase system:
1. Using SQL*Plus, log in to your database as the SYSTEM user and grant database linking privileges to the BRM user pin:

   ```
   $ sqlplus system/manager@databaseAlias
   SQL> grant create database link to pin;
   Grant succeeded.
   ```

2. Verify that JServer is installed on your system:

   ```
   SQL> SELECT object_name, object_type FROM all_objects WHERE object_type = 'PACKAGE' and object_name = 'DBMS_JAVA';
   ```

   If JServer is installed on your system, you receive the following:

   ```
   OBJECT_NAME    OBJECT_TYPE
   ------------------------  -----------
   DBMS_JAVA       PACKAGE
   ```

   If JServer is not installed, you receive the following:

   ```
   no rows selected
   ```

3. Install JServer if it is not already installed on your system.
   a. Add the following entry to the Oracle initSID.ora file ($ORACLE_HOME/dbs/initSID.ora):

      ```
      java_pool_size=20971520
      ```

   b. Restart Oracle so that the database instance is initialized with your changes.

   c. Install JServer manually by running the Oracle initjvm script:

      ```
      $ sqlplus sys/change_on_install@databaseAlias
      SQL> @$ORACLE_HOME/javavm/install/initjvm.sql
      ```

      For information, see your Oracle documentation.

4. Modify the entries listed in Table 30–1 in the Oracle initSID.ora file ($ORACLE_HOME/dbs/initSID.ora):

<table>
<thead>
<tr>
<th>Entry</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>global_names</td>
<td>Specifies whether a database link is required to have the same name as the database to which it connects. Set this to False.</td>
</tr>
<tr>
<td>utl_file_dir</td>
<td>Specifies the location of the Oracle utl_file. Set this to a writable directory for the user oracle.</td>
</tr>
</tbody>
</table>

5. If your database contains multiple schemas, ensure that you assigned unique database instance names to each schema in your system:
   a. Open the tnsnames.ora file in a text editor.
   b. Assign a unique database instance name to each schema. For example, for the first schema:

      ```
      Alias1 = (DESCRIPTION = (ADDRESS = (PROTOCOL=TCP)(HOST=DatabaseHostName)(PORT= 1521))
      (CONNECT_DATA = (SID =DatabaseSID)) )
      ```
Installing AMM

For the second schema:

Alias2 = (DESCRIPTION = (ADDRESS = (PROTOCOL=TCP)(Host=DatabaseHostName)(Port= 1521)) (CONNECT_DATA = (SID =DatabaseSID)) )

c. Save and close the file.

6. Restart Oracle so that the database instance is initialized with your changes.

Installing the AMM Software on the Primary Installation Machine

To install AMM, perform the following steps on the primary installation machine:

1. Log in as user pin and make sure the environment variables listed in Table 30–2 are set correctly in the .cshrc or .profile file:

<table>
<thead>
<tr>
<th>Environment variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PATH</td>
<td>Make sure this includes the path to SQL*Plus and Xterm.</td>
</tr>
<tr>
<td>DISPLAY</td>
<td>Set this to WorkstationName:0.0. Note: You must set this variable in order to have the AMM software open an Xterm window that displays the AMM Controller’s status in real time.</td>
</tr>
<tr>
<td>PIN_HOME</td>
<td>Specifies the directory where BRM is installed. The default is /opt/portal/7.4.</td>
</tr>
<tr>
<td>ORACLE_HOME</td>
<td>Specifies the directory where Oracle is installed.</td>
</tr>
</tbody>
</table>

2. Download the software to a temporary directory (temp_dir).

Important:
- If you download to a Windows workstation, use FTP to copy the .bin file to a temporary directory on your UNIX server.
- You must increase the heap size used by the Java Virtual Machine (JVM) before running the installation program to avoid “Out of Memory” error messages in the log file. See "Increasing Heap Size to Avoid ‘Out of Memory’ Error Messages" in BRM Installation Guide.

3. Go to the directory where you installed the Third-Party package and source the source.me file.

Caution: You must source the source.me file to proceed with installation, otherwise “suitable JVM not found” and other error messages appear.

Bash shell:
source source.me.sh

C shell:
source source.me.csh

4. Go to the temp_dir directory and enter this command:

7.4_AccountMigrationMgr_platform_32_opt.bin

where platform is the operating system name.

---

**Note:** You can use the -console parameter to run the installation in command-line mode. To enable a graphical user interface (GUI) installation, install a GUI application such as X Windows and set the DISPLAY environment variable before you install the software.

---

5. Follow the instructions displayed during installation. The default installation directory for AMM is opt/portal/7.4.

---

**Note:** The installation program does not prompt you for the installation directory if BRM or AMM is already installed on the machine and automatically installs the package at the BRM_Home location.

---

6. Go to the directory where you installed the AMM package and source the source.me file:

Bash shell:
source source.me.sh

C shell:
source source.me.csh

7. Go to the BRM_Home/setup directory and run the pin_setup script.

8. Verify that the BRM_Home/setup/scripts/pin_multidb.conf file contains accurate information about each database in your system. The pin_amt_install script uses the information in this file to set up your AMM environment.

9. For optimal performance, store your AMM job management tables and indexes in their own physical tablespaces. Map the following entries to four separate physical tablespaces by modifying the BRM_Home/setup/scripts/pin_tables.values file:

$PIN_CONF_TBLSPACE14
$PIN_CONF_TBLSPACE15
$PIN_CONF_TBLSPACEX16
$PIN_CONF_TBLSPACEX17

See “Database Configuration and Tuning” in BRM Installation Guide.

10. Log in as user pin, go to the BRM_Home/setup/scripts directory, and run the pin_amt_install script:

```
# su - pin
% cd BRM_Home/setup/scripts
% perl pin_amt_install.pl
```
11. Verify that installation was successful by checking the AMM installation log file
       (BRM_Home/setup/scripts/pin_amt_install.log).
12. Restart your BRM processes. For information, see "Starting and Stopping the BRM
       System".

Configuring your Oracle DM to Check for Invalid Objects

During account migration, the AMM software marks all migrated objects in the source
database as invalid. To prevent BRM applications from accessing these objects, you
must configure your Oracle Data Manager (DM) to check for invalid objects during
account searches.

Caution: If you do not make this change, BRM applications retrieve
duplicate data from your source and destination databases. For
example, if an account is migrated from database 0.0.0.1 to database
0.0.0.2, the application retrieves the account object from both
databases.

To configure your system to check for invalid objects, perform the following on every
machine containing an Oracle DM:

1. Add the following line to the BRM_Home/sys/dm_oracle/pin.conf file:
   - dm dm_nul_poid_db_chk 1
2. Restart dm_oracle so that your Oracle DM is initialized with your changes. See
   "Starting and Stopping the BRM System".

Connecting AMM to Your Databases

During installation, the AMM installer generates an Infranet.properties configuration
file that specifies how to connect to your databases and the number and configuration
of your AMM Controllers. The installer populates the connection parameters with
values from your multidatabase pin_multidb.conf file and provides default values for
setting up one AMM Controller. If you want more than one AMM Controller, you
must create additional AMM Controller entries.

Before migrating accounts, verify that the AMM Infranet.properties file contains:

- Accurate information for connecting to all databases in your system.
- The optimal number of AMM Controllers for your system. See "About Using
  Multiple AMM Controllers".
- The optimal number of threads per AMM Controller for your system.

To modify your AMM configuration file, perform the following on the primary
installation machine:

1. Go to the BRM_Home/sys/amt directory and open the Infranet.properties file in a
text editor.
2. Verify that the configuration entries reflect your environment. When checking the
   Infranet.properties file, make sure it contains:
   - A set of 0.0.0.x entries for each database in your system.
For example, if you have three databases in your system, the file should contain a set of entries prefixed by 0.0.0.1, a set prefixed by 0.0.0.2, and a set prefixed by 0.0.0.3.

- A set of controller_N entries for the number of AMM Controllers you want in your system.

The AMM installer automatically creates a set of Controller_1 entries and populates them with default values. If you’d like additional controllers, create a set of Controller_2 entries, Controller_3 entries, and so on. In general, increasing the number of AMM Controllers permits multiple jobs to be processed in parallel.

Note: The content of the Infranet.properties file conforms to the Java properties file conventions. Options are key-value pairs separated by the equal sign (=). For example, 0.0.0.1_user_name=pin and 0.0.0.1_primary=true.

Table 30–3 shows the parameters used in defining the Infranet.properties configuration file.

<table>
<thead>
<tr>
<th>Entry</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0.0.x_user_name</td>
<td>String</td>
<td>Specifies the Oracle user name for the specified database.</td>
</tr>
<tr>
<td>0.0.0.x_user_password</td>
<td>String</td>
<td>Specifies the Oracle user password for the specified database.</td>
</tr>
<tr>
<td>0.0.0.x_instance_name</td>
<td>String</td>
<td>Specifies the SQL*Net database alias name you assigned in the tnsnames.ora file.</td>
</tr>
<tr>
<td>0.0.0.x_primary</td>
<td>true or false</td>
<td>Flag that indicates whether the database is the primary database. For the primary database, set this to true. For all secondary databases, set this to false.</td>
</tr>
<tr>
<td>0.0.0.x_mover_log_file_dir</td>
<td>Path name</td>
<td>Specifies the directory of the AMM Mover log file on the specified database. Important: This path must match the path specified in the utl_file_dir entry of the initSID.ora file.</td>
</tr>
<tr>
<td>0.0.0.x_mover_log_file_flag</td>
<td>Y or N</td>
<td>Specifies whether you want the AMM Mover to create log files on the specified database.</td>
</tr>
<tr>
<td>0.0.0.xgrp_srch_log_file_flag</td>
<td>Y or N</td>
<td>Specifies whether you want AMM to create a log file for the account group stored procedure. Note: The stored procedure finds all account group members related to a specific account.</td>
</tr>
<tr>
<td>0.0.0.xgrp_srch_log_file_dir</td>
<td>Path name</td>
<td>Specifies the directory for the account group stored procedure log file.</td>
</tr>
<tr>
<td>controller_N_log_directory</td>
<td>Path name</td>
<td>Specifies the directory in which to create the AMM Controller log file.</td>
</tr>
<tr>
<td>controller_N_port_number</td>
<td>Integer &gt; 1024</td>
<td>Specifies the TCP/IP port number for the connection between the pin_amt utility and the AMM Controller. Each AMM Controller instance requires a unique port number.</td>
</tr>
<tr>
<td>controller_N_server</td>
<td>String</td>
<td>Specifies the host name of the machine that runs the AMM Controller.</td>
</tr>
<tr>
<td>Entry</td>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>controller_N_thread_count</td>
<td>Positive integer</td>
<td>Specifies the number of AMM Controller processing threads. For optimal performance, the number of AMM Controller threads should be 1 to 2 times the number of CPUs on the destination database that are dedicated to AMM.</td>
</tr>
<tr>
<td>controller_N_syslog_priority</td>
<td>1 (low) through 7 (high)</td>
<td>AMM Controller log message priority threshold. Messages with a lower priority are suppressed.</td>
</tr>
<tr>
<td>pin_amt_log_directory</td>
<td>Path name</td>
<td>Specifies the path to the pin_amt log file.</td>
</tr>
<tr>
<td>controller_N_event_generation</td>
<td>true or false</td>
<td>Specifies whether the AMM Controller migrates accounts when your pipelines are running. This configures AMM to notify Pipeline Manager when accounts are migrated. The default is false.</td>
</tr>
<tr>
<td>controller_N_concurrent_job_number</td>
<td>Positive integer</td>
<td>Specifies how many jobs the AMM Controller starts concurrently. The default is 20.</td>
</tr>
<tr>
<td></td>
<td>Note: This entry is required only if the controller_N_event_generation entry is set to Y.</td>
<td></td>
</tr>
<tr>
<td>controller_N_hold_period</td>
<td>Positive integer</td>
<td>Specifies how long the AMM Controller waits, in minutes, before it starts to migrate accounts. This provides time for your pipelines to flush any EDRs targeted for accounts that are being migrated. The default is 120.</td>
</tr>
<tr>
<td></td>
<td>Note: This entry is required only if the controller_N_event_generation entry is set to Y.</td>
<td></td>
</tr>
<tr>
<td>controller_N_amt_queue_owner_name</td>
<td>String</td>
<td>Specifies the user that created the acknowledgment queue.</td>
</tr>
<tr>
<td></td>
<td>Note: This entry is required only if the controller_N_event_generation entry is set to Y.</td>
<td></td>
</tr>
<tr>
<td>controller_N_amt_queue_name</td>
<td>String</td>
<td>Specifies the name of the acknowledgment queue. The AMM Controller dequeues pipeline acknowledgment events from this queue.</td>
</tr>
<tr>
<td></td>
<td>Note: This entry is required only if the controller_N_event_generation entry is set to Y.</td>
<td></td>
</tr>
</tbody>
</table>
Sample Infranet.properties File
The following sample Infranet.properties file is for a system with:

- One database with two schemas: Schema1Alias and Schema2Alias
- One AMM Controller
- Pipeline migration enabled

```
# Connection entries for the Primary Database
0.0.0.1_user_name=pin
0.0.0.1_user_password=pin
0.0.0.1_instance_name=Schema1Alias
0.0.0.1_primary=true
0.0.0.1_mover_log_file_dir=./mover/log
0.0.0.1_mover_log_file_flag=y

# Connection entries for the Secondary Database
0.0.0.2_user_name=pin
0.0.0.2_user_password=pin
0.0.0.2_instance_name=Schema2Alias
0.0.0.2_primary=false
0.0.0.2_mover_log_file_dir=./mover/log
0.0.0.2_mover_log_file_flag=y

# Controller entries
```

Sample Infranet.properties File
The following sample Infranet.properties file is for a system with:

- One database with two schemas: Schema1Alias and Schema2Alias
- One AMM Controller
- Pipeline migration enabled

```
# Connection entries for the Primary Database
0.0.0.1_user_name=pin
0.0.0.1_user_password=pin
0.0.0.1_instance_name=Schema1Alias
0.0.0.1_primary=true
0.0.0.1_mover_log_file_dir=./mover/log
0.0.0.1_mover_log_file_flag=y

# Connection entries for the Secondary Database
0.0.0.2_user_name=pin
0.0.0.2_user_password=pin
0.0.0.2_instance_name=Schema2Alias
0.0.0.2_primary=false
0.0.0.2_mover_log_file_dir=./mover/log
0.0.0.2_mover_log_file_flag=y

# Controller entries
```

Table 30–3 (Cont.) AMM Infranet.properties Values

<table>
<thead>
<tr>
<th>Entry</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
</table>
| infranet.login.type    | 1 or 0| Specifies whether AMM requires a user name and password to log in to the Connection Manager (CM).
|                        |       | ■ 1 specifies that AMM must provide a user name and password.
|                        |       | ■ 0 specifies that AMM uses a “trusted” login that comes through a CM Proxy, for example, and doesn’t require a user name and password in the properties file. |
|                        |       | Note: This entry is required only if the controller_N_event_generation entry is set to Y. |
| infranet.connection    | String| Specifies the full URL for connecting to the primary CM. |
|                        |       | ■ For a type 1 login, the URL must include a user name and password. You must specify the service name and service POID (“1”), but the CM determines the database number. For example: `pcp://root.0.0.0.1:password@hostname:12009/service/pcm_client` |
|                        |       | ■ For a type 0 login, the URL requires a full POID, including the database number. |
|                        |       | Note: This entry is required only if the controller_N_event_generation entry is set to Y. |
| publish_migrated_objects | String | Specifies the storable classes whose objects are stored in the MIGRATED_OBJECTS_T cross-reference table. You can list multiple storable classes by using a comma (,) as a delimiter. For example: `/service,/billinfo,/payinfo`. |
|                        |       | Note: This entry is required only if you integrate AMM with your external application using Oracle Application Integration Architecture (Oracle AIA) in a multischema environment. |
Configuring AMM for Additional Databases

You must reconfigure the AMM software whenever you add a database to an existing multidatabase system.

To configure AMM for additional databases, perform the following on the primary installation machine:

1. Delete all existing account migration jobs in the queue:
   ```
   % pin_amt -d JobID
   ```

2. Log in as user pin and run the pin_amt_install.pl script:
   ```
   # su - pin
   % cd BRM_Home/setup/scripts
   % perl pin_amt_install.pl
   ```

What’s Next?

AMM installation is now complete.

To migrate accounts while the Pipeline Manager is online, you must perform additional configuration steps. See “Configuring Your System to Migrate Accounts When the Pipeline Manager Is Running”.

If your system doesn’t use batch rating or you plan to shut down Pipeline Manager during account migration, you can begin migrating accounts. See “Using Account Migration Manager”.

Configuring the JDBC Driver Jar File for BRM

If you are using the Oracle 11g database, you must update the JDBC reference to ojdbc5.jar.

To update the JDBC reference:

1. Open the BRM_Home/bin/pin_amt file in a text editor.
2. Find the classes12.zip entry and add an entry for ojdbc5.jar in the same line as classes12.zip.
   For example:
   ```
   ORACLE_Home/jdbc/lib/ojdbc5.jar
   ```
3. Save and close the file.
This script reinstall the job management tables on your databases and re-creates all of your database links.

**Configuring AMM for New Custom Tables**

AMM migrates data to and from the tables listed in the AMM data dictionary. This list includes all BRM tables and any custom tables that were on your system when you installed AMM. If you add any tables after you install AMM, you must update the AMM data dictionary.

To update the AMM data dictionary, perform the following on your primary installation machine:

1. Log in as user **pin** and go to the **BRM_Home/setup/scripts** directory:
   ```bash
   % su - pin
   % cd BRM_Home/setup/scripts
   ```

2. Run the **pin_amt_install.pl** script with the **-m** parameter:
   ```bash
   % perl pin_amt_install.pl -m
   ```

   This script updates the AMM data dictionary tables, AMT_META_DATA_T and AMT_POID_TYPE_MAP_T, on all databases in your system.

**Tuning Your Database for Optimal Account Migration Performance**

To tune your database for optimal account migration performance:

- Use cost-based optimization.
- Set the number of Oracle rollback segments to approximately two times the number of AMM Controller threads. Multiple rollback segments allow Oracle to automatically allocate one rollback segment for each transaction.
- Set the `transactions_per_rollback_segment` parameter in the `$ORACLE_HOME/dbs/initSID.ora` file to a small number. For best results, set it to 1 or 2.
- Set the initial size for the rollback segment to twice the total data volume of the account batch. You can estimate the account batch data volume by multiplying the average number of events per batch with the batch size.
- Set the optimal size for the rollback segments to twice the initial size.
- Set the next size for the rollback segments to half the initial size.
- Set the maximum number of extends to unlimited.

For more information, see your Oracle documentation. For information on additional ways to tune your database for AMM, contact your Oracle BRM representative.
This document explains:

- How the Account Migration Manager (AMM) software migrates accounts when Pipeline Manager is running.
- How to configure your Oracle Communications Billing and Revenue Management (BRM) system to migrate accounts when Pipeline Manager is running.

Before you read this document, you should be familiar with AMM concepts and configuration. See the following documents:

- Understanding Account Migration Manager
- Installing and Configuring Account Migration Manager

**About Migrating Accounts When Pipeline Manager Is Online**

When you migrate accounts while Pipeline Manager is online, your pipelines stop processing any EDRs that apply to accounts undergoing migration. Your pipelines continue processing all other EDRs.

*Figure 31–1* shows AMM interaction with the Pipeline Manager.
To coordinate account migration with your pipelines:

- AMM notifies the pipelines about a job’s migration status by sending business events. See "About Notifying the Pipelines About Account Migration".
- The pipelines notify AMM about EDR processing status by sending acknowledgment events. See "About Notifying AMM About EDR Processing".
- The account-router Pipeline Manager suspends, recycles, and routes EDRs. See "About the Account Router Instance of the Pipeline Manager".

**How AMM Interacts with Your Pipelines during Account Migration**

The following steps outline how AMM interacts with your pipelines when processing account migration jobs:

1. AMM fetches a configurable number of migration jobs. See "About Starting Multiple Jobs Concurrently".
2. AMM notifies the account-router Pipeline Manager to hold EDRs for all accounts in a job.
3. The account-router Pipeline Manager begins holding all EDRs for the specified list of accounts and sends an acknowledgment to AMM. See "About Suspending Call Records".
4. AMM waits a specified amount of time before migrating accounts. See "About Waiting before Migrating Accounts".
5. AMM migrates all accounts in the job. See "About the AMM Controller".
6. AMM determines whether the job migrated successfully.
   - If migration completed successfully, AMM notifies the account router, source, and destination instances of the Pipeline Manager.
Migrating Accounts with the Pipeline Manager Running

About Migrating Accounts When Pipeline Manager Is Online

If migration failed, AMM does not send any notification to your pipelines and job processing stops.

Important: When migration fails, your pipelines continue to suspend all EDRs for the specified accounts. You must fix the problem and re-migrate the job before the pipeline can begin reprocessing suspended EDRs.

About Waiting before Migrating Accounts

After the account-router Pipeline Manager begins suspending EDRs, AMM waits a configurable amount of time before migrating a job. This provides time for your pipelines to flush any EDRs targeted for accounts in the migration job.

The default wait time is 120 minutes. You specify how long the AMM Controller waits before migrating accounts by using the Controller_N_hold_period entry in the AMM Infranet.properties file. For information, see “Connecting AMM to Your Databases”.

About Starting Multiple Jobs Concurrently

You can minimize the amount of time AMM spends in the waiting period by configuring AMM to start multiple migration jobs concurrently. In this configuration, AMM:

1. Fetches a configurable number of jobs.
2. Notifies the account-router Pipeline Manager to hold EDRs for multiple jobs.
3. Starts the timer for each job.
4. Once the waiting period is over, AMM migrates jobs individually.

This increases the number of jobs in the queue that are ready to be migrated.

You specify how many jobs an AMM Controller processes concurrently by using the Controller_N_concurrent_job_number entry in the AMM Infranet.properties file. For information, see "Connecting AMM to Your Databases".

About Notifying the Pipelines About Account Migration

AMM notifies your pipelines about account migration by sending a series of business events through the Account Synchronization architecture.

Note: If configured to do so, AMM also notifies any external applications.
About Migrating Accounts When Pipeline Manager Is Online

About AMM Business Events

AMM generates the five business events listed in Table 31–1 to notify the account router, source, and destination instances of the Pipeline Manager when account migration occurs:

<table>
<thead>
<tr>
<th>Event</th>
<th>Recipient</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HoldCDRProcessing</td>
<td>Account-Router Pipeline Manager</td>
<td>Notifies the account-router Pipeline Manager to suspend all EDRs for a specified list of accounts.</td>
</tr>
<tr>
<td>ResumeCDRProcessing</td>
<td>Account-Router Pipeline Manager</td>
<td>Notifies the account-router Pipeline Manager to resume processing all suspended and new EDRs for the specified list of accounts.</td>
</tr>
<tr>
<td>MigrateAcct</td>
<td>Account-Router Pipeline Manager</td>
<td>Notifies the account-router Pipeline Manager and any external applications to update the account database location for the specified list of accounts.</td>
</tr>
<tr>
<td>MigrateSource</td>
<td>Source Pipeline Manager</td>
<td>Notifies the source Pipeline Manager that all accounts in the job migrated successfully.</td>
</tr>
<tr>
<td>MigrateDestination</td>
<td>Destination Pipeline Manager</td>
<td>Notifies the destination Pipeline Manager that all accounts in the job migrated successfully. The destination pipeline then reads account information from the database.</td>
</tr>
</tbody>
</table>

Table 31–1 AMM Business Events

About Sending AMM Business Events to the Pipelines

AMM sends business events to the pipelines by using a series of Account Synchronization queues. Each instance of the Pipeline Manager contains its own queue, which is dedicated to receiving business events from BRM and AMM.

**Note:** If configured to do so, AMM also sends business events to a queue for external applications.

AMM sends business events to a Pipeline Manager as follows:

1. AMM sends an event to the primary Connection Manager (CM).
2. The primary CM sends the event to the Account Synchronization architecture.
3. The Account Synchronization architecture uses its `ifw_sync_queuenames` file to publish the business event to the appropriate queue.
4. The Pipeline Manager’s DAT_Listener module dequeues the event and then forwards it to the appropriate pipeline data module.
You configure your system to send AMM business events to your pipelines by:

- Connecting AMM to the primary CM.
- Creating an Oracle database queue for each instance of the Pipeline Manager.
- Configuring Account Synchronization to publish AMM business events to your queues.
- Configuring each instance of the Pipeline Manager to dequeue AMM business events from its associated Account Synchronization queue.

Figure 31–2 shows the AMM business events process described. For more information, see “Configuring AMM to Send Business Events to Your Pipelines”.

About Notifying AMM About EDR Processing

Your pipelines notify AMM when it begins holding EDRs, reprocessing EDRs, or updating account data by sending acknowledgment events through a dedicated acknowledgment queue.

About Acknowledgment Events

Each instance of the Pipeline Manager generates acknowledgment events when the following listed in Table 31–2 occur:

<table>
<thead>
<tr>
<th>Pipeline instance</th>
<th>Sends acknowledgments when...</th>
</tr>
</thead>
</table>
| Account router Pipeline Manager | ■ It begins suspending EDRs for a specified migration job.  
■ It resumes processing EDRs for a specified migration job.  
■ It completes an update of account locations in pipeline memory. This occurs after a migration job completes successfully. |
| Pipeline Managers connected to a BRM database | ■ It completes an update of account locations in pipeline memory. This occurs when accounts in a job are successfully migrated to its associated database.  
■ It receives a MigrateSource event from AMM to indicate that accounts were successfully migrated away from its associated database. |
About Sending Acknowledgments to AMM

Each instance of the Pipeline Manager sends acknowledgment events to AMM by using a dedicated acknowledgment queue as shown in Figure 31–3.

Each Pipeline Manager sends acknowledgments as follows:

1. The DAT AccountBatch module sends an acknowledgment event to the DAT Listener module.
2. The DAT Listener module publishes the event to the acknowledgment queue.
3. The AMM Controller dequeues the event.

Figure 31–3 Sending Pipeline Acknowledgment Events to AMM

To configure your pipelines to send acknowledgments to AMM, you must:

- Configure the DAT Listener module in each instance of the Pipeline Manager to publish acknowledgment events.
- Create a single database queue that is dedicated to acknowledgment events. All instances of the Pipeline Manager use this single queue.
- Configure the AMM Controller to dequeue events from the acknowledgment queue.

For more information, see "Configuring Your Pipelines to Send Acknowledgments to AMM".

About the Account Router Instance of the Pipeline Manager

The account router instance of the Pipeline Manager is used in multibase systems to route EDRs to the correct instance of the Pipeline Manager. For example, EDRs targeted for accounts that reside in database 3, are routed to the Pipeline Manager instance associated with database 3.

When configured for migration, the account-router Pipeline Manager also performs the following:

- Suspends EDRs targeted for accounts that are undergoing migration. See "About Suspending Call Records".
You configure the account-router Pipeline Manager for migration by creating three separate pipelines:

- A **routing pipeline** that routes EDRs to the appropriate instance of the Pipeline Manager and suspends any EDRs targeted for accounts undergoing migration. This pipeline must include the FCT_AccountRouter module, set to router mode; the FCT_PreSuspense module; the FCT_Reject module; and the FCT_Suspend module.

- A **pre-recycling pipeline** that processes previously suspended EDRs, determines whether an EDR is targeted for an account undergoing migration, and then routes the EDR to the appropriate output stream. This pipeline must include the FCT_AccountRouter module, set to recycle mode.

- A **re-suspending pipeline** that automatically suspends all EDRs. This pipeline must include the FCT_PreSuspense module, the ISC_Migration iScript, the FCT_Reject module, and the FCT_Suspend module.

You must also configure the account router data pool to pass migration status information to your pipelines. Figure 31–4 shows the necessary pipelines and account router data pool.

Figure 31–4 Account Router Pipeline Manager

For information on how to configure the account-router Pipeline Manager, see "Configuring Your Account-Router Pipeline Manager".

**About Suspending Call Records**

The account-router Pipeline Manager initially routes and suspends call records in the **routing pipeline**.
After AMM notifies the account-router Pipeline Manager that a job is being migrated, the routing pipeline performs the following:

1. (Optional) The FCT_CallAssembly module assembles EDRs that were split into multiple records.

   **Important:** Any call assembling must occur in the account router instance of the Pipeline Manager and not in other instances.

   For more information, see "FCT_CallAssembling" in *BRM Configuring Pipeline Rating and Discounting*.

2. (Optional) The FCT_DuplicateCheck module checks whether EDRs have been previously rated by the Pipeline Manager.

   **Important:** Any checking for duplicate EDRs must occur in the account router instance of the Pipeline Manager and not in other instances.

   See "FCT_DuplicateCheck" in *BRM Configuring Pipeline Rating and Discounting*.

3. The FCT_PreSuspense module adds suspense-related data to the EDR.

4. The FCT_AccountRouter module, set to **Router** mode:
   - Flags the EDR for the target Pipeline Manager.
   - Determines whether an EDR is for an account undergoing migration. If it is, FCT_AccountRouter flags the EDR for suspension. See "FCT_Account" in *BRM Configuring Pipeline Rating and Discounting*.

5. The FCT_Reject module routes EDRs with a specified error status, such as warning or critical, to the suspense output stream.

6. The FCT_Suspense module determines whether an EDR is flagged for suspension. If it is, FCT_Suspense places the EDR in a separate suspense output stream, where it is eventually loaded into the BRM database by the Suspense Event (SE) Loader.

   For more information about recycling suspended EDRs in BRM, see "About the EDR Recycling Features" in *BRM Configuring Pipeline Rating and Discounting*.

   **Note:** You can use either standard recycling or Suspense Manager with AMM.

### About Reprocessing Suspended Call Records

After AMM successfully migrates a job, it calls the PCM_OP_SEARCH_RECYCLE opcode to recycle previously suspended EDRs through the pipeline. Then, the account-router Pipeline Manager recycles suspended EDRs through the **pre-recycling pipeline** and the **re-suspending pipeline**.

The account-router Pipeline Manager recycles EDRs as follows:

1. In the pre-recycling pipeline, the FCT_AccountRouter module, set to **Recycle** mode, determines whether an EDR is targeted for an account that is being migrated by a new job.
If the account is being migrated by a new job, FCT_AccountRouter flags the EDR for suspension and routes the EDR to a separate suspense output stream, where it is processed by the re-suspending pipeline.

If the account is not being migrated, FCT_AccountRouter flags the EDR for the appropriate instance of the Pipeline Manager. The EDR is then rated by the target Pipeline Manager.

2. The re-suspending pipeline automatically routes EDRs to a separate suspense output stream, which is eventually loaded into the BRM database by Suspense Event (SE) Loader.
   a. The FCT_PreSuspense module adds suspense-related data to the EDR.
   b. The ISC_Migration iScript automatically flags the EDR for suspension.
   c. The FCT_Reject module routes EDRs with a specified error status to the suspense output stream.
   d. The FCT_Suspense module routes the EDR to a suspense output stream, which is eventually loaded into the BRM database by SE Loader.

For more information about Suspense Manager, see "About Suspense Manager" in BRM Configuring Pipeline Rating and Discounting.

Configuring Your System to Migrate Accounts When the Pipeline Manager Is Running

You configure your BRM system to migrate accounts when your pipelines are online by:

1. Configuring Your Account-Router Pipeline Manager
2. Configuring BRM to Handle Suspended EDRs
3. Configuring AMM to Send Business Events to Your Pipelines
4. Configuring Your Pipelines to Send Acknowledgments to AMM

Configuring Your Account-Router Pipeline Manager

To configure your account router instance of the Pipeline Manager, perform the following:

- Configuring Your Routing Pipeline
- Configuring Your Pre-Recycling Pipeline
- Configuring Your Re-Suspending Pipeline
- Configuring the Data Pool

Configuring Your Routing Pipeline

You configure your routing pipeline to route and suspend EDRs by using the following pipeline modules:

- **FCT_PreSuspense.** To make suspense fields queryable in Suspense Management Center, set the following FCT_PreSuspense registry entries:
  - Use the **Active** entry to enable this module.
  - Use the **QueryableFields** entry to specify the tables and fields that you can perform queries on in Suspense Management Center.
See "FCT_PreSuspense" in BRM Configuring Pipeline Rating and Discounting.

- **FCT_AccountRouter** set to **Router** mode. To flag EDRs for suspension and for the appropriate Pipeline Manager, set the following FCT_AccountRouter registry entries:
  - Use the **Active** entry to enable this module.
  - Use the **Mode** entry to specify **Router** mode.
  - Use the **Streams** entry to map EDRs to the appropriate output stream.

For more information, see "FCT_AccountRouter" in BRM Configuring Pipeline Rating and Discounting.

- **FCT_Reject**. To route EDRs with a specified error status to the suspense output stream:
  - Use the **Active** entry to enable this module.
  - Set the **UseRejectStream** entry to **True**. This sends EDRs to the reject stream.
  - Use the **MinErrorSeverity** entry to reject EDRs that have the specified error severity.
  - Use the **StreamMap** entry to map errors to specific output streams.

**Important:** You must also configure an instance of the Out_Reject module for rejected EDRs. All rejected EDRs must be set to the suspense output stream. See "OUT_Reject" in BRM Configuring Pipeline Rating and Discounting.

For more information about FCT_Reject, see "FCT_Reject" in BRM Configuring Pipeline Rating and Discounting.

- **FCT_Suspense**. To send EDRs to the suspense output stream, set the following FCT_Suspense registry entries:
  - Use the **Active** entry to enable this module.
  - Use the **SuspenseCreateStream** entry to specify the output stream for suspended EDRs.
  - Use the **SuspenseUpdateStream** entry to specify the output stream for recycled EDRs.
  - Use the **DataConnection** entry to specify how to connect to the BRM database.

See "FCT_Suspense" in BRM Configuring Pipeline Rating and Discounting.

If you want your pipelines to assemble EDRs or check for duplicate EDRs, you must also use the FCT_CallAssembly and FCT_DuplicateCheck modules in the routing pipeline. See "FCT_CallAssembling" and "FCT_DuplicateCheck" in BRM Configuring Pipeline Rating and Discounting.

### Configuring Your Pre-Recycling Pipeline

You configure your pre-recycling pipeline to recycle or suspend EDRs by using the FCT_AccountRouter module set to **Recycle** mode. Make sure you also set the following FCT_AccountRouter registry entries:

- Use the **Active** entry to enable this module.
- Use the **Mode** entry to specify **Recycle** mode.
Use the **Streams** entry to map EDRs to the appropriate output stream.
For more information, see "FCT_AccountRouter" in *BRM Configuring Pipeline Rating and Discounting*.

**Configuring Your Re-Suspending Pipeline**

You configure your re-suspending pipeline to automatically suspend all EDRs by using the following pipeline modules:

- **FCT_PreSuspense.** To make suspense fields queryable in Suspense Management Center, set the following FCT_PreSuspense registry entries:
  - Use the **Active** entry to enable this module.
  - Use the **QueryableFields** entry to specify the tables and fields that you can perform queries on in Suspense Management Center.
For more information, see "FCT_PreSuspense" in *BRM Configuring Pipeline Rating and Discounting*.

- **ISC_Migration.** To automatically flag all EDRs for suspension, set the following ISC_Migration registry entries:
  - Use the **Active** entry to enable this module.
  - Use the **Filename** entry to specify the path to the ISC_Migration file.
For more information, see "ISC_Migration" in *BRM Configuring Pipeline Rating and Discounting*.

- **FCT_Reject.** To route EDRs with a specified error status to the suspense output stream:
  - Use the **Active** entry to enable this module.
  - Set the **UseRejectStream** entry to **True**. This sends EDRs to the reject stream.
  - Use the **MinErrorSeverity** entry to reject EDRs that have the specified error severity.
  - Use the **StreamMap** entry to map errors to specific output streams.

**Important:** You must also configure an instance of the Out_Reject module for rejected EDRs. All rejected EDRs must be set to the suspense output stream. See "OUT_Reject" in *BRM Configuring Pipeline Rating and Discounting*.

For more information about FCT_Reject, see "FCT_Reject" in *BRM Configuring Pipeline Rating and Discounting*.

- **FCT_Suspense.** To send EDRs to the suspense output stream, set the following FCT_Suspense registry entries:
  - Use the **Active** entry to enable this module.
  - Use the **SuspenseCreateStream** entry to specify the output stream for suspended EDRs.
  - Use the **SuspenseUpdateStream** entry to specify the output stream for recycled EDRs.
  - Use the **DataConnection** entry to specify how to connect to the BRM database.
Configuring the Data Pool

You configure the account-router Pipeline Manager data pool to pass account migration data to your pipelines by using the following pipeline data modules:

- **DAT_AccountBatch** stores AMM business events. In addition to setting the standard connection registry entries:
  - Set the **UseAsRouter** entry to True. This is required.
  - (Optional) Use the **PrintAMTDData** entry to specify whether to print AMM data to a log file. You can use this data for troubleshooting.
  - (Optional) Use the **PrintAMTJobData** entry to specify whether to print data about one migration job to a log file. You can use this data for troubleshooting.

For more information, see "DAT_AccountBatch" in *BRM Configuring Pipeline Rating and Discounting*.

- **DAT_BalanceBatch**. To provide accurate account balances during migration, set the following DAT_BalanceBatch registry entries:
  - Use the **IntegrateConnection** entry to specify how to connect to the pipeline database. This entry points to the **Login** registry section.
  - Use the **InfranetConnection** entry to specify how to connect to the BRM database. This entry points to the **LoginInfranet** registry section.
  - Use the **ListenerDataModule** entry to specify how to connect to the DAT_Listener module. This entry points to the **Listener** registry section.

For more information, see "DAT_BalanceBatch" in *BRM Configuring Pipeline Rating and Discounting*.

- **DAT_Listener**. To retrieve business events from BRM and send acknowledgment events directly to the acknowledgment queue, set the following DAT_Listener registry entries:
  - Use the **InfranetConnection** entry to specify how to connect to the database that contains your queues. This entry points to the **LoginInfranet** registry section.
  - Use the **AckQueueName** entry to specify the name of the acknowledgment queue.
  - Use the **QueueName** entry to specify the name of the Account Synchronization queue that stores the AMM business events.

For more information, see "DAT_Listener" in *BRM Configuring Pipeline Rating and Discounting*.

Configuring BRM to Handle Suspended EDRs

BRM offers both the default standard recycling feature and the optional Suspense Manager feature to recycle EDRs. For a comparison of the two, see "About the EDR Recycling Features" in *BRM Configuring Pipeline Rating and Discounting*.

AMM works with both standard recycling and Suspense Manager.

Both standard recycling and Suspense Manager allow you to:

- Load suspended EDRs into the BRM database.
Configuring Your System to Migrate Accounts When the Pipeline Manager Is Running

- View, edit, write-off, or recycle suspended EDRs.
- Retrieve suspended EDRs from the BRM database.

You must configure your pipeline before you can migrate accounts with the pipeline running. For information on how to configure your pipeline, see “Configuring Standard Recycling” in BRM Setting Up Pricing and Rating. If you have purchased Suspense Manager, see the Suspense Manager documentation for further configuration instructions.

Configuring AMM to Send Business Events to Your Pipelines

To configure AMM to send business events to your pipelines, perform the following:

1. Connecting AMM to the Primary CM
2. Configuring Account Synchronization
3. Configuring Your Pipelines to Dequeue AMM Business Events

Connecting AMM to the Primary CM

You connect AMM to the primary Connection Manager (CM) so that AMM can send business events to the Account Synchronization architecture, where they are eventually routed to your pipelines.

You connect AMM to the primary CM by using the infranet.connection and infranet.login.type parameters in the AMM Infranet.properties file. For information, see “Connecting AMM to Your Databases”.

Configuring Account Synchronization

You configure the Account Synchronization architecture to send AMM business events to the appropriate instance of the Pipeline Manager by performing the following:

1. Configuring Account Synchronization to send BRM events to your pipelines. To do this, follow the instructions in “About Sending Account Data to Pipeline Manager” (see BRM Installation Guide).
2. Creating queues for sending AMM business events to your pipelines. You must create a queue for each Pipeline Manager instance in your system. See "Configuring Your Account Synchronization Queues for AMM Business Events".
3. Mapping the AMM business events to your Oracle database queues. See "Mapping AMM business events to your queues".

Configuring Your Account Synchronization Queues for AMM Business Events

The Account Synchronization framework uses a set of Oracle database queues to send both BRM events and AMM business events to your pipelines.

Each instance of the Pipeline Manager must have its own database queue. For example, if your system contains three BRM databases, Account Synchronization requires a total of four queues. That is, one for each of these instances:

- Account-router Pipeline Manager
- Pipeline Manager for BRM database 1
- Pipeline Manager for BRM database 2
- Pipeline Manager for BRM database 3
If your system does not already contain a queue for each instance of the Pipeline Manager, you can create additional queues by using the Account Synchronization `pin_ifw_sync_oracle` utility. For more information about creating queues, see "About Sending Account Data to Pipeline Manager" in BRM Installation Guide.

**Mapping AMM business events to your queues**

You map which types of events Account Synchronization sends to your queues by using the `ifw_sync_queuenames` file (`$BRM_Home/sys/dm_ifw_sync/ifw_sync_queuenames`). This file lists all queues in your system as well the events to route to each one.

You configure the `ifw_sync_queuenames` file to map AMM business events, in addition to your BRM events, to each of your queues by using the following syntax:

```
QueueName@DatabaseLink
{  Criteria }
```

Where:

- `QueueName` specifies the name of the queue.
- `DatabaseLink` specifies the database link for connecting to queues on other databases. Provide a link only for queues that reside on a separate database from which the Account Synchronization DM connects.
- `Criteria` specifies which events to send to the queue. You can configure the Account Synchronization DM to send all business events, only events from a specific database, or only specific event types.

For example, assume your BRM system contains three databases and four queues, as shown in Figure 31–5:

---

**Figure 31–5  Sample Pipeline Account Synchronization Architecture**

---
In this system, you need to configure the `ifw_sync_queuenames` file so that the Account Synchronization architecture sends:

- **HoldCDRProcessing**, **ResumeCDRProcessing**, and **MigrateAcct** events to the queue for the account-router Pipeline Manager.
- **MigrateSource** and **MigrateDestination** events to each queue connected to a BRM database.

For more information about AMM business events, see "About Notifying the Pipelines About Account Migration".

In this example, the `ifw_sync_queuenames` file requires the following additional entries:

```plaintext
ROUTER_Q
{
  HoldCDRProcessing
  ResumeCDRProcessing
  MigrateAcct
}

Q_1 # local database queue
{
  MigrateSource
  MigrateDestination
}

Q_2@database_link_1 # remote database
{
  MigrateSource
  MigrateDestination
}

Q_3@database_link_2 # remote database
{
  MigrateSource
  MigrateDestination
}
```

### Configuring Your Pipelines to Dequeue AMM Business Events

You must configure each instance of the Pipeline Manager to retrieve AMM business events from the Account Synchronization queue. To do this, connect the DAT_Listener modules to your Account Synchronization queues by setting the following registry entries:

- Use the **InfranetConnection** entry to specify how to connect to the database that contains the Account Synchronization queue. This entry points to the `LoginInfranet` registry section.
- Use the **QueueName** entry to specify the name of the Account Synchronization queue that holds the AMM business events.

See "DAT_Listener" in BRM Configuring Pipeline Rating and Discounting.

### Configuring Your Pipelines to Send Acknowledgments to AMM

You configure your pipelines to send acknowledgment events to a centralized queue, where they are retrieved by the AMM Controller, by performing the following:

1. **Creating the Acknowledgment Queue**
2. **Connecting AMM Directly to Your Acknowledgment Queue**
3. Configuring Your Pipelines to Send Acknowledgment Events

Creating the Acknowledgment Queue

You create a centralized acknowledgment queue for sending events from your pipelines to the AMM Controller.

You create the acknowledgment queue by using the `pin_ifw_sync_oracle` utility. Enter the following commands at a UNIX prompt:

```bash
% su - pin
% cd BRM_Home/apps/pin_ifw_sync
% pin_ifw_sync_oracle.pl create [-l username/password@DatabaseAlias] [-q queue_name] [-t queue_table]
```

The utility creates a database queue named IFW_SYNC_QUEUE and a queue table named IFW_SYNC on the specified database. To use non-default names, use the `-q` and `-t` options to specify names for the queue and queue table.

**Important:** In multidatabase systems, all queues and queue tables must use unique names. You must also make sure the acknowledgment queue is accessible by the `pin` user.

For more information, see the following documents:

- "pin_ifw_sync_oracle" in BRM Installation Guide
- "About Sending Account Data to Pipeline Manager" in BRM Installation Guide

Connecting AMM Directly to Your Acknowledgment Queue

You connect the AMM Controller to the acknowledgment queue so that it can retrieve acknowledgment events.

You connect the AMM Controller to the acknowledgment queue by using the `controller_N_amt_queue_name` and `controller_N_amt_queue_owner_name` entries in the AMM Infranet.properties file. See "Connecting AMM to Your Databases".

Configuring Your Pipelines to Send Acknowledgment Events

You configure your pipelines to send acknowledgment events to AMM by configuring the DAT_Listener module in each Pipeline Manager.

Configure the DAT_Listener registry entries in each instance of the Pipeline Manager to specify the following:

- Use the `InfranetConnection` entry to specify how to connect to the database that contains your acknowledgment queue.
- Use the `AckQueueName` entry to specify the name of the acknowledgment queue. This is the queue you created in "Creating the Acknowledgment Queue".

For more information, see "DAT_Listener" in BRM Installation Guide.
Using Account Migration Manager

This document describes how to use the Oracle Communications Billing and Revenue Management (BRM) Account Migration Manager (AMM) software to migrate accounts from one database to another.

Overview of Account Migration Tasks

Migrating accounts includes the following general tasks. Although you can perform some tasks at any time, the following order is recommended:

1. Create the account search configuration file.
2. Submit the account migration job.
3. For account group migration, run a group_details report to verify that each account group includes all account members.

Caution: You must verify that the job includes all accounts in the account group. Any missing accounts will be stored in a separate database from the account group, which severs the account’s relationship with the group.

4. Enable the account migration job.
5. Start the AMM Controller.
6. Monitor the job’s progress.
7. Fix account migration failures, when necessary.
8. Purge the migrated accounts from the source database.
9. Stop the AMM Controller.

Creating the Account Search Configuration File

You use the account search configuration file to specify the source and destination databases, the search criteria, the maximum number of accounts in a job, and the number of accounts in each batch.

AMM can search for accounts that meet five default criteria:

- Account creation date
- Account status
- Billing day of month
Creating the Account Search Configuration File

- Product name
- POID

If you’d like to migrate accounts that meet some other custom criteria, see "Creating Custom Account Search Criteria".

To create an account search configuration file:

1. Copy the sample account search configuration file (`BRM_Home/apps/amt/account_search.cfg`) and save it with another name. Use this file, which contains all of the configuration entries, as a template.

2. Edit the entries listed in Table 32–1 in the file.

---

**Note:** Only the source database, destination database, batch size, and one other entry is required. If you do not want to use an entry, leave it blank.

---

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>src_database</td>
<td>Specifies the source database, which is the database from which you are</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>migrating accounts. For example, enter <code>0.0.0.1</code>. This value must match one</td>
<td></td>
</tr>
<tr>
<td></td>
<td>of the database numbers specified in the <code>Infranet.properties</code> file.</td>
<td></td>
</tr>
<tr>
<td>dest_database</td>
<td>Specifies the destination database, which is the database to which you are</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>migrating accounts. For example, enter <code>0.0.0.2</code>. This value must match one</td>
<td></td>
</tr>
<tr>
<td></td>
<td>of the database numbers specified in the <code>Infranet.properties</code> file.</td>
<td></td>
</tr>
<tr>
<td>start_creation_date</td>
<td>Use this parameter to migrate accounts that were created in a specific date</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>range. AMM migrates accounts created between midnight (00:00:00) on the</td>
<td></td>
</tr>
<tr>
<td></td>
<td>start date and 23:59:59 on the end date. For example, to migrate accounts</td>
<td></td>
</tr>
<tr>
<td></td>
<td>created after midnight on August 1, 2004, enter <code>08/01/2004</code>.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Important:</strong> If you set this parameter, you must also set the</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>end_creation_date</code> parameter.</td>
<td></td>
</tr>
<tr>
<td>end_creation_date</td>
<td>Use this parameter to migrate accounts that were created in a specific date</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>range. AMM migrates accounts created between midnight (00:00:00) on the</td>
<td></td>
</tr>
<tr>
<td></td>
<td>start date and 23:59:59 on the end date. For example, to migrate accounts</td>
<td></td>
</tr>
<tr>
<td></td>
<td>created on or before 11:59:59 p.m. on August 10, 2004, enter <code>08/10/2004</code>.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Important:</strong> If you set this parameter, you must also set the <code>start_creation_date</code> parameter.</td>
<td></td>
</tr>
<tr>
<td>migration_mode</td>
<td>Specifies whether to migrate account groups. When AMM finds an account that</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>belongs to a hierarchy, sponsorship, or resource sharing group, AMM</td>
<td></td>
</tr>
<tr>
<td></td>
<td>migrates all accounts related to that account.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>IncludeAccountGroup</strong> specifies to migrate account groups.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>ExcludeAccountGroup</strong> specifies to exclude account groups from migrations.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The default is <strong>ExcludeAccountGroup</strong>.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Important:</strong> If you set this parameter, you must also set the <code>max_group_size</code> parameter.</td>
<td></td>
</tr>
<tr>
<td>max_group_size</td>
<td>Specifies the maximum size of an account group that AMM can migrate. If an</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>account group exceeds the maximum number of accounts, AMM excludes the</td>
<td></td>
</tr>
<tr>
<td></td>
<td>account group from the job. The default is <strong>100</strong>.</td>
<td></td>
</tr>
</tbody>
</table>
Creating the Account Search Configuration File

Using Account Migration Manager

3. Save the file.

Sample Account Search Configuration File

The following sample account search configuration file specifies to:

- Migrate accounts from database 0.0.0.1 to database 0.0.0.2.
- Migrate in batches of 50 accounts.
- Migrate only non-member accounts.
- Migrate accounts that meet the following criteria:
  - Created between January 1, 2004 and June 31, 2004
  - Have an active account status
  - Purchased the Product 1b - Email Account product

```plaintext
src_database=0.0.0.1
dest_database=0.0.0.2
start_creation_date=01/01/2004
end_creation_date=06/31/2004
migration_mode=ExcludeAccountGroup
max_group_size=
product_name=Product 1b - Email Account
account_status=Active
bill_day_of_month=
max_accounts=
batch_size=50
poid_list=
```

Table 32–1 (Cont.) account_search.cfg Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>product_name</td>
<td>Migrates accounts that purchased the specified product. For example, Product 1b - Email Account.</td>
<td>No</td>
</tr>
<tr>
<td>account_status</td>
<td>Migrates accounts based on the specified account status.</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>- <strong>Active</strong> specifies to migrate active accounts only.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- <strong>Inactive</strong> specifies to migrate inactive accounts only.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- <strong>Closed</strong> specifies to migrate closed accounts only.</td>
<td></td>
</tr>
<tr>
<td>bill_day_of_month</td>
<td>Migrates accounts that have the specified billing day of month. You can specify any number from 1 through 31. For example, enter 4 to migrate all accounts that are billed on the 4th of the month.</td>
<td>No</td>
</tr>
<tr>
<td>max_accounts</td>
<td>Specifies the maximum number of accounts to move in a job.</td>
<td>No</td>
</tr>
<tr>
<td>batch_size</td>
<td>Specifies the number of accounts in each batch. You can specify any amount from 1 through 1,000. However, for optimal performance, set this to an integer between 50 and 100. Important:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Using a batch size of more than 50 accounts does not improve performance.</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>- If you set this to a number greater than 100, you must increase the size of your Oracle rollback segments. For more information, contact your Oracle BRM representative.</td>
<td></td>
</tr>
<tr>
<td>poid_list</td>
<td>Migrates accounts based on the POID. Use comma separators, for example, 22860, 22861, 22862. Limit the number of accounts to 1,000 or less.</td>
<td>No</td>
</tr>
</tbody>
</table>
Submitting the Account Search File

When you submit an account search file, the `pin_amt` utility searches the source database and populates the job management tables on the primary, source, and destination databases with a list of accounts meeting the specified criteria.

1. Submit your account search information to the `pin_amt` utility:

   ```
   % pin_amt -s AccountSearchFile
   submitted job
   job_id=30
   ```

   The `pin_amt` utility notifies you if it successfully submitted the file and gives you the job ID number.

2. Write down the job ID number, because you will need it later.

Enabling Migration Jobs in the Queue

The AMM Controller can only begin processing an account migration job after it’s enabled in the queue.

To enable a job in the queue, enter this command:

```
% pin_amt -e JobID
enabled job
```
Monitoring Account Migration

Using Account Migration Manager

32-5

The *pin_amt* utility notifies you that the AMM Controller is up or down. If the AMM Controller is down or cannot be started, check the AMM Controller log file for more information.

Checking the AMM Controller Log File

The AMM Controller log file contains a detailed list of all transactions executed by the AMM Controller. This log is created in the directory specified in the `controller_N_log_directory` entry of the *Infranet.properties* file. You can open the log file by using a text editor.

Monitoring the AMM Controller in Real Time

You can use the *pin_amt* utility to see what the AMM Controller is doing in real time.

To monitor the AMM Controller in real time, enter this command:

```
% pin_amt -c log [-a ControllerID]
```

A separate Xterm window opens. For best viewing, set the Xterm width to 120. If an Xterm window fails to open, make sure your *DISPLAY* environment variable is set correctly.

Monitoring Account Migration

You can monitor the status of jobs in the queue by running three special AMM reports: *list_jobs*, *job_details*, and *group_details*.

Monitoring Job Status

The *list_jobs* report provides the status of each job in the queue, including the number of batches that failed to migrate.

To run the *list_jobs* report, enter this command:

```
% pin_amt -r list_jobs
```

Sample output from a *list_jobs* report:

```
Tue Mar 12
page 1

AMT jobs

Total  Failed  Succ.  Proc.  Job
Account  Account

Job Name  User Name  Job ID  batches  batches  batches  Job
Status    time[sec]  time    Accounts

Note: If your system contains multiple AMM Controllers, use the -a option to specify which AMM Controller to check.
```
Monitoring Account Migration

-------------

If any batches failed, you can see greater detail on why the batch failed by running the `job_details` report.

### Checking Job Details

The `job_details` report provides detailed information about a job’s status, including why a batch failed.

To run the `job_details` report, enter this command:

```
% pin_amt -r job_details
```

enter job id:

Sample output from a `job_details` report:

```
Tue Mar 12 page 1

AMT job details

--- --- -------- ------------ ------------------- --------
Job ID batches Status date time[sec] Accounts
--- --- -------- ------------ ------------------- --------
3 1 FINISHED 03/01/2002 18:42
25 50
2 FAILED ORA-02055: distributed 03/01/2002 18:42 5 0
update operation failed; rollback required
ORA-02049: timeout: distributed transaction waiting for lock
ORA-06512: at "PIN.AMT_MV", line 454
ORA-06512: at line 1

3 FINISHED 03/01/2002 18:42
25 50
4 FINISHED 03/01/2002 18:43
25 50
5 FINISHED 03/01/2002 18:43
25 50
6 FINISHED 03/01/2002 18:44
25 50
7 FINISHED 03/01/2002 18:44
25 50
8 FINISHED 03/01/2002 18:45
25 50
```

The report lists any error messages from the Oracle database. For information, see the Oracle documentation.
Checking Account Group Details

The `group_details` report lists the accounts in each account group and provides information about each group’s migration status. You use this information to verify that all account members are included in a group.

---

**Caution:** All accounts in a hierarchy, sponsorship, or resource sharing group must reside in the same database. Any accounts separated from a parent account will no longer be associated with the account group.

---

To run the `group_details` report, enter this command:

```
% pin_amt -r group_details
```

enter job id:

enter group id:

Sample output from a `group_details` report:

```
Tue Mar 12 AMT group details

Job ID  Account  Batch  Group  Status  Group ID  Status
------- ------ ------- -------- --------- ----------------
   3     1      FINISHED 1  NOT_PROCESSED

Tue Mar 12 AMT group member details

Account  batches  Accounts ID  Accounts DB
------- ---------- ------------
   1       17009   2
   1       17289   2
   1       16489   2
   1       17313   2
   1       16465   2
   1       17066   2
```

Handling Account Migration Failures

An account batch may fail for several reasons. The most common reasons are as follows:

- An application is accessing or modifying the data you are attempting to migrate.
- The primary, source, and/or destination database is down.

Finding Debugging Information

For information on why a batch failed, you can run a `job_details` report or check any of the following files, which are located in the directories you specified in the `Infranet.properties` file.

- AMM installation log file (`pin_amt_install.log`)
Purging Migrated Objects from the Source Database

- AMM Controller log file (controller_N_YYYYMMDDhhmm.log)
- pin_amt log file (pin_amt.log)
- AMM configuration file (Infranet.properties)
- Account search configuration file (account_search.cfg)
- AMM Mover log files (amt_migrate_JobID_BatchNumber.log)
- AMM delete log file (amt_delete_JobID_BatchNumber.log)

If you need assistance in resolving migration failures, send these files along with any additional information about the problem to your Oracle BRM representative.

Reprocessing Failed Batches

To reprocess a batch that failed:

1. Fix the problem.
2. Change the status of the batch from FAILED to NOT PROCESSED:
   ```
   % pin_amt -b JobID:BatchNumber
   ```
3. Enable the job in the queue again:
   ```
   % pin_amt -e JobID
   ```

   The AMM Controller processes all batches that have a NOT PROCESSED status and ignores batches with a FINISHED status.

Purging Migrated Objects from the Source Database

After you successfully migrate your accounts, you can improve your overall system performance by purging the migrated (invalid) objects from your source database. Also, because the purging process uses only one thread, purges accounts sequentially, and doesn’t affect data used by BRM, you can purge accounts at any time.

To purge successfully migrated objects from the source database, enter this command:

```
pin_amt -p SourceDatabase
```

Deleting Jobs from the Source Database

You can use the delete option to:

- Remove both failed and successfully migrated jobs from your databases
- Free up disk space

The delete option performs the actions listed in Table 32–2:

<table>
<thead>
<tr>
<th>Job type</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failed jobs</td>
<td>Deletes the job from the AMM job management tables.</td>
</tr>
<tr>
<td>Successfully migrated jobs</td>
<td>■ Deletes the job from the AMM job management tables.</td>
</tr>
<tr>
<td></td>
<td>■ Deletes account-related data from the source database.</td>
</tr>
</tbody>
</table>

To delete a job, run the `pin_amt` script with the delete option:
Stopping the AMM Controller

You can stop the AMM Controller at any time. If you stop the AMM Controller while it is processing a batch, it finishes the batch before stopping.

To stop the AMM Controller, enter this command:

```
% pin_amt -c stop
controller is stopped
controller_id=1
```

Pausing and Resuming Account Migration

If a job contains a large number of accounts, but you only have a limited amount of time in which to migrate accounts, you can migrate the job in stages. The AMM software allows you to start an account migration job and then pause it when your window of opportunity is over. When you reach the next window of opportunity, you can resume the job where it left off.

To pause an account migration job, enter this command:

```
% pin_amt -c pause
paused controller
controller_id=1
```

To resume an account migration job, enter this command:

```
% pin_amt -c continue
continued controller
controller_id=1
```

Automating Account Migration

You can use an external scheduler, such as `cron`, to automate account migration during your maintenance window. If you need to migrate a large number of accounts, you can set up `cron` to stop and restart account migration at specific times.

For example, scheduling account migration for every Sunday from 2:00 a.m. to 4:00 a.m. requires these tasks:

1. Stop the AMM Controller.
2. Create your account search configuration files.
3. Submit your jobs.
4. Enable your jobs in the queue.
5. Configure one `cron` job to start the AMM Controller every Sunday at 2:00 a.m. and check for errors. See "AMM Return Codes and Messages".
6. Configure a second `cron` job to stop the AMM Controller every Sunday at 4:00 a.m. and check for errors.
Modifying Applications to Work with AMM

This document provides information on modifying custom client applications and custom reports to work with the Oracle Communications Billing and Revenue Management (BRM) Account Migration Manager (AMM) software. It also provides the list of AMM return codes that are used for automating AMM.

This document is for system administrators, database administrators, and programmers. Modifying applications for AMM requires knowledge of the following:

■ BRM error handling
■ BRM database schema
■ BRM opcodes
■ BRM Reports

Modifying Custom Client Applications for AMM

Custom client applications that connect to a specific database and try to access an object based on a POID may receive an PIN_ERR_INVALID_OBJ error if the object was migrated to another database. You must modify any custom client applications to handle that error and then perform a global search to find the object’s correct location.

To obtain the correct POID of a storable object, modify your application to call the PCM_OP_GLOBAL_SEARCH opcode from its exception handling routine.

This example shows a call to the PCM_OP_GLOBAL_SEARCH opcode when the PIN_ERR_INVALID_OBJ error is returned from the Oracle DM:

/* Error? */

if (PIN_ERR_IS_ERR(ebufp)) {
    PIN_ERR_LOG_EBUF(PIN_ERR_LEVEL_ERROR,
                       "sample_read_obj_search error", ebufp);
}

/* Call the DM to do a global search.*/

PCM_OP(ctxp, PCM_OP_GLOBAL_SEARCH, 0, flistp, &r_flistp, ebufp);
return;

The following opcodes return the PIN_ERR_INVALID_OBJ error when a POID specified in an input flist is invalid:

■ PCM_OP_READ_OBJ
■ PCM_OP_READ_FLDS
Modifying Custom BRM Reports for AMM

After account migration, any custom BRM reports created prior to Infranet Release 6.2 ServicePak1 might retrieve and process duplicate data from your source and destination databases. For example, if an account object is migrated from database 0.0.0.1 to database 0.0.0.2, your report might retrieve the account object from both databases.

To prevent this, use the Oracle Business Intelligence Publisher to add the following line to the WHERE clause of each custom report’s query:

\[ \text{TABLE}_T \cdot \text{POID_DB} > 0 \]

where \( \text{TABLE}_T \) satisfies these conditions:

- It is a database table used by the report.
- It is one of the tables moved from the source database to the destination database when account data is migrated.
- It is associated with every record the report must retrieve.

---

**Note:** If a single table doesn’t satisfy the last condition, add the same line for several tables that together satisfy the last condition.

---

AMM Return Codes and Messages

AMM uses the return codes and messages listed in Table 33–1. To automate account migration, you can modify your external application to check for the following return codes and respond appropriately.

### Table 33–1 AMM Return Codes and Messages

<table>
<thead>
<tr>
<th>Return code number</th>
<th>Return code</th>
<th>Return message</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>CONTROLLER_STARTED_SUCC</td>
<td>controller is started</td>
</tr>
<tr>
<td>101</td>
<td>CONTROLLER_STOPPED_SUCC</td>
<td>controller is stopped</td>
</tr>
<tr>
<td>102</td>
<td>CONTROLLER_PAUSED_SUCC</td>
<td>paused controller</td>
</tr>
<tr>
<td>103</td>
<td>CONTROLLER.Continued_SUCC</td>
<td>continued controller</td>
</tr>
<tr>
<td>104</td>
<td>CONTROLLER.UP_SUCC</td>
<td>controller status is up</td>
</tr>
<tr>
<td>105</td>
<td>CONTROLLER.DOWN_SUCC</td>
<td>controller status is down</td>
</tr>
<tr>
<td>106</td>
<td>SUBMIT_JOB_SUCC</td>
<td>submitted job</td>
</tr>
<tr>
<td>107</td>
<td>DELETE_JOB_SUCC</td>
<td>deleted job</td>
</tr>
<tr>
<td>108</td>
<td>PURGE_DATABASE_SUCC</td>
<td>purged database</td>
</tr>
</tbody>
</table>
### Table 33–1  (Cont.) AMM Return Codes and Messages

<table>
<thead>
<tr>
<th>Return code number</th>
<th>Return code</th>
<th>Return message</th>
</tr>
</thead>
<tbody>
<tr>
<td>109</td>
<td>ENABLE_JOB_SUCCEED</td>
<td>enabled job</td>
</tr>
<tr>
<td>110</td>
<td>REPORT_SUCCEED</td>
<td>generated report</td>
</tr>
<tr>
<td>111</td>
<td>ENABLE_BATCH_SUCCEED</td>
<td>enabled batch</td>
</tr>
<tr>
<td>200</td>
<td>CONTROLLER_RUNNING_ERROR</td>
<td>ERROR: controller is already running</td>
</tr>
<tr>
<td>201</td>
<td>CONTROLLER_PAUSED_ERROR</td>
<td>ERROR: controller is already paused</td>
</tr>
<tr>
<td>202</td>
<td>CONTROLLER_SPEC_ACCESS_ERROR</td>
<td>ERROR: controller specification does not exist</td>
</tr>
<tr>
<td>203</td>
<td>CONTROLLER_COMM_ERROR</td>
<td>ERROR: controller cannot be reached</td>
</tr>
<tr>
<td>204</td>
<td>SEARCH_SPEC_IO_ERROR</td>
<td>ERROR: account search specification could not be accessed</td>
</tr>
<tr>
<td>205</td>
<td>OPERATION_ROLLBACK_ERROR</td>
<td>ERROR: operation rollback</td>
</tr>
<tr>
<td>206</td>
<td>SEARCH_SPEC_PARSE_ERROR</td>
<td>ERROR: account search specification cannot be parsed</td>
</tr>
<tr>
<td>207</td>
<td>OPERATION_PERM_ERROR</td>
<td>ERROR: operation not permitted for current user OR job_id/batch_id does not exist</td>
</tr>
<tr>
<td>208</td>
<td>CONTROLLER_UNKNOWN_HOST_ERROR</td>
<td>ERROR: controller host not found</td>
</tr>
<tr>
<td>209</td>
<td>CONTROLLER_PROCESS_ERROR</td>
<td>ERROR: controller process could not be created</td>
</tr>
<tr>
<td>210</td>
<td>REPORT_PROCESS_ERROR</td>
<td>ERROR: external process interruption</td>
</tr>
<tr>
<td>211</td>
<td>REPORT_SCRIPT_ACCESS_ERROR</td>
<td>ERROR: reporting tool not found or report type does not exist</td>
</tr>
<tr>
<td>212</td>
<td>OPT_PARAM_REQ_ERROR</td>
<td>ERROR: one optional parameter is required</td>
</tr>
<tr>
<td>213</td>
<td>CONFIG_FILE_ACCESS_ERROR</td>
<td>ERROR: configuration file cannot be accessed</td>
</tr>
<tr>
<td>214</td>
<td>INIT_ERROR</td>
<td>ERROR: could not create new object</td>
</tr>
<tr>
<td>215</td>
<td>EMPTY_RESULTSET_ERROR</td>
<td>ERROR: account search resulted in 0 accounts, job submission failed</td>
</tr>
<tr>
<td>216</td>
<td>CONVERSION_CLASS_LOAD_ERROR</td>
<td>ERROR: dynamic loading of custom Conversion class failed</td>
</tr>
</tbody>
</table>
Modifying the Account Migration Manager

This document describes how to create custom search criteria for the Oracle Communications Billing and Revenue Management (BRM) Account Migration Manager (AMM).

This document is for system administrators, database administrators, and programmers. Modifying AMM requires knowledge of the following:

- Java programming language
- PL/SQL
- BRM database schema

Creating Custom Account Search Criteria

AMM allows you to migrate accounts that meet custom criteria. For example, you can create custom criteria for finding and migrating accounts located in a certain state or belonging to a particular service provider.

To create a custom search criteria, perform these tasks:

1. Creating a Search Template
2. Adding New Entries to the Account Search Configuration File
3. Implementing and Compiling the Conversion Interface
4. Verifying Your Search Criteria

Creating a Search Template

AMM searches for accounts in a database by using SQL statements generated from an account search template. Before AMM can generate a SQL statement with new search criteria, you must first create a template for it in the custom account search properties file.

To create a template for your search criteria:

1. Open the custom account search properties file (BRM_Home/apps/amt/com/portal/amt/custom_account_search.properties) in a text editor.

2. Add SQL fragments for your search criteria by using the following syntax:

   `criteria_name=AND SQL_condition
   
   Where:`
Creating Custom Account Search Criteria

- **criteria_name** is the name of your selection criteria.
- **SQL_condition** is a valid SQL condition that searches a BRM table and references one or more search variables, as shown below. Search variables must be surrounded by curly braces "{}" and match an entry in the account_search.cfg file.

```
condition_text '{SearchVariable}'...
```

**Important**: **SearchVariable** must use a unique name and must not match one of the BRM-defined search variable names. For the list of BRM-defined search variables, see "Creating the Account Search Configuration File".

For information on the SQL condition, see your Oracle documentation. For a list of BRM tables, see "Storable Class Definitions" in BRM Developer’s Reference.

3. Save and exit the file.

**Sample Search Template**

The following sample search template enables AMM to search for accounts located in a particular state. It tells AMM to search the ACCOUNT_NAME_INFO_T table for objects with the state field set to a specified value.

```
# select accounts based on state
cust_acct_search_account_state_constraint=
AND EXISTS \
(SELECT an.obj_id0 FROM account_nameinfo_t an \\
WHERE an.obj_id0 = a.poid_id0 and an.state = '{account_state}')
```

**Adding New Entries to the Account Search Configuration File**

When building a query, AMM replaces the search variables in your account search template with values from the account search configuration file (BRM_Home/apps/amt/account_search.cfg).

To add an entry for your search variable:

1. Open the BRM_Home/apps/amt/account_search.cfg file in a text editor.
2. Add your new search entry and comments to the file.

**Important**: **SearchVariable** must match the search variable name referenced in the custom_account_search.properties file.

```
# - You should add comments about the new search entry and
#   valid values.

SearchVariable=
```

3. Save and exit the file.

**Sample Account Search Configuration File**

A sample search entry for the **account_state** search criteria:

```
# - Migrates accounts located in a specific state. Valid values
```

For information on the SQL condition, see your Oracle documentation. For a list of BRM tables, see "Storable Class Definitions" in BRM Developer’s Reference.
Implementing and Compiling the Conversion Interface

Each custom search variable must have a corresponding Java implementation of the Conversion interface.

1. Run the appropriate profile script for your shell. This script sets your CLASSPATH and PATH environment variables to the appropriate values. For example, for the c shell:

   ```
   cd BRM_Home/apps/amt
   source profile.csh
   ```

2. Create a class that implements the Conversion interface.

3. Save and compile your SearchVariable.java source file in the BRM_Home/apps/amt/com/portal/am directory.

   ```
   cd BRM_Home/apps/amt/com/portal/am
   javac SearchVariable.java
   ```

   This creates a SearchVariable.class file in the same directory.

---

**Important:** For AMM to successfully build a search with your custom search criteria:

- The class name must match the search variable name used in the custom_account_search.properties and account_search.cfg files.
- The class must reside in the BRM_Home/apps/amt/com/portal/am directory.

---

Sample Class Implementing Conversion Interface

The following sample class, account_state.class, allows users to search for accounts from California or Oregon.

```java
package com.portal.amt;
public class account_state implements Conversion {
    public String convert(String stateName) throws ConversionException {
        String stateCode = null;
        if(stateName.equals("California")) {
            stateCode = "CA";
        } else if(stateName.equals("Oregon")) {
            stateCode = "OR";
        } else {
            throw new
            ConversionException("Error: account_state " + stateName + " unknown.");
        }
    return(stateCode);
    }
}
```  

Verifying Your Search Criteria

Before migrating accounts with the new search criteria, verify its accuracy by:

1. **Verifying That the Search Criteria Creates Valid SQL Statements**
2. Verifying That the Search Criteria Finds Correct Accounts

**Verifying That the Search Criteria Creates Valid SQL Statements**

Use the `pin_amt_test` utility to verify that your custom search template generates a valid SQL statement. See `pin_amt_test`.

1. Open your account search configuration file (`BRM_Home/apps/amt/account_search.cfg`) in a text editor.

2. Enter values for the source and destination databases, the batch size, and your custom search criteria.

   For example, you might enter the following to test the `account_state` criteria:

   ```
   src_database=0.0.0.1
dest_database=0.0.0.2
start_creation_date=
end_creation_date=
product_name=
account_status=
bill_day_of_month=
max_accounts=
batch_size=50
poid_list=
account_state=California
   ```

3. Save and exit the file.

4. Use the `pin_amt_test` utility to generate a SQL statement with your new search criteria.

   ```bash
   % pin_amt_test -c AccountSearchFile
   ```

   If successful, the utility displays the resulting SQL statement. For example, the sample `account_state` criteria generates the following:

   ```
   Compile: account_search.cfg
   ------
   account search SELECT statement:
   -- acct_search_select: default
   SELECT
   DISTINCT a.poid_id0
   FROM account_t a
   WHERE
   ...
   ```

   If the compilation failed, the utility returns the file name and line number where the error occurred. For example, the utility returns the following when users enter an invalid state for the sample `account_state` criteria:

   ```
   compile: account_search.cfg
   ------
   account_search.cfg:32: mapping of account_state field value Florida failed
   ```

5. Verify that the resulting SQL statement is correct.

**Verifying That the Search Criteria Finds Correct Accounts**

Use the `pin_amt_test` utility to verify that your search criteria works properly. This utility only displays results on the screen and does not migrate your objects.

To verify your search query:
1. Create a database with a precisely defined set of account data. The database should contain a small number of accounts.

2. Create a list of account POIDs that meet the custom criteria you are testing. For example, write down the POIDs of all accounts created in California.

3. Open your account search configuration file (BRM_Home/apps/amt/account_search.cfg) in a text editor.

4. Enter values for the source and destination databases, the batch size, and your custom search criteria.

   For example, you might enter the following to test the account_state criteria:

   ```
   src_database=0.0.0.1
   dest_database=0.0.0.2
   account_state=California
   ```

5. Save and exit the file.

6. Use the pin_amt_test utility to execute your search query against the source database:

   ```
   % pin_amt_test -e AccountSearchFile
   ```

7. The utility prints to the screen a list of account POIDs that meet your search criteria. Compare this list of POIDs with the list you created in step 2.

   If the lists match, your new search criteria works properly and you can start using it to migrate accounts.

   If the lists do not match, make sure:

   - Your search template generates a valid SQL statement.
   - Your search template, search configuration file, and class all refer to the same variable name.
### AMT ER Diagram

#### Figure 35–1  AMT ER Diagram

<table>
<thead>
<tr>
<th>AMT_JOB_T</th>
<th>AMT_ACCOUNT_BATCH_T</th>
<th>AMT_ACCOUNT_BATCH_TABLE_T</th>
</tr>
</thead>
<tbody>
<tr>
<td>PK</td>
<td>JOB_ID</td>
<td>JOB_ID</td>
</tr>
<tr>
<td>USER_NAME</td>
<td>BATCH_HUM</td>
<td>BATCH_ID</td>
</tr>
<tr>
<td>SPEC_NAME</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRC_DATABASE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEST_DATABASE</td>
<td>ACCOUNT_CNT</td>
<td></td>
</tr>
<tr>
<td>START_CREATION_DATE</td>
<td>PROCESS_START_T</td>
<td>SRCH_TYPE</td>
</tr>
<tr>
<td>END_CREATION_DATE</td>
<td>PROCESS_END_T</td>
<td>TABLE_NAME</td>
</tr>
<tr>
<td>PRODUCT_NAME</td>
<td>PROCESSING_TIME</td>
<td>POID_TYPE</td>
</tr>
<tr>
<td>ACCOUNT_STATUS</td>
<td>STATUS</td>
<td></td>
</tr>
<tr>
<td>BILL_DAY_OF_MONTH</td>
<td>FAILURE</td>
<td></td>
</tr>
<tr>
<td>MAX_ACCOUNTS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BATCH_SIZE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JOB_CREATION_T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROCESS_START_T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROCESS_END_T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROCESSING_TIME</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STATUS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BATCH_CNT</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AMT_METADATA_T</th>
<th>AMT_ACCOUNT_DETAIL_T</th>
<th>AMT_BASE_DETAIL_T</th>
</tr>
</thead>
<tbody>
<tr>
<td>PK</td>
<td>H</td>
<td>H</td>
</tr>
<tr>
<td>DD_OBJID</td>
<td>POID_ID0</td>
<td>POID_ID0</td>
</tr>
<tr>
<td>FIELD_TYPE</td>
<td>POID_DB</td>
<td>POID_DB</td>
</tr>
<tr>
<td>SRC_TYPE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TABLE_NAME</td>
<td>ACCOUNT_HUM</td>
<td>POID_TYPE</td>
</tr>
<tr>
<td>INSERT_COLUMN_LIST</td>
<td>JOB_ID</td>
<td>JOB_ID</td>
</tr>
<tr>
<td>SELECT_COLUMN_LIST</td>
<td>BATCH_ID</td>
<td>BATCH_ID</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AMT_POD_TYPE_MAP_T</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>H</td>
</tr>
<tr>
<td>POID_TYPE</td>
<td>BASE_TABLE_NAME</td>
</tr>
<tr>
<td>TABLE_NAME</td>
<td></td>
</tr>
</tbody>
</table>
This chapter provides reference information for Oracle Communications Billing and Revenue Management (BRM) Account Migration Manager utilities.
**pin_amt**

Use this utility to migrate accounts from one Oracle Communications Billing and Revenue Management (BRM) database to another in a multidatabase system.

You define which accounts to migrate in the account search configuration file in the `BRM_Home/apps/amt` directory. See "Creating the Account Search Configuration File".

---

**Note:** To connect to the BRM databases, the `pin_amt` utility needs a configuration file in the `BRM_Home/sys/amt` directory. For information on how to create this file, see "Connecting AMM to Your Databases".

---

**Location**

`BRM_Home/bin`

**Syntax**

```
pin_amt [-f ConfigFileName] [-a ControllerID]
[-c start | stop | pause | continue | status | log]
[-s AccountSearchFile] [-d JobID]
[-r list_jobs | job_details | group_details]
[-p SourceDatabase] [-e JobID] [-b JobID:BatchNumber] [-h]
```

**Parameters**

- `-f ConfigFileName`
  Specifies the name of the configuration file that defines how to connect to each database in your system. By default, the `pin_amt` utility looks in the `BRM_Home/sys/amt` directory. If your configuration file is located in a different directory, you must specify the entire path for the file.

  If you use the `BRM_Home/sys/amt/Infranet.properties` file, you can ignore this parameter.

- `-a ControllerID`
  Specifies the AMM Controller to use.

  Use this option only with the `-c` option and only when your system contains multiple AMM Controllers. If your system contains only one AMM Controller, ignore this option.

- `-c start | stop | pause | continue | status | log`
  Sets the AMM Controller. When your system contains multiple AMM Controllers, you must also use the `-a` option.

  Use one of these options with the parameter:

  - `-c start` starts the AMM Controller.

  - `-c stop` stops the AMM Controller. If you stop the AMM Controller while it is processing a batch, it finishes processing the batch before stopping.

  - `-c pause` pauses the processing of a job in the queue. If you pause the Controller while it is processing a batch, it finishes processing the batch before pausing.

  - `-c continue` restarts processing a job that was paused.
-c status displays the current status of the AMM Controller.

-c log displays all AMM Controller transactions in real time through an Xterm window. To use this option, you must set the DISPLAY environment variable correctly.

-s AccountSearchFile
Specifies the name of the configuration file that defines which accounts to migrate. For information on how to create the file, look at the sample account search configuration file (BRM_Home/apps/amt/account_search.cfg).

By default, the pin_amt utility looks in the current working directory. If your configuration file is located in a different directory, you must specify the entire path for the file.

-d JobID
Deletes the specified job from the job management tables. When deleting a job that migrated successfully, this option also purges all migrated accounts from the source database.

-e JobID
Enables the specified job in the queue.

-r list_jobs | job_details | group_details
Runs the preconfigured report. Use one of these options with the parameter:

-r list_jobs displays the status of all jobs currently in the queue.

-r job_details displays the details of the specified job.

-r group_details displays the details of the specified account group.

-p SourceDatabase
Purges all accounts that were successfully migrated from the source database. For example, to purge invalid objects from your primary database, enter:

```
pin_amt -p 0.0.0.1
```

-h
Displays the syntax and parameters for this utility.

-b JobID:BatchNumber
Changes the status of the batch from FAILED to NOT PROCESSED, and the job from FINISHED to DISABLED. For information, see "About Batch Status Flags" and "About Batch Status Flags".

Results
The pin_amt utility notifies you when it successfully completes a command.

For error information about each job, run a report or look in the AMM Mover log file. The log file is in the directory specified by the 0.0.0.x_mover_log_file_dir entry in the Infranet.properties file.

For error information about the AMM Controller, look in the AMM Controller log file. The log file is in the directory specified by the controller_N_log_directory entry in the Infranet.properties file.

The history of all pin_amt commands is located in the pin_amt log file.
pin_amt_test

Use this utility to test your custom Oracle Communications Billing and Revenue Management (BRM) account search criteria. This utility safely executes your search criteria against a source database and displays either a SQL SELECT statement or a list of account POIDs meeting your search criteria.

You define which custom search criteria to test in the account search configuration file ($BRM_Home/apps/amt/account_search.cfg). See "Creating the Account Search Configuration File".

Note: To connect to the BRM databases, the pin_amt_test utility needs a configuration file in the $BRM_Home/sys/amt directory. For information on how to create this file, see "Connecting AMM to Your Databases".

Location
$BRM_Home/apps/amt

Syntax
pin_amt_test [-f ConfigFileName ]
    -c AccountSearchFile | -e AccountSearchFile | -h

Parameters

-1 ConfigFileName
Specifies the name of the configuration file that defines how to connect to each database in your system. By default, the pin_amt_test utility looks in the $BRM_Home/sys/amt directory. If your configuration file is located in a different directory, you must specify the entire path for the file.

If you use the $BRM_Home/sys/amt/Infranet.properties file, you can ignore this parameter.

-c AccountSearchFile
Displays the SQL SELECT statement generated with the account search criteria specified in AccountSearchFile.

By default, the pin_amt_test utility looks in the current working directory. If your account search file is located in a different directory, you must specify the entire path for the file.

-e AccountSearchFile
Executes the SQL SELECT statement for the specified search criteria against the source database and displays the list of account POIDs meeting the search criteria.

By default, the pin_amt_test utility looks in the current working directory. If your account search file is located in a different directory, you must specify the entire path for the file.

-h
Displays the syntax and parameters for this utility.
Results

The **pin_amt_test** utility prints to the screen the SQL SELECT statement, a list of accounts meeting your search criteria, or an Oracle error message. For information about Oracle error messages, see your Oracle documentation.
Part IX provides reference information about Oracle Communications Billing and Revenue Management (BRM) configuration files. It contains the following chapters:

- Business Logic pin.conf Reference
- System Administration pin.conf Reference
- business_params Reference
This document lists the business logic `pin.conf` settings used for configuring Oracle Communications Billing and Revenue Management (BRM).

For information about the syntax of `pin.conf` entries, see "Syntax for Configuration Entries".

For information about other configuration entries, see the following:
- `business_params Reference`
- `System Administration pin.conf Reference`

**Accounts Receivable pin.conf Entries**

Table 37–1 lists the Accounts Receivable `pin.conf` entries.
### Table 37–1 Accounts Receivable pin.conf Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
<th>pin.conf File</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>balance_coordinator</td>
<td>fm_bal</td>
<td>Specifies how BRM tracks the total resources reserved by a balance group. See &quot;Configuring How BRM Calculates Reservation Balances&quot; in BRM Telco Integration.</td>
<td>CM</td>
<td>This value is read by the utility when it runs. You do not need to restart the CM.</td>
</tr>
<tr>
<td>calc_cycle_from_cycle_start_t</td>
<td>fm_bill</td>
<td>Specifies whether to calculate product fees based on the product’s purchase date (PIN_FLD_CYCLE_START_T). See &quot;Calculating Product Cycle Fees for Backdating&quot; in BRM Configuring and Running Billing.</td>
<td>CM</td>
<td>This value is read by the utility when it runs. You do not need to restart the CM.</td>
</tr>
</tbody>
</table>
| cycle_tax_interval          | fm_bill | Determines whether deferred taxes are calculated separately for a parent and its nonpaying child accounts or are consolidated into a single tax item for both the parent and child accounts. See the following in BRM Calculating Taxes:  
  - Taxware: "Specifying How to Calculate Taxes"  
  - Vertex: "Specifying How to Calculate Taxes" | CM            | Cached by the CM. Restart the CM after changing this entry.                                                                                  |
| item_search_batch           | fm_bill | Specifies the number of items returned by a step search. See "Improving Item Search Performance" in BRM Managing Accounts Receivable.                                                                         | CM            | This value is read by the utility when it runs. You do not need to restart the CM.             |
| overdue_tolerance           | fm_bill | Specifies how BRM treats amounts applied to the item when they are less than the amount due as a result of euro and Economic and Monetary Union (EMU) conversions. See "Rounding Errors for Overpayments and Underpayments" in BRM Managing Customers. | CM            | This value is read by the utility when it runs. You do not need to restart the CM.             |
| underdue_tolerance          | fm_bill | Specifies how BRM treats amounts applied to the item when they are more than the amount due as a result of euro and EMU conversions. See "Rounding Errors for Overpayments and Underpayments" in BRM Managing Customers. | CM            | Cached by the CM. Restart the CM after changing this entry.                                    |
# Billing pin.conf Entries

Table 37–2 lists the Billing pin.conf entries.

<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
<th>pin.conf File</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>actg_dom</td>
<td>fm_cust_pol</td>
<td>Used during account creation to determine the day of the month to run billing.</td>
<td>CM Billing utilities</td>
<td>This value is read by the utility when it runs. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Setting the Default Accounting Day of Month (DOM)&quot; in BRM Configuring and Running Billing.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>actg_type</td>
<td>fm_cust_pol</td>
<td>Specifies the default billing type.</td>
<td>CM Billing utilities</td>
<td>This value is read by the utility when it runs. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Setting the Default Accounting Type&quot; in BRM Configuring and Running Billing.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>advance_bill_cycle</td>
<td>fm_bill</td>
<td>Sets the first billing date to be the day after account creation.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Setting the First Billing Cycle to the Day after Account Creation&quot; in BRM Configuring and Running Billing.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>attach_item_to_event</td>
<td>fm_act</td>
<td>Specifies how BRM assigns event and service combinations to bill items.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;About Using Event and Service Combinations to Assign Bill Items&quot; in BRM Configuring and Running Billing.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bill_when</td>
<td>fm_cust_pol</td>
<td>Specifies the default billing-cycle length.</td>
<td>CM</td>
<td>The new value becomes effective immediately. You do not have to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Setting the Default Billing-Cycle Length&quot; in BRM Configuring and Running Billing.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>billing_segment_config_refresh_delay</td>
<td>fm_cust</td>
<td>Specifies how often data in the cached /config/billing_segment object is automatically refreshed from the database.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Updating Billing Segments&quot; in BRM Configuring and Running Billing.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>config_billing_cycle</td>
<td>fm_bill</td>
<td>Specifies how long after the billing cycle ends that new events are considered for the previous month’s bill.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Customizing How to Bill Events That Occur between Billing Cycles&quot; in BRM Configuring and Running Billing.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 37–2 (Cont.) Billing pin.conf Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
<th>pin.conf File</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>config_billing_delay</td>
<td>fm_bill</td>
<td>Specifies the billing delay interval during which both old events (for the previous cycle) and new events (for the current cycle) can be processed. When specified, the system creates the next bill object (for the next billing cycle). Bill total calculation occurs only outside of this delay interval. See &quot;Setting Up Delayed Billing&quot; in BRM Configuring and Running Billing.</td>
<td>CM Billing utilities</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>custom_bill_no</td>
<td>fm_bill</td>
<td>Specifies the accounting cycle (first or last) in which to assign a bill number to a bill in a multi-month billing cycle. See &quot;Specifying When to Apply Custom Bill Numbers&quot; in BRM Configuring and Running Billing.</td>
<td>CM Billing utilities</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>cycle_delay_align</td>
<td>fm_bill</td>
<td>Specifies whether to align the product purchase, cycle, and usage start and end times to the accounting cycle. See &quot;Aligning Account and Cycle Start and End Times&quot; in BRM Configuring and Running Billing.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>cycle_delay_use_special_days</td>
<td>fm_bill</td>
<td>Sets the delayed cycle start date to the 1st of the following month for all deals purchased on the 29th, 30th, or 31st. See &quot;Setting Delayed Cycle Start Dates to the 29th, 30th, or 31st&quot; in BRM Configuring and Running Billing.</td>
<td>CM Billing utilities</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>deadlock_retry_count</td>
<td>pin_bill_accts</td>
<td>Specifies the number of retries to attempt when a deadlock occurs during a billing run. See “Specifying the Number of Retries in Case of a Deadlock”.</td>
<td>pin_billd</td>
<td>This entry is read by the utility when it runs. You do not need to restart the CM.</td>
</tr>
<tr>
<td>delay_cycle_fees</td>
<td>fm_bill</td>
<td>In systems set up for delayed billing, specifies when to apply cycle forward fees and cycle rollovers (during partial billing or final billing). See “Specifying When to Apply Cycle Forward Fees and Cycle Rollovers” in BRM Configuring and Running Billing.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>delta_step</td>
<td>pin_billd</td>
<td>Reduces contention at the database level during billing. See 'Rearranging Accounts to Improve Billing Performance'.</td>
<td>pin_billd</td>
<td>This entry is read by the utility when it runs. You do not need to restart the CM.</td>
</tr>
</tbody>
</table>
Table 37–2  (Cont.) Billing pin.conf Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
<th>pin.conf File</th>
<th>Implementation</th>
</tr>
</thead>
</table>
| enforce_billing             | pin_bill_accts | Enforces partial billing from the billing application inside the billing delay interval.  
  See "Enforcing Partial Billing in the Billing Delay Interval" in BRM Configuring and Running Billing. | pin_billd     | This entry is read by the utility when it runs. You do not need to restart the CM. |
| enforce_scoping             | cm        | Turns off branding conditions during billing.  
  See "Activating Branded Service Management" in BRM Managing Customers. |               | CM Cached by the CM. Restart the CM after changing this entry.         |
| keep_cancelled_products_or__discounts | fm_subscription_pol | Specifies whether to keep canceled products and discounts.  
  See 'Providing Discounts to Closed Accounts' in BRM Configuring and Running Billing. | CM            | The new value becomes effective immediately. You do not need to restart the CM. |
| minimum                     | fm_pymt_pol | Specifies the minimum payment to collect.  
  See "Specifying the Minimum Payment to Collect" in BRM Configuring and Running Billing. | CM            | The new value becomes effective immediately. You do not need to restart the CM. |
| num_billing_cycles          | fm_subs   | Specifies the maximum number of billing cycles allowed between the current time and the backdated event time of a backdated operation.  
  See "Setting the Thresholds That Trigger Automatic Rerating" in BRM Setting Up Pricing and Rating. | CM            | The new value becomes effective immediately. You do not need to restart the CM. |
| open_item_actg_include_prev_total | fm_bill | Includes previous bill totals in the pending receivable value calculated during billing for accounts that use open item accounting.  
  See 'Including Previous Balances in the Current Amount Due in Open Item Accounting' in BRM Configuring and Running Billing. | CM            | Cached by the CM. Restart the CM after changing this entry.         |
| purchase_fees_backcharge    | fm_bill   | Specifies which cycle (current or next) to apply the deferred purchase fees to during billing when the deferred time coincides with the billing time.  
  See "Specifying Which Billing Cycle to Assign to Deferred Purchase Fees" in BRM Configuring and Running Billing. | CM            | Cached by the CM. Restart the CM after changing this entry.         |
**Table 37–3** (Cont.) Billing pin.conf Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
<th>pin.conf File</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>rating_longcycle_roundup_flag</td>
<td>fm_rate</td>
<td>Enables rounding up for the long cycle. See &quot;Rounding Up Long Billing Cycles&quot; in BRM Configuring and Running Billing.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>stop_bill_closed_accounts</td>
<td>fm_bill</td>
<td>Stops billing of closed /billinfo objects when all items have zero due. See &quot;Suspending Billing of Closed Accounts&quot; in BRM Configuring and Running Billing.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>timestamp_rounding</td>
<td>fm_bill</td>
<td>Rounds timestamps of account products to midnight. See &quot;Configuring the Billing Cutoff Time&quot; and &quot;Calculating the Unit Interval&quot; in BRM Configuring and Running Billing. See also &quot;About Tracking Resources in Account Sub-Balances&quot; in BRM Setting Up Pricing and Rating.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>unset_error_status</td>
<td>pin_bill_accts</td>
<td>Sets billing status in the /billinfo object when billing errors occur. See &quot;Setting the Bill Unit Status When Billing Errors Occur&quot; in BRM Configuring and Running Billing.</td>
<td>pin_billld</td>
<td>This entry is read by the utility when it runs. You do not need to restart the CM.</td>
</tr>
<tr>
<td>use_number_of_days_in_month</td>
<td>fm_bill</td>
<td>Specifies how to calculate proration when a cycle product is purchased or canceled. See &quot;Calculating Prorated Cycle Fees&quot; in BRM Configuring and Running Billing.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
</tbody>
</table>

**Customer Management pin.conf Entries**

Table 37–3 lists the Customer Management pin.conf entries.
Table 37–3 Customer Management pin.conf Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
<th>pin.conf File</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>apply_folds</td>
<td>fm_bill</td>
<td>If you do not use folds, you can disable fold calculation by using this entry. This entry is on by default, that is, folds are applied, but you can turn it off to increase performance.</td>
<td>CM</td>
<td>Restart the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>apply_rollover</td>
<td>fm_bill</td>
<td>If you do not use rollovers, you can disable rollover calculation by using this entry. This entry is on by default, that is, rollovers are applied, but you can turn it off to increase performance.</td>
<td>CM</td>
<td>Restart the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>cycle_arrear_proration</td>
<td>fm_rate</td>
<td>Specifies when to prorate cycle arrears fees, at purchase or at cancellation. See “Proration for Special Cases” in BRM Configuring and Running Billing.</td>
<td>CM</td>
<td>Restart the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>em_db</td>
<td>fm_delivery</td>
<td>Specifies the Email Data Manager database number. See “Configuring the Email Data Manager” in BRM Managing Customers.</td>
<td>CM</td>
<td>Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>intro_dir</td>
<td>fm_cust_pol</td>
<td>Specifies the location of the introductory message that enables a customer to confirm account creation. See “Changing the Introductory Message Location” in BRM Managing Customers.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
</tbody>
</table>

Discounting pin.conf Entries

Table 37–4 lists the Discounting pin.conf entries.
### Table 37–4  Discounting pin.conf Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
<th>pin.conf File</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>cdc_line_cancel_day_include</td>
<td>fm_subscription</td>
<td>Specifies whether the day that the subscription was canceled is deducted from contract days.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See “Specifying Whether to Count the Days on Which Subscription Status Changes” in BRM Configuring Pipeline Rating and Discounting.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cdc_line_create_day_include</td>
<td>fm_subscription</td>
<td>Specifies whether to include or exclude the days on which the subscription service status changes.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See “Specifying Whether to Count the Days on Which Subscription Status Changes” in BRM Configuring Pipeline Rating and Discounting.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>non_currency_cdc</td>
<td>fm_subscription</td>
<td>Counts aggregated contract days for billing-time discount.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Configuring BRM to Track the Number of Contract Days” in BRM Configuring Pipeline Rating and Discounting.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>non_currency_mfuc</td>
<td>fm_subscription</td>
<td>Counts aggregated monthly fee and usage.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See ”Configuring BRM to Track Monthly Fees and Usage” in BRM Configuring Pipeline Rating and Discounting.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>propagate_discount</td>
<td>fm_subscription</td>
<td>Enables immediate propagation of shared discount when a discount is added to/deleted from the group or a member subscribes to/unsubscribes from the group.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Configuring the Start and End Times for Discount Sharing” in BRM Managing Accounts Receivable.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rollover_zeroout_discounts</td>
<td>fm_rate</td>
<td>Zeroes-out the positive bucket.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Configuring the Start and End Times for Discount Sharing” in BRM Managing Accounts Receivable.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>time_stamp_cdc</td>
<td>fm_subscription</td>
<td>Specifies whether the day of service reactivation is included in contract days.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See “Specifying Whether to Count the Days on Which Subscription Status Changes” in BRM Configuring Pipeline Rating and Discounting.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## General Ledger pin.conf Entries

Table 37–5 lists the General Ledger pin.conf entries.

### Table 37–5 General Ledger pin.conf Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
<th>pin.conf File</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>gl_segment</td>
<td>fm_cust_pol</td>
<td>Specifies the default G/L segment for an account during account creation.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Changing the Default G/L Segment&quot; in BRM Collecting General Ledger Data.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>transaction_grouping</td>
<td>pin_ledger_report</td>
<td>Specifies how many A/R accounts to group in a single transaction for ledger_report to run.</td>
<td>pin_bildd</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Setting the Number of A/R Accounts per G/L Report&quot; in BRM Collecting General Ledger Data.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## Invoicing pin.conf Entries

Table 37–6 lists the Invoicing pin.conf entries.

### Table 37–6 Invoicing pin.conf Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
<th>pin.conf File</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>database</td>
<td>pin_inv_accts</td>
<td>Specifies the database to which the following utilities should connect:</td>
<td>pin_inv</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td>pin_inv_export</td>
<td>■ pin_inv_accts</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>pin_inv_send</td>
<td>■ pin_inv_export</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>pin_inv_upgrade</td>
<td>■ pin_inv_send</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ pin_inv_upgrade</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Configuring pin.conf for Multiple Databases&quot; in BRM Designing and Generating Invoices.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>delivery_preference</td>
<td>pin_collections_send</td>
<td>Specifies the default delivery method, e-mail or print, for non-invoice accounts.</td>
<td>pin_collections</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td>dunning</td>
<td>See &quot;Setting the Delivery Option&quot; in BRM Collections Manager.</td>
<td>pin_inv</td>
<td></td>
</tr>
<tr>
<td></td>
<td>pin_inv_send</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>email_body</td>
<td>pin_collections_send</td>
<td>Specifies the path to a text file containing a customized message.</td>
<td>pin_collections</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td>dunning</td>
<td>See &quot;Specifying a File for the Email Body&quot; in BRM Collections Manager.</td>
<td>pin_inv</td>
<td></td>
</tr>
<tr>
<td></td>
<td>pin_inv_send</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Program</td>
<td>Description</td>
<td>pin.conf File</td>
<td>Implementation</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------------</td>
<td>-----------------------------------------------------</td>
</tr>
<tr>
<td>email_option</td>
<td>pin_coll sent</td>
<td>Specifies whether to attach the invoice or include it in the email body.</td>
<td>pin_coll</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td>send_dunning</td>
<td>See “Setting the Email Delivery Preference” in BRM Collections Manager.</td>
<td>pin_inv</td>
<td></td>
</tr>
<tr>
<td>event_cache</td>
<td>fm_inv</td>
<td>Enables the event cache for the PIN_FLDBAL_IMPACTS array.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See “Specifying Event Fields to Cache for Invoicing” in BRM Designing and Generating Invoices.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>export_dir</td>
<td>pin_inv_export</td>
<td>Specifies the path to the invoice directory.</td>
<td>pin_inv</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See “Exporting Invoices” in BRM Designing and Generating Invoices.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>from</td>
<td>pin_inv_send</td>
<td>Specifies the e-mail address of the sender.</td>
<td>pin_inv</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See “About Invoices” and “pin_inv_send” in BRM Designing and Generating Invoices.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>html_template</td>
<td>fm_inv_pol</td>
<td>Specifies the HTML template file to use to generate invoices.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See “Using HTML Invoice Templates” in BRM Designing and Generating Invoices.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>inv_item_fetch_size</td>
<td>fm_inv</td>
<td>Sets the number of items to fetch when a step search is used instead of a regular search.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See “Improving Performance When Generating Invoices” in BRM Designing and Generating Invoices.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>invoice_db</td>
<td>pin_inv_export</td>
<td>Specifies the database number.</td>
<td>pin_inv</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td>pin_inv_send</td>
<td>See “Configuring BRM to Use a Separate Invoice Database” in BRM Designing and Generating Invoices.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>invoice_dir</td>
<td>fm_bill</td>
<td>Specifies the directory where invoices are stored.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See “Exporting Invoices” in BRM Designing and Generating Invoices.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>invoice_fmt</td>
<td>pin_inv_export</td>
<td>Specifies the invoice format.</td>
<td>pin_inv</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td>pin_inv_send</td>
<td>See “Exporting Invoices” in BRM Designing and Generating Invoices.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Payments pin.conf Entries

Table 37–7 lists the Payments pin.conf entries.

<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
<th>pin.conf File</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>inv_perf_features</td>
<td>fm_inv</td>
<td>Improves performance by removing unnecessary details from your invoices.</td>
<td></td>
<td>CM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Improving Performance by Removing Invoice Details You Do Not Need&quot; in</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>BRM Designing and Generating Invoices.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sender</td>
<td>fm_cust_pol</td>
<td>Specifies the name of the e-mail sender.</td>
<td>pin_inv</td>
<td>CM</td>
</tr>
<tr>
<td></td>
<td>pin_inv_send</td>
<td>See &quot;Changing the Welcome Message Sender Address&quot; in BRM Managing Customers.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>service_centric_invoice</td>
<td>fm_inv_pol</td>
<td>Enables service-centric invoicing.</td>
<td></td>
<td>CM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Creating Service-Centric Invoices&quot; in BRM Designing and Generating Invoices.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>show rerate_details</td>
<td>fm_inv_pol</td>
<td>Specifies whether to display shadow adjustment details on invoices.</td>
<td></td>
<td>CM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Including Shadow Event Adjustment Details in Invoices&quot; in BRM Designing and Generating Invoices.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>subject</td>
<td>pin_inv_send</td>
<td>Specifies the subject of the e-mail.</td>
<td>pin_inv</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;About Invoices&quot; and &quot;pin_inv_send&quot; in BRM Designing and Generating Invoices.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Program</td>
<td>Description</td>
<td>pin.conf File</td>
<td>Implementation</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>---------------</td>
<td>---------------------------------------------------------------------</td>
</tr>
<tr>
<td>cc_collect</td>
<td>fm_pynt_pol</td>
<td>Specifies whether to perform real-time authorization of the fixed charges information with the customer’s credit card during registration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Charging Customers at Registration&quot; in BRM Managing Customers.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td>cc_revalidation_interval</td>
<td>fm_pynt_pol</td>
<td>Specifies the credit card revalidation interval.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td>cc_validate</td>
<td>fm_pynt_pol</td>
<td>Specifies whether to validate a customer’s credit card information during registration.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td>cid_required</td>
<td>fm_pynt_pol</td>
<td>Specifies whether to use American Express CID (Card identifier) fraud protection for Paymentech transactions.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td>config_payment</td>
<td>fm_pynt</td>
<td>Specifies the database and POID of the /config/payment object. Change this value if you create a custom /config/payment object.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td>cvv2_required</td>
<td>fm_pynt_pol</td>
<td>Specifies whether to use Visa CVV2 fraud protection for Paymentech transactions.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td>dd_collect</td>
<td>fm_pynt_pol</td>
<td>Specifies whether to perform real-time authorization of the fixed charges information with the customer’s debit card during registration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Enabling Direct Debit Processing&quot; in BRM Configuring and Collecting Payments.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td>Name</td>
<td>Program</td>
<td>Description</td>
<td>pin.conf File</td>
<td>Implementation</td>
</tr>
<tr>
<td>----------------------</td>
<td>------------</td>
<td>------------------------------------------------------------------------------</td>
<td>---------------</td>
<td>-----------------------------------------------------</td>
</tr>
<tr>
<td>dd_revalidation_</td>
<td>fm_pymt_pol</td>
<td>Specifies the debit card revalidation interval.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td>interval</td>
<td></td>
<td>See “Enabling Direct Debit Processing” in BRM Configuring and Collecting Payments.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>dd_validate</td>
<td>fm_pymt_pol</td>
<td>Specifies whether to validate a customer’s direct debit (debit card) information during registration.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See “Enabling Direct Debit Processing” in BRM Configuring and Collecting Payments.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>merchant</td>
<td>fm_cust_pol</td>
<td>Specifies the merchant to receive money collected during credit-card processing.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See “Specifying Merchant IDs and Merchant Numbers” in BRM Configuring and Collecting Payments.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>minimum</td>
<td>fm_pymt_pol</td>
<td>Specifies the minimum balance for retrieving accounts for collection.</td>
<td>CM pin_bildd</td>
<td>The new value becomes effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Use this entry to set a minimum threshold for the amount due on an account when searching for accounts for collection.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>See “Specifying the Minimum Payment to Collect” in BRM Configuring and Running Billing.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>minimum_refund</td>
<td>fm_pymt_pol</td>
<td>Specifies the minimum refund amount that the system allows.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See “Specifying the Minimum Amount to Refund” in BRM Configuring and Running Billing.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>payment_batch_</td>
<td>fm_pynt</td>
<td>Specifies whether Payment Tool locks accounts at the account level or the batch level when processing payments.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>lock</td>
<td></td>
<td>See “Configuring Payment Tool to Lock at the Account Level during Batch Processing” in BRM Configuring and Collecting Payments.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>validate_acct</td>
<td>fm_pymt_pol</td>
<td>Allows use of the customer’s credit card on an account different from the root account.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See “Specifying the Account That Records Credit Card Validations” in BRM Managing Customers.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>


## Pricing and Rating pin.conf Entries

Table 37–8 lists the Pricing and Rating pin.conf entries.

<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
<th>pin.conf File</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>backdate_trigger_auto_rerate</td>
<td>fm_subs</td>
<td>Specifies whether to create auto-rerate job objects used by pin_rerate.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See “Configuring Automatic Rerating of Backdated Events” in BRM Setting Up Pricing and Rating.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>backdate_window</td>
<td>fm_subs</td>
<td>Specifies the minimum time difference needed between the current time and the backdated event time for triggering automatic rerating.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See “Configuring Automatic Rerating of Backdated Events” in BRM Setting Up Pricing and Rating.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cache_references_at_start</td>
<td>fm_price</td>
<td>Specifies whether to store objects referenced by price objects in memory in the CM cache when the CM starts.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Improving Performance for Loading Large Price Lists&quot;.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cancel_tolerance</td>
<td>fm_bill</td>
<td>Specifies the cancellation tolerance of account products, in minutes.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Canceling Products without Charging a Cancel Fee&quot; in BRM Managing Customers.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>delay</td>
<td>pin_rerate</td>
<td>Specifies the delay for rerating jobs.</td>
<td>pin_rerate</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See “Specifying the Event Sequence for Rerating” in BRM Setting Up Pricing and Rating.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>extra_rate_flags</td>
<td>fm_rate</td>
<td>Turns optional rating features on and off.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Setting Optional Rating Flags&quot; in BRM Setting Up Pricing and Rating.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>fm_price_cache_beid</td>
<td>cm_cache</td>
<td>Specifies attributes of the CM cache for the size of balance element IDs (beid).</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Improving Performance for Loading Large Price Lists&quot;.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>fm_price_prod_provisioning_cache</td>
<td>cm_cache</td>
<td>Specifies the attributes of the CM cache for the size of product provisioning tags.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Improving Performance for Loading Large Price Lists&quot;.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Program</td>
<td>Description</td>
<td>pin.conf File</td>
<td>Implementation</td>
</tr>
<tr>
<td>--------------------------</td>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
<td>---------------</td>
<td>--------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>log_price_change_event</td>
<td>fm_price</td>
<td>Specifies whether to create and log events for price object changes.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Improving Performance for Loading Large Price Lists&quot;.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>log_refresh_product</td>
<td>fm_rate</td>
<td>Specifies whether to log price list changes. You can use the log entries to generate notifications about such changes.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Logging Changes to Price Lists&quot; in BRM Setting Up Pricing and Rating.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>non_currency_linecount</td>
<td>fm_sub</td>
<td>Counts aggregated number of lines.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td>subscription</td>
<td>See &quot;Configuring BRM to Track the Number of Subscriptions&quot; in BRM Configuring Pipeline Rating and Discounting.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rate_change</td>
<td>fm_sub</td>
<td>Enables the enhanced rate change management feature.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td>subscription</td>
<td>See &quot;Configuring BRM to Apply Multiple Rates in a Cycle&quot; in BRM Configuring and Running Billing.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rating_max_scale</td>
<td>fm_rate</td>
<td>Specifies the precision level of decimal values.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Changing the Precision of Rounded and Calculated Values&quot;.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rating_quantity_rounding_scale</td>
<td>fm_rate</td>
<td>Specifies the precision of rounded values.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Changing the Precision of Rounded and Calculated Values&quot;.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rating_timezone</td>
<td>fm_rate</td>
<td>Specifies the server time zone.</td>
<td>CM</td>
<td>Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Specifying a Time Zone&quot; in BRM Setting Up Pricing and Rating.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>refresh_product_interval</td>
<td>fm_rate</td>
<td>Specifies the interval in which price lists are refreshed in cache memory.</td>
<td>CM</td>
<td>Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Setting the Interval for Checking for Price List Changes&quot;.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 37–8 (Cont.) Pricing and Rating pin.conf Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
<th>pin.conf File</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>timezone_file</td>
<td>fm_rate</td>
<td>Specifies the location of the timezone.txt file.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>update_interval</td>
<td>fm_zonemap_pol</td>
<td>Specifies an interval for revising the zone map.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>validate_deal_dependencies</td>
<td>fm_utils</td>
<td>Specifies whether to validate the deal prerequisites and mutually exclusive relations.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
</tbody>
</table>

### Registration pin.conf Entries

Table 37–9 lists the Registration pin.conf entries.

### Table 37–9 Registration pin.conf Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
<th>pin.conf File</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>allow_active_service_with_inactive_account</td>
<td>fm_cust</td>
<td>Specifies whether services can be activated (during a SET_STATUS operation) if the account or /billinfo is inactive.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>cc_checksum</td>
<td>fm_cust_pol</td>
<td>Specifies whether to run checksum validation on the customer's credit card during account creation.</td>
<td>CM</td>
<td>The new value becomes effective immediately and applies to the next account created. The CM does not need to be restarted.</td>
</tr>
<tr>
<td>config_dir</td>
<td>fm_cust_pol</td>
<td>Specifies the location of the ISP configuration data, which is stored in the default.config file. Used by the PCM_OP_CUST_POL_GET_CONFIG policy opcode.</td>
<td>CM</td>
<td>The new value becomes effective immediately and applies to the next account created. The CM does not need to be restarted.</td>
</tr>
<tr>
<td>country</td>
<td>fm_cust_pol</td>
<td>Specifies the default country for new accounts (default is USA).</td>
<td>CM</td>
<td>The new value becomes effective immediately and applies to the next account created. The CM does not need to be restarted.</td>
</tr>
</tbody>
</table>

37-16   BRM System Administrator's Guide
Table 37–9  (Cont.)  Registration pin.conf Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
<th>pin.conf File</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>currency</td>
<td>fm_cust_pol</td>
<td>Specifies the default currency for new accounts.</td>
<td></td>
<td>CM The new value becomes effective immediately and applies to the next account created. The CM does not need to be restarted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Setting the Default Account Currency&quot; in BRM Managing Customers.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>domain</td>
<td>fm_cust_pol</td>
<td>Specifies the email domain assigned to customers during account creation.</td>
<td></td>
<td>CM The new value becomes effective immediately and applies to the next account created. The CM does not need to be restarted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Defining Customer Email Domain Names&quot; in BRM Managing Customers.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>new_account_welcome_msg</td>
<td>fm_cust_pol</td>
<td>Specifies whether the system should send the default welcome message on account creation.</td>
<td></td>
<td>CM The new value becomes effective immediately and applies to the next account created. The CM does not need to be restarted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Enabling the Welcome Message&quot; in BRM Managing Customers.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>welcome_dir</td>
<td>fm_cust_pol</td>
<td>Specifies the location of the welcome message sent to customers after account creation.</td>
<td></td>
<td>CM The new value becomes effective immediately and applies to the next account created. The CM does not need to be restarted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Specifying the Welcome Message Location&quot; in BRM Managing Customers.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Revenue Assurance pin.conf Entries

Table 37–10 lists the Revenue Assurance pin.conf entries.

Table 37–10  Revenue Assurance pin.conf Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
<th>pin.conf File</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>writeoff_control_point_id</td>
<td>fm_process_audit</td>
<td>Changes the control point ID.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Changing the Control Point for Data on Written-Off EDRs&quot; in BRM Collecting Revenue Assurance Data.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Services Framework pin.conf Entries

Table 37–11 lists the Services Framework pin.conf entries.
<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
<th>pin.conf File</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>agent_return</td>
<td>fm_tcf</td>
<td>Specifies the provisioning status when provisioning is simulated.</td>
<td>CM</td>
<td>Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Using the Network Simulator&quot; in BRM Telco Integration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>commit_at_prep</td>
<td>dm_provision</td>
<td>Specifies whether to send the payload to the agent at PREP_COMMIT time and not to send anything at COMMIT time.</td>
<td>dm_prov_telco</td>
<td>This entry is read by the utility when it runs. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Configuring the Provisioning Data Manager&quot; in BRM Telco Integration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>connect_retries</td>
<td>dm_provision</td>
<td>Specifies the number of times dm_provision attempts to connect to the Infranet agent on connection failure.</td>
<td>dm_prov_telco</td>
<td>This entry is read by the utility when it runs. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Provisioning Data Manager Configuration File Entries&quot; in BRM Telco Integration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>connect_retry_interval</td>
<td>dm_provision</td>
<td>Specifies how many seconds to wait before retrying to connect to the Infranet agent on connection failure.</td>
<td>dm_prov_telco</td>
<td>This entry is read by the utility when it runs. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Provisioning Data Manager Configuration File Entries&quot; in BRM Telco Integration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>prov_db</td>
<td>fm_tcf</td>
<td>Specifies the number of the database to which provisioning connects to send the service order.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Provisioning Data Manager Configuration File Entries&quot; in BRM Telco Integration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>prov_ptr</td>
<td>dm_provision</td>
<td>Specifies where to find the Provisioning Data Manager.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Configuring the Provisioning Data Manager&quot; in BRM Telco Integration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>prov_timeout</td>
<td>dm_provision</td>
<td>Specifies the length of time to wait to receive a complete response from the provisioning agent.</td>
<td>dm_prov_telco</td>
<td>This entry is read by the utility when it runs. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See the instructions in the pin.conf file.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Table 37–11 (Cont.) Service Framework pin.conf Entries**

<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
<th>pin.conf File</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>provisioning_enabled</td>
<td>fm_tcf</td>
<td>Enables product-level provisioning. See &quot;Enabling Wireless Provisioning&quot; in BRM Telco Integration.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>simulate_agent</td>
<td>fm_tcf</td>
<td>Creates a response and updates the service order. See &quot;Using the Network Simulator&quot; in BRM Telco</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td>wait_for_all_interim_stop_request</td>
<td>fm_tcf_aaa</td>
<td>Checks for all interim stop accounting requests before processing the final stop accounting request.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
</tbody>
</table>

**Tax Calculation pin.conf Entries**

Table 37–12 lists the Tax Calculation pin.conf entries.

**Table 37–12 Tax Calculation pin.conf Entries**

<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
<th>pin.conf File</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>commtax_config_name</td>
<td>dm_vertex</td>
<td>Specifies the Communications Tax Q Series configuration name. See &quot;Configuring the Vertex DM for Communications Tax Q Series&quot; in BRM Calculating Taxes.</td>
<td>dm_vertex</td>
<td>Valid only when commtax_series is set to 6. Restart the DM after changing this entry.</td>
</tr>
<tr>
<td>commtax_config_path</td>
<td>dm_vertex</td>
<td>Specifies the location of Communications Tax Q Series configuration file (ctqcfg.xml). See &quot;Configuring the Vertex DM for Communications Tax Q Series&quot; in BRM Calculating Taxes.</td>
<td>dm_vertex</td>
<td>Valid only when commtax_series is set to 6. Restart the DM after changing this entry.</td>
</tr>
<tr>
<td>commtax_series</td>
<td>dm_vertex</td>
<td>Specifies the Vertex tax package that BRM uses to calculate telecom taxes. See &quot;Configuring the Vertex DM for Communications Tax Q Series&quot; in BRM Calculating Taxes.</td>
<td>dm_vertex</td>
<td>Restart the DM after changing this entry.</td>
</tr>
</tbody>
</table>
### Table 37–12 (Cont.) Tax Calculation pin.conf Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
<th>pin.conf File</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>include_zero_tax</td>
<td>fm_rate</td>
<td>Specifies whether to include zero tax amounts in the tax jurisdictions for the event’s balance impacts. See &quot;Reporting Zero Tax Amounts&quot; in <em>BRM Calculating Taxes</em>.</td>
<td>CM</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
</tbody>
</table>
| provider_loc          | fm_rate_pol | Specifies the city, state, ZIP code, and country where services are provided for taxation. See the following in *BRM Calculating Taxes*:  
- Taxware: "Defining a Default Ship-From Locale"  
- Vertex: "Defining a Default Ship-From Locale" | CM            | The new value becomes effective immediately and all subsequent tax calculations use the new address. You do not need to restart the CM. |
<p>| quantum_logfile       | dm_vertex | Specifies the QSUT API to log debug information (into the specified file name) about the current transaction to be processed for tax calculation. See “Setting Up Tax Calculation for Vertex” in <em>BRM Calculating Taxes</em>. | dm_vertex     | The new value becomes effective immediately. You do not need to restart the CM. |
| quantumdb_passwd      | dm_vertex | Specifies the Oracle user password. See “Specifying the Database Server Name, User ID, and Password for Sales Tax Q Series” in <em>BRM Calculating Taxes</em>. | dm_vertex     | Restart the Vertex DM after changing this entry.                               |
| quantumdb_register    | dm_vertex | Specifies whether to log an audit trail of invoices in the Quantum Register database. See “Setting Up Tax Calculation for Vertex” in <em>BRM Calculating Taxes</em>. | dm_vertex     | The new value becomes effective immediately. You do not need to restart the CM. |
| quantumdb_server      | dm_vertex | Specifies the name of the database server than contains the Quantum tables. See “Specifying the Database Server Name, User ID, and Password for Sales Tax Q Series” in <em>BRM Calculating Taxes</em>. | dm_vertex     | Restart the Vertex DM after changing this entry.                               |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
<th>pin.conf File</th>
<th>Implementation</th>
</tr>
</thead>
</table>
| quantumdb_source      | dm_vertex | Specifies the schema where the STQ tables reside. For Indexed Sequential Access Method (ISAM) databases, specifies the ISAM data file directory.  
See "Specifying the Database Server Name, User ID, and Password for Sales Tax Q Series” in BRM Calculating Taxes. | dm_vertex     | Restart the Vertex DM after changing this entry.         |
| quantumdb_user        | dm_vertex | Specifies the name of the Oracle user.  
See "Specifying the Database Server Name, User ID, and Password for Sales Tax Q Series” in BRM Calculating Taxes. | dm_vertex     | Restart the Vertex DM after changing this entry.         |
| tax_now               | fm_ar    | Specifies when tax is calculated for account-level adjustments (at the end of billing or at the time of adjustment).  
See "Configuring the Default Tax Method for Account-Level Adjustments” in BRM Managing Accounts Receivable. | CM            | The new value is effective immediately. You do not need to restart the CM. |
| tax_return_juris      | fm_rate  | Specifies whether to summarize taxes by jurisdiction or to itemize taxes.  
See "Itemizing Taxes by Jurisdiction” in BRM Calculating Taxes. | CM            | Restart the CM after changing this entry.                |
| tax_return_loglevel   | fm_rate  | Specifies how to log messages returned from the taxation DM.  
See "Modifying Tax Data after Calculating Taxes” in BRM Calculating Taxes. | CM            | Restart the CM after changing this entry.                |
| tax_reversal_with_tax | fm_ar    | Specifies whether to apply a tax reversal for an adjustment, dispute, or settlement  
See "Configuring Taxes for Adjustments, Disputes, and Settlements” in BRM Calculating Taxes. | CM            | The new value is effective immediately. You do not need to restart the CM. |
| tax_supplier_map      | fm_rate  | Specifies the location of the tax_supplier_map file.  
See the following in BRM Calculating Taxes:  
• Taxware: “Specifying the Location of the tax_supplier_map File”  
• Vertex: “Specifying the Location of the tax_supplier_map File” | CM Billing utilities | Restart the CM after changing this entry. |

Table 37–12 (Cont.) Tax Calculation pin.conf Entries
### Table 37–12 (Cont.) Tax Calculation pin.conf Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
<th>pin.conf File</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>tax_valid</td>
<td>fm_cust_pol</td>
<td>Specifies how to validate the state and zip code of the billing address during account registration.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>See the following in BRM Calculating Taxes:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Taxware: “Specifying Whether to Validate ZIP Codes”</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Vertex: “Specifying Whether to Validate ZIP Codes”</td>
<td></td>
<td></td>
</tr>
<tr>
<td>taxation_switch</td>
<td>fm_bill</td>
<td>Enables taxation.</td>
<td>CM</td>
<td>Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Enabling Taxation During Real-Time Rating or Billing&quot; in BRM Calculating Taxes.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>taxcodes_map</td>
<td>fm_rate</td>
<td>Specifies the location of the taxcodes_map file.</td>
<td>CM</td>
<td>Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See the following in BRM Calculating Taxes:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &quot;Calculating Flat Taxes by Using the taxcodes_map File&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Taxware: &quot;Specifying the Location of the taxcodes_map File&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Vertex: &quot;Specifying the Location of the taxcodes_map File&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>taxware_db</td>
<td>fm_rate</td>
<td>Specifies the database number of the Taxware database.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See “Specifying Connection Entries” in BRM Calculating Taxes.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>vertex_db</td>
<td>fm_rate</td>
<td>Specifies the database number of the Vertex database.</td>
<td>CM</td>
<td>The new value is effective immediately. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See “Specifying Connection Entries” in BRM Calculating Taxes.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This chapter lists the pin.conf settings used by system administrators in Oracle Communications Billing and Revenue Management (BRM).

See also "business_params Reference" and "Business Logic pin.conf Reference".

Connection Manager (CM) pin.conf Entries

Table 38–1 lists the CM pin.conf entries.

Table 38–1  CM pin.conf Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cm_cache_space</td>
<td>cm</td>
<td>(Optional) Reserves the cache memory for all the Facilities Modules. The default is 6291456. The value is always in multiples of 1024.</td>
</tr>
<tr>
<td>cm_data_dictionary_cache</td>
<td>cm_cache</td>
<td>Increases the size of the CM cache for the data dictionary. See &quot;Increasing the Size of the CM Cache for the Data Dictionary&quot; in BRM Developer’s Guide.</td>
</tr>
<tr>
<td>cm_data_file</td>
<td>cm</td>
<td>Specifies the name and location of the shared memory file that caches global information for the CM. See the instructions in the CM pin.conf file.</td>
</tr>
<tr>
<td>cm_logfile</td>
<td>cm</td>
<td>Specifies the full path to the log file used by the CM. See &quot;Changing the Name or Location of a Log File&quot;.</td>
</tr>
<tr>
<td>cm_logformat</td>
<td>cm</td>
<td>Specifies which PINLOG format to use. See &quot;Setting the CM Log Time Resolution&quot;.</td>
</tr>
<tr>
<td>cm_max_connects</td>
<td>cm</td>
<td>Specifies the maximum number of client connections to the CM. See &quot;Specifying the Number of Connections to CMs&quot;.</td>
</tr>
<tr>
<td>cm_name</td>
<td>cm</td>
<td>Specifies the name of the computer where the CM runs. See &quot;About Connecting BRM Components&quot;.</td>
</tr>
<tr>
<td>cm_opcode_stats</td>
<td>cm</td>
<td>Specifies whether to collect statistics about CM opcode performance. See &quot;Getting Quality of Service Statistics from the CM&quot;.</td>
</tr>
<tr>
<td>cm_ports</td>
<td>cm</td>
<td>Specifies the port number of the computer where the CM runs. See &quot;About Connecting BRM Components&quot;.</td>
</tr>
<tr>
<td>Name</td>
<td>Program</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>--------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| cm_timeout            | cm           | Specifies the time-out value for receiving the next opcode request from an application.  
                             See "Setting the CM Time Interval between Opcode Requests".                            |
| creation_logging      | fm_cust      | Specifies whether to log non-dollar events.  
                             See "Logging Non-Currency Events" and "Improving Connection Manager Performance". |
| dm_attributes         | cm           | Specifies attributes associated with a particular database.  
                             See "Activating Branded Service Management" in BRM Managing Customers and "Setting Data Manager Attributes". |
| dm_pointer            | cm           | Specifies where to find one or more DMs for the BRM database.  
                             See "Setting Data Manager Attributes".                                                  |
| dm_port               | dm           | Specifies the port number of the computer where the CM runs.  
                             See "About Connecting BRM Components".                                                  |
| em_group              | cm           | Configures the CM to send requests to the NET_EM module.  
                             See "Configuring the CM to Send Real-Time Requests to the NET_EM Module".           |
| em_pointer            | cm           | Specifies where to find external managers (EMs) that provide other opcodes to the CM.  
                             See "Configuring the CM to Send Real-Time Requests to the NET_EM Module".          |
| enable_pcm_op_call_stack | dm         | Specifies whether PCM.OP_CALL_STACK is printed in the cm.pinlog file after each opcode from a nap (Network Application Program) is completed.  
                             See "Recording Opcode Calls in the CM Log File" and "Getting Opcode Statistics from IMDB Cache DM". |
| execute_all_actions   | fm_collections | Specifies whether to execute all actions, including actions due before the date on which pin_collections_process is run for the first time, or only to execute actions due after the date on which pin_collections_process is run for the first time. |
| fetch_size            | pin_billd    | Specifies the number of objects received from the database in a block and cached in system memory for processing.  
                             See "Tuning the Account Cache Size for Billing Utilities (fetch_size)" and "Tuning the Account Cache Size for Invoice Utilities (fetch_size)". |
|                        | pin_mta      |                                                                                                                                              |
| fm_utils_content_srvc_profile_cache | cm_cache | Specifies attributes of the content service profile cache.  
                             See "Setting up Content Manager" in BRM Content Manager.                            |
| group_children_fetch  | fm_bill      | Used to perform a step search for child accounts in groups when the parent group contains many members. See "Filtering Search Results".   |
| group_members_fetch   | fm_bill      | Used to perform a step search for members of the group sharing object when the parent group contains many members. See "Filtering Search Results". }
### Table 38–1 (Cont.) CM pin.conf Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ip</td>
<td></td>
<td>Specifies the IP address of the devices managed by IP Address Manager. Check “Using the IP Address Manager APIs” in <em>BRM Telco Integration</em>.</td>
</tr>
<tr>
<td>item_fetch_size</td>
<td>fm_bill</td>
<td>Used to perform a step search for items. Refer to &quot;Filtering Search Results&quot;.</td>
</tr>
<tr>
<td>login_audit</td>
<td>cm</td>
<td>Creates the session event’s storable object for each client application. See “Turning Off Session-Event Logging”.</td>
</tr>
<tr>
<td>max_pcm_op_call_stack_</td>
<td>dm</td>
<td>Specifies the number of entries allocated for PCM_OP_CALL_STACK. See “Recording Opcode Calls in the CM Log File” and “Getting Opcode Statistics from IMDB Cache DM”.</td>
</tr>
<tr>
<td>entries</td>
<td></td>
<td></td>
</tr>
<tr>
<td>monitor</td>
<td>pin_mta</td>
<td>Specifies the path and name of a shared memory map file used by the <em>pin_mta_monitor</em> utility. See “Creating Configuration Files for BRM Utilities”.</td>
</tr>
<tr>
<td>passwd_age</td>
<td>cm</td>
<td>Specifies the number of days after which the password expires. See “Setting the Default Password Expiry Duration”.</td>
</tr>
<tr>
<td>primary_database</td>
<td></td>
<td>Specifies the primary database. See the instructions in the pin.conf file.</td>
</tr>
<tr>
<td>primary_db</td>
<td>cm</td>
<td>Specifies the primary database. See ”Running pin_multidb.pl -i“ in <em>BRM Installation Guide</em>.</td>
</tr>
<tr>
<td>sample_handler_logfile</td>
<td></td>
<td>Specifies the full path to the log file for the Batch Controller. Check &quot;Configuring the Batch Controller&quot; and &quot;Customizing SampleHandler&quot;.</td>
</tr>
<tr>
<td>support_multiple_so</td>
<td>fm_tcf</td>
<td>Specifies whether Services Framework provisioning can make in-flight changes to service orders. See &quot;Enabling In-Flight Changes to Service Orders” in <em>BRM Telco Integration</em>.</td>
</tr>
</tbody>
</table>

**DataManager (DM) pin.conf Entries**

Table 38–2 lists the DM pin.conf entries.
<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>config_dir</td>
<td>fm_cust_pol</td>
<td>Specifies the location of the ISP configuration data. Used by the PCM_OP_CUST_POL_GET_CONFIG policy opcode. See &quot;Sending Account Information to Your Application When an Account Is Created&quot; in BRM Managing Customers.</td>
</tr>
<tr>
<td>crypt</td>
<td>NA</td>
<td>Associates a four-byte tag with an encryption algorithm and secret key combination. See &quot;Configuring the Data Manager (DM) for AES Encryption&quot; in BRM Developer’s Guide.</td>
</tr>
<tr>
<td>dd_write_enable_fields</td>
<td>dm</td>
<td>Specifies whether this DM can create fields in the data dictionary. See &quot;Modifying the pin.conf file to Enable Changes&quot; in BRM Developer’s Guide.</td>
</tr>
<tr>
<td>dd_write_enable_objects</td>
<td>dm</td>
<td>Specifies whether this DM can edit, create, and delete custom storable classes in the data dictionary. See &quot;Modifying the pin.conf file to Enable Changes&quot; in BRM Developer’s Guide.</td>
</tr>
<tr>
<td>dd_write_enable_portal_objects</td>
<td>dm</td>
<td>Specifies whether this DM can delete predefined BRM storable classes and add and delete fields in one of those classes. See &quot;Modifying the pin.conf file to Enable Changes&quot; in BRM Developer’s Guide.</td>
</tr>
<tr>
<td>dd_write_enable_types</td>
<td>dm</td>
<td>Specifies whether this DM can edit, create, and delete custom object type names in the data dictionary.</td>
</tr>
<tr>
<td>dm_bigsizer</td>
<td>dm</td>
<td>Specifies the size of the DM shared memory. See &quot;Setting DM Shared Memory Size&quot;.</td>
</tr>
<tr>
<td>dm_debug</td>
<td>dm</td>
<td>Specifies the debugging information to send to the log file. See the instructions in the pin.conf file.</td>
</tr>
<tr>
<td>dm_in_batch_size</td>
<td>dm</td>
<td>Specifies the number of objects to retrieve from subtables (arrays or substructs) in a search query. See the instructions in the pin.conf file.</td>
</tr>
<tr>
<td>dm_logfile</td>
<td>dm</td>
<td>Specifies the full path to the log file used by this DM. See &quot;Changing the Name or Location of a Log File&quot;.</td>
</tr>
<tr>
<td>dm_max_per_fe</td>
<td>dm</td>
<td>Specifies the maximum number of connections for each front end. See the instructions in the pin.conf file.</td>
</tr>
<tr>
<td>dm_mr_enable</td>
<td>dm</td>
<td>Specifies whether to use database cursors to fetch multiple rows. See &quot;Configuring Oracle Databases&quot; in BRM Installation Guides.</td>
</tr>
<tr>
<td>dm_n_be</td>
<td>dm</td>
<td>Specifies the number of back ends the program creates and uses. See &quot;Configuring DM Front Ends and Back Ends&quot;.</td>
</tr>
<tr>
<td>dm_n_fe</td>
<td>dm</td>
<td>Specifies the number of front ends the program creates and uses. See &quot;Configuring DM Front Ends and Back Ends&quot;.</td>
</tr>
</tbody>
</table>
### Table 38–2 (Cont.) DM pin.conf Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dm_port</td>
<td>dm</td>
<td>Specifies the port number for this DM. See &quot;About Connecting BRM Components&quot;.</td>
</tr>
<tr>
<td>dm_restart_children</td>
<td>dm</td>
<td>Specifies whether to replace child processes. See &quot;Replacing Failed DM Child Processes&quot;.</td>
</tr>
<tr>
<td>dm_restart_delay</td>
<td>dm</td>
<td>Specifies the interval delay when DM back ends are spawned and respawned. See &quot;Customizing BRM Multithreaded Client Applications&quot; and &quot;Configuring Your Multithreaded Application&quot; in BRM Developer's Guide.</td>
</tr>
<tr>
<td>dm_sequence_cache_size</td>
<td>dm</td>
<td>(Optional) Specifies the number of Portal object IDs (POIDs) to be cached when each instance of a DM is started. The default is 1000. The POIDs are cached in the memory when the DM is started. Whenever a database object is created, the DM uses the POIDs instead of accessing the database each time.</td>
</tr>
<tr>
<td>dm_shmsize</td>
<td>dm</td>
<td>Specifies the size of the shared-memory segment shared between the front and back ends for this BRM process. See &quot;Setting DM Shared Memory Size&quot;.</td>
</tr>
</tbody>
</table>
| dm_sql_retry    | dm      | Specifies the number of times an SQL statement is retried if the ORA-01502: index 'PINPAPI_EVENT_ITEM_OBJ__ID' or partition of such index is in unusable state error occurs.  
Note: This is not a mandatory parameter in the pin.conf file. The default behavior is not to try running the SQL statement if the error occurs.  
See 'Problem: ORA-01502: Index 'PINPAPI_EVENT_ITEM_OBJ__ID' or Partition of Such Index Is in Unusable State'. |
| dm_trans_be_max | dm      | Specifies the maximum number of back ends that can be used for processing transactions. See "Configuring DM Front Ends and Back Ends".                                                                     |
| dm_trans_timeout| dm      | Specifies the time-out value for receiving the next opcode request from an application. See "Setting the DM Time Interval between Opcode Requests".                                                             |
| extra_search    | dm      | Specifies whether to perform an extra search count (*) on subtables for optimal memory allocation. See "Optimizing Memory Allocation during Database Searches".                                                  |
| sm_database     | dm      | Specifies the database alias name. See "Connecting a Data Manager to the BRM Database".                                                                                                                   |
| sm_id           | dm      | Specifies the database user name that the DM uses to log in to the BRM database. This entry is set when you install BRM, but it can be changed. See "Connecting a Data Manager to the BRM Database". |
Table 38–2 (Cont.) DM pin.conf Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dm_http_100_continue</td>
<td>dm</td>
<td>Specifies whether the DM waits for and reads a 100 Continue response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See ”Configuring EAI Manager to Publish to an HTTP Port” in BRM Developer’s Guide.</td>
</tr>
<tr>
<td>dm_http_agent_ip</td>
<td>dm</td>
<td>Specifies a pointer to the HTTP agent.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See ”Configuring EAI Manager to Publish to an HTTP Port” in BRM Developer’s Guide.</td>
</tr>
<tr>
<td>dm_http_delim_crlf</td>
<td>dm</td>
<td>Specifies the HTTP server-dependent delimiter used in the header.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See ”Configuring EAI Manager to Publish to an HTTP Port” in BRM Developer’s Guide.</td>
</tr>
<tr>
<td>dm_http_header_send_host_name</td>
<td>dm</td>
<td>Specifies whether the DM sends the host name as part of the header.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See ”Configuring EAI Manager to Publish to an HTTP Port” in BRM Developer’s Guide.</td>
</tr>
<tr>
<td>dm_http_read_success</td>
<td>dm</td>
<td>Specifies whether the DM waits for and reads a 20x success response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See ”Configuring EAI Manager to Publish to an HTTP Port” in BRM Developer’s Guide.</td>
</tr>
<tr>
<td>dm_http_url</td>
<td>dm</td>
<td>(Optional) Specifies the complete URL of the HTTP server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See ”Configuring EAI Manager to Publish to an HTTP Port” in BRM Developer’s Guide.</td>
</tr>
<tr>
<td>eai_pointer</td>
<td>cm</td>
<td>Specifies where to find the EM that provides the opcode for the EAI Manager.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See ”Configuring the EAI DM” in BRM Developer’s Guide.</td>
</tr>
<tr>
<td>cm_eai_group</td>
<td>cm</td>
<td>Specifies the member opcode in a group provided by the EAI Manager.</td>
</tr>
</tbody>
</table>

Table 38–3 EAI Manager pin.conf Entries

Table 38–3 lists the EAI Manager pin.conf entries.

<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sm_oracle_ddl</td>
<td>dm</td>
<td>Specifies whether to use Data Definition Language (DDL) when object types are updated in the data dictionary tables.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See ”Using DDL When Updating the Data Dictionary Tables” in BRM Developer’s Guide.</td>
</tr>
<tr>
<td>sm_pw</td>
<td>dm</td>
<td>Specifies the password for the user specified in the sm_id entry.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See ”Configuring the Data Manager (DM) for AES Encryption” in BRM Developer’s Guide.</td>
</tr>
<tr>
<td>stmt_cache_entries</td>
<td>dm</td>
<td>Specifies the maximum number of Oracle statement handles to cache to improve performance.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See ”How to Use the Statement-Handle Cache”.</td>
</tr>
</tbody>
</table>
Table 38–4 lists the MTA Framework `pin.conf` entries.

<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>em_group</code></td>
<td>cm</td>
<td>Specifies a member opcode in a group of opcodes provided by an EM. See &quot;Configuring the Connection Manager for EAI&quot; in BRM Developer’s Guide.</td>
</tr>
<tr>
<td><code>em_pointer</code></td>
<td>cm</td>
<td>Specifies where to find EMs that provide other opcodes to CM. See &quot;Configuring the CM to Send Real-Time Requests to the NET_EM Module&quot;.</td>
</tr>
<tr>
<td><code>enable_publish</code></td>
<td><code>fm_publish</code></td>
<td>Enables publishing of business events by using the EAI Manager. See &quot;Configuring the Connection Manager for EAI&quot; in BRM Developer’s Guide.</td>
</tr>
<tr>
<td><code>plugin_name</code></td>
<td><code>dm</code></td>
<td>Specifies a pointer to a shared library that contains the code that implements the required interfaces of <code>dm_eai</code> as defined in <code>dm_eai_plugin.h</code>. See &quot;Configuring the EAI DM&quot; in BRM Developer’s Guide.</td>
</tr>
</tbody>
</table>

Multithreaded Application (MTA) Framework `pin.conf` Entries

Table 38–4 lists the MTA Framework `pin.conf` entries.

Table 38–4  MTA Framework `pin.conf` Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>children</code></td>
<td><code>pin_inv_accts</code></td>
<td>Specifies the number of worker threads spawned to perform the specified work. See &quot;Tuning the Number of Children for Billing Utilities (fetch_size)&quot;.</td>
</tr>
<tr>
<td></td>
<td><code>pin_inv_export</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>pin_inv_send</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>pin_inv_upgrade</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>pin_mta</code></td>
<td></td>
</tr>
<tr>
<td><code>cm_login_module</code></td>
<td><code>cm</code></td>
<td>Specifies the protocol for verifying applications that try to log in to the CM. See &quot;Turning Off the Checking of Logons and Passwords&quot;.</td>
</tr>
<tr>
<td><code>enable_ara</code></td>
<td><code>pin_mta</code></td>
<td>Enables revenue assurance activities through the various MTA applications (for example, <code>pin_bill_accts</code>, <code>pin_cycle_fees</code>, <code>pin_collect</code>, and <code>pin_inv_accts</code>). See &quot;Configuring BRM Billing to Collect Revenue Assurance Data&quot; in BRM Collecting Revenue Assurance Data.</td>
</tr>
<tr>
<td><code>fetch_size</code></td>
<td><code>pin_mta</code></td>
<td>Specifies the number of objects received from the database in a block and cached in system memory for processing. See &quot;Tuning the Account Cache Size for Billing Utilities (fetch_size)&quot;.</td>
</tr>
<tr>
<td><code>hotlist</code></td>
<td><code>pin_mta</code></td>
<td>Specifies the path and file name of the hotlist file. See &quot;Creating Configuration Files for BRM Utilities&quot;.</td>
</tr>
</tbody>
</table>
Table 38–4  (Cont.) MTA Framework pin.conf Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
</tr>
</thead>
</table>
| logfile      | pin_mta | Specifies the name and location of the log file to record debug, warning, and error messages of running MTA-based applications.  
|              |         | See "About the BRM MTA Framework” in BRM Developer’s Guide.                   |
| login_name   | nap     | Specifies the user login name.                                              |
|              |         | See "Configuring System Passwords” in BRM Developer’s Guide and "Login Information for Optional Service Integration Components”. |
| login_pw     | nap     | Specifies the login password.                                               |
|              |         | See "Configuring System Passwords” in BRM Developer’s Guide and "Login Information for Optional Service Integration Components”. |
| login_type   | nap     | Specifies whether a login name and password are required.                   |
|              |         | See "Configuring System Passwords” in BRM Developer’s Guide and "Login Information for Optional Service Integration Components”. |
| loglevel     | pin_mta | Specifies how much information is recorded in the log file specified by the logfile parameter in the pin.conf file.  
|              |         | See "Setting the Reporting Level for Logging Messages".                     |
| max_errs     | pin_mta | Specifies the maximum number of errors allowed in the application.          |
|              |         | See "Creating Configuration Files for BRM Utilities".                       |
| max_time     | pin_mta | Specifies the maximum time, measured from application start time, for job processing before the application exits.  
|              |         | See "Creating Configuration Files for BRM Utilities".                       |
| multi_db     | pin_mta | Specifies whether the application works with a BRM multidatabase system.    |
|              |         | See "Using Multithreaded Applications with Multiple Databases" in BRM Developer’s Guide. |
| per_batch    | pin_mta | Specifies the number of account objects processed by each worker thread in batch mode.  
|              |         | See "Tuning the Batch Size for Billing Utilities (per_batch)".              |
| per_step     | pin_mta | Specifies the number of account objects returned by each search step.       |
|              |         | See "Setting the Batch Size for Invoice Utilities (per_step)".              |
| respawn_threads | pin_mta | Specifies whether to respawn worker threads if they exit because of an error.  
|              |         | See "Creating Configuration Files for BRM Utilities".                       |
Table 38–4  (Cont.) MTA Framework pin.conf Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Program</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>retry_mta_srch</td>
<td>pin_mta</td>
<td>Specifies whether to retry MTA searches when an insufficient memory error occurs.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Creating Configuration Files for BRM Utilities&quot;.</td>
</tr>
<tr>
<td>return_worker_error</td>
<td>pin_mta</td>
<td>Specifies whether to return an error code when there is an error in any of the threads.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Creating Configuration Files for BRM Utilities&quot;.</td>
</tr>
<tr>
<td>userid</td>
<td>cm</td>
<td>Specifies the CM for logging in to the DM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See &quot;Configuration Entry Syntax&quot;.</td>
</tr>
</tbody>
</table>
This chapter lists the `business_params` entries used for configuring Oracle Communications Billing and Revenue Management (BRM).

See also "Business Logic pin.conf Reference" and "System Administration pin.conf Reference" for more information.

**Accounts Receivable business_params Entries**

Table 39–1 lists the Accounts Receivable `business_params` entries.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>AutoWriteOffReversal</td>
<td>Enables automatic write-off reversal on receipt of payment.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td>See &quot;Enabling Automatic Write-Off Reversals during Payment Collection&quot; in BRM Managing Accounts Receivable.</td>
<td></td>
</tr>
<tr>
<td>BalanceMonitoring</td>
<td>Enables balance monitoring.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td>See &quot;Enabling Balance Monitoring in BRM&quot; in BRM Managing Accounts Receivable.</td>
<td></td>
</tr>
<tr>
<td>LockConcurrency</td>
<td>Locks the account object, system-wide, at the account level or balance-group level.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td>See &quot;Disabling Granular Object Locking&quot; in BRM Developer's Guide.</td>
<td></td>
</tr>
<tr>
<td>NonrefundableCreditItems</td>
<td>Specifies the types of items that will not be refunded with an outstanding credit balance.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td>See &quot;Defining Nonrefundable Items&quot; in BRM Configuring and Running Billing.</td>
<td></td>
</tr>
<tr>
<td>PaymentIncentive</td>
<td>Enables payment incentives on early payment-in-full.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td>See &quot;Enabling BRM for Payment Incentives&quot; in BRM Configuring and Collecting Payments.</td>
<td></td>
</tr>
<tr>
<td>PaymentSuspense</td>
<td>Enables payment suspense management.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td>See &quot;Enabling Payment Suspense in BRM&quot; in BRM Configuring and Collecting Payments.</td>
<td></td>
</tr>
</tbody>
</table>
**Billing business_params Entries**

Table 39–1 (Cont.) Accounts Receivable business_params Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>SearchBillAmount</td>
<td>Enables searches for /bill objects by total due amount. See “Finding Bills by Due Amount” in BRM Configuring and Collecting Payments.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>SortValidityBy</td>
<td>Specifies the default consumption rule when consuming validity-based sub-balances. See “Setting the Default Consumption Rule” in BRM Setting Up Pricing and Rating.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>WriteOffLevel</td>
<td>Specifies the level of write-off (account level, bill-unit level, or bill level) to track write-off reversals. See “Configuring Write-Offs and Write-Off Reversals” and “How BRM Performs Write-Offs” in BRM Managing Accounts Receivable.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
</tbody>
</table>

**Billing business_params Entries**

Table 39–2 Billing business_params Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>AcctCycleDelayPeriod</td>
<td>Use when billing occurred after an event was rated by Pipeline Manager but before processing by RE Loader (that is, the event has not impacted the item). In such cases, RE Loader tries to locate the item from the next cycle if the event is created after billing in more than X days. See “Configuring an Accounting Cycle Delay Period” in BRM Configuring Pipeline Rating and Discounting.</td>
<td>Not cached by the CM.</td>
</tr>
<tr>
<td>ApplyCycleFeeForBillNow</td>
<td>Indicates whether to apply cycle forward arrears and cycle arrears fees for Bill Now. See “Prorating Cycle Arrears and Cycle Forward Arrears for Bill Now” in BRM Configuring and Running Billing.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>AutoTriggeringLimit</td>
<td>Suppresses auto-triggered billing for events processed after the billing_delay interval. See “Disabling Auto-Triggered Billing by Setting AutoTriggeringLimit” in BRM Configuring and Running Billing.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>BillingCycleOffset</td>
<td>Specifies the hours of the day when the accounting and billing cycles start. See “Configuring the Billing Cutoff Time” in BRM Configuring and Running Billing.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>BillingFlowDiscount</td>
<td>Indicates the order of billing for discount-parents and discount-members (that is, who is billed first). See “Setting Up Billing for Sponsorship” in BRM Configuring and Running Billing.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
</tbody>
</table>
Table 39–2  (Cont.) Billing business_params Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>BillingFlowSponsorship</td>
<td>Specifies the order of billing sponsors and sponsored accounts.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>BillTimeDiscountWhen</td>
<td>Enables billing-time discounts at the end of the billing cycle:</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td>See &quot;Defining When Billing-Time Discounts Are Applied&quot; in BRM Configuring Pipeline Rating and Discounting.</td>
<td></td>
</tr>
<tr>
<td>CacheResidencyDistinction</td>
<td>Enables BRM to handle all objects as postpaid.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td>Cache residency objects are objects associated with prepaid subscribers.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>See &quot;Enabling Cache Residency Distinction&quot; and &quot;About Convergent BRM Systems&quot;.</td>
<td></td>
</tr>
<tr>
<td>CreateTwoBillNowBillsInDelay</td>
<td>Creates two Bill_Now objects during the billing delay interval.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td>The first Bill_Now object includes charges for the previous cycle.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The second Bill_Now object contains charges from the events for the new (next) cycle.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>See &quot;Creating Two Bills during the Delayed Billing Period&quot; in BRM Configuring and Running Billing.</td>
<td></td>
</tr>
<tr>
<td>DefaultBusinessProfile</td>
<td>Sets the default cache type for the system.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td>By default, all reference object instances get assigned an object cache type value of DEFAULT (Convergent business profile) if they have not been associated with a business profile.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>See &quot;Changing the Default Business Profile&quot;.</td>
<td></td>
</tr>
<tr>
<td>EnableARA</td>
<td>Enables Revenue Assurance Manager.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>GenerateJournalEpsilon</td>
<td>Indicates whether to apply rounding to item totals before calculating the bill total.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td>See &quot;Configuring BRM to Record Rounding Difference&quot; in BRM Setting Up Pricing and Rating.</td>
<td></td>
</tr>
<tr>
<td>MoveDayForward</td>
<td>Specifies an appropriate option for the 31 billing feature.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td>If this feature is not used, the billing day of month (DOM) cannot be greater than 28. Otherwise, any day of the month can be used for billing.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>See &quot;Setting the Forward and Back Billing Options&quot; in BRM Configuring and Running Billing.</td>
<td></td>
</tr>
<tr>
<td>NonCurrencyResourceJournaling</td>
<td>Controls the creation of /journal objects for non-currency resources.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td>See &quot;Disabling /journal Objects for Non-Currency Resources&quot; in BRM Collecting General Ledger Data.</td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
<td>Implementation</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>PaymentIncentive</td>
<td>Enables payment (grant) incentives when a payment is received early in the billing cycle.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td>See &quot;Enabling BRM for Payment Incentives&quot; in BRM Configuring and Collecting Payments.</td>
<td></td>
</tr>
<tr>
<td>ProdEndOffsetPlanTransition</td>
<td>Enables a phased-out service to remain active for any number of days between 1 and 31. ❧ Configuring Services for a Generation Change in BRM Managing Customers.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>RemoveSponsoree</td>
<td>Determines whether to remove a member account from sponsor groups. ❧ How Account Status Changes Affect Sponsor Groups in BRM Managing Accounts Receivable.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>RerateDuringBilling</td>
<td>Enables borrowing from the rollover amount during the current billing cycle. ❧ Modifying Business Parameters to Enable Rerating and Rollover Correction in BRM Configuring and Running Billing.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>RolloverCorrectionDuringBilling</td>
<td>Enables borrowing from the rollover amount during the current billing cycle. ❧ Modifying Business Parameters to Enable Rerating and Rollover Correction in BRM Configuring and Running Billing.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>SequentialCycleDiscounting</td>
<td>Enables BRM to evaluate cycle fee discounts purchased or canceled mid-cycle in conjunction with other discounts that are valid during the same period.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td>See &quot;Enabling Sequential Discounting of Cycle Fees&quot; in BRM Configuring Pipeline Rating and Discounting.</td>
<td></td>
</tr>
<tr>
<td>SortValidityBy</td>
<td>Defines the default value for consumption rule. When a customer uses a service, BRM needs to know which minutes (or sub-balance) to use first. You use resource consumption rules to specify the order in which resource sub-balances are consumed, according to the validity start time and end time.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td>See &quot;Setting the Default Consumption Rule&quot; in BRM Setting Up Pricing and Rating.</td>
<td></td>
</tr>
</tbody>
</table>
**Table 39–3**  (Cont.)  **Billing business_params Entries**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Implementation</th>
</tr>
</thead>
</table>
| SplitSponsorItemByMember    | Enables you to divide accumulated charges across sponsored members of an account.  
                                | See “Creating Custom Sponsored Items” in BRM Configuring and Running Billing. | Cached by the CM. Restart the CM after changing this entry.                   |
| SubBalValidity              | Enables you to extend the validity period of the original sub-balance when a subscription service is transferred.  
                                | See “Configuring Sub-Balance Validity for Subscription Service Transfer” in BRM Managing Customers. | Cached by the CM. Restart the CM after changing this entry.                   |
| ValidateDiscountDependency  | Enables discount exclusion rules, which establish a mutually exclusive relationship between discounts or between a discount and a plan.  
                                | See “Configuring and Defining Exclusion Rules” in BRM Configuring Pipeline Rating and Discounting. | Cached by the CM. Restart the CM after changing this entry.                   |

**Customer Management business_params Entries**

Table 39–3 lists the Customer Management business_params entries.

**Table 39–3**  **Customer Management business_params Entries**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Implementation</th>
</tr>
</thead>
</table>
| AutomatedMonitorSetup         | Enables automated balance monitoring.  
                                | See “Enabling AMS in BRM” in BRM Managing Accounts Receivable.               | Cached by the CM. Restart the CM after changing this entry.                   |
| BestPricing                   | Enables best pricing feature. If best pricing is enabled, the best pricing calculation is performed for the member service at billing.  
                                | See “Enabling Best Pricing” in BRM Configuring and Running Billing.          | Cached by the CM. Restart the CM after changing this entry.                   |
| CancelFullDiscountImmediate   | Enables BRM to cancel discounts immediately when the discount validity rule is set to **Full Discount**.  
                                | See “Managing Discount End Dates During Mid-Cycle Cancellations” in BRM Configuring Pipeline Rating and Discounting. | Cached by the CM. Restart the CM after changing this entry.                   |
| DiscountBasedOnContractDaysFeature | Enables the contract days counter (CDC), which provides a discount based on the cumulative number of contract days.  
                                | See “Enabling Support for Discounts Based on Contract Days” in BRM Configuring Pipeline Rating and Discounting. | Cached by the CM. Restart the CM after changing this entry.                   |
### Table 39–3 (Cont.) Customer Management business_params Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Implementation</th>
</tr>
</thead>
</table>
| PaymentSuspense                   | Enables Payment Suspense Manager, which suspends payments exhibiting certain problems instead of failing or wrongly allocating them, and postpones them for later investigation. This enables the payment posting process to be completed without requiring immediate intervention to fix the errors.  
See “Enabling Payment Suspense in BRM” in BRM Configuring and Collecting Payments. | Cached by the CM. Restart the CM after changing this entry.                                         |
| RolloverTransfer                  | Enables rollover transfers.  
See "Enabling Rollover Transfers in BRM" in BRM Managing Accounts Receivable.                                                                                                                             | Cached by the CM. Restart the CM after changing this entry.                                         |
| SubsDis74BackDateValidations      | Enables users to create accounts with services or resources backdated prior to the account creation date.  
See “Allowing Accounts To Be Created with Backdated Services or Resources” in BRM Managing Customers.                                       | Cached by the CM. Restart the CM after changing this entry.                                         |

### General Ledger business_params Entries

Table 39–4 lists the General Ledger business_params entries.

### Table 39–4 General Ledger business_params Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Implementation</th>
</tr>
</thead>
</table>
| CustomJournalUpdate               | Enables custom updates to general ledger data.  
See "Enabling Custom Updates to G/L Data” in BRM Collecting General Ledger Data.                                                                                                                                         | Cached by the CM. Restart the CM after changing this entry.                                         |
| GeneralLedgerReporting            | Use general ledger reporting.  
See "Enabling and Disabling General Ledger Collection in BRM” in BRM Collecting General Ledger Data.                                                                                                                   | Cached by the CM. Restart the CM after changing this entry.                                         |
| UseActualBilledTimeForGLReport    | Enables generation of billed revenue general ledger reports to use the actual billing date for the billed revenue.  
See "Generating General Ledger Reports For Billed Revenue to be Based on the Actual Time of Billing” in BRM Collecting General Ledger Data.                                        | Cached by the CM. Restart the CM after changing this entry.                                         |

### Invoicing business_params Entries

Table 39–5 lists the Invoicing business_params entries.
## Table 39–5  Pricing and Rating business_params Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADSTaxHandle</td>
<td>Groups taxes on invoices based on the tax supplier IDs.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td>See &quot;Aggregating Taxes on Invoices&quot; in BRM Designing and Generating Invoices.</td>
<td></td>
</tr>
<tr>
<td>EnableInvoicingIntegration</td>
<td>Generates invoice documents by using the BRM-BI Publisher integration framework.</td>
<td>This entry is read by the utility when it runs. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td>See &quot;Enabling the BRM-BI Publisher Integration&quot; in BRM Designing and Generating Invoices.</td>
<td></td>
</tr>
<tr>
<td>InvoiceStorageType</td>
<td>Specifies the format in which to store invoices in the database.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td>See &quot;Specifying the Default Format in Which to Store Invoices in BRM&quot; in BRM Designing and Generating Invoices.</td>
<td></td>
</tr>
<tr>
<td>PromotionDetailDisplay</td>
<td>Displays promotion details on invoices.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td></td>
<td>See &quot;Specifying Whether BRM Displays Promotion Details on Invoices&quot; in BRM Designing and Generating Invoices.</td>
<td></td>
</tr>
<tr>
<td>SubARItemsIncluded</td>
<td>Displays A/R items for each subordinate account on the parent invoice.</td>
<td>This entry is read by the utility when it runs. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td>See &quot;Setting Defaults for Hierarchical Group Invoices&quot; in BRM Designing and Generating Invoices.</td>
<td></td>
</tr>
<tr>
<td>ThresholdSubordsDetail</td>
<td>Sets the threshold for including subordinate account details on parent detailed invoices.</td>
<td>This entry is read by the utility when it runs. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td>See &quot;Setting Defaults for Hierarchical Group Invoices&quot; in BRM Designing and Generating Invoices.</td>
<td></td>
</tr>
<tr>
<td>ThresholdSubordsSummary</td>
<td>Sets the threshold for including subordinate account details on parent summary invoices.</td>
<td>This entry is read by the utility when it runs. You do not need to restart the CM.</td>
</tr>
<tr>
<td></td>
<td>See &quot;Setting Defaults for Hierarchical Group Invoices&quot; in BRM Designing and Generating Invoices.</td>
<td></td>
</tr>
</tbody>
</table>

## Pricing and Rating business_params Entries

Table 39–6 lists the Pricing and Rating business_params entries
Table 39–6 Pricing and Rating business_params Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>AllocateReratingAdjustments</td>
<td>Determines whether to allocate automatic adjustments from rerating to original bills</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>BatchRatingPipeline</td>
<td>Enables Account Synchronization to pass business events between pin_rerate and Pipeline Manager. See “Specifying Whether the Batch Rating Pipeline Is Enabled” in BRM Setting Up Pricing and Rating.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>EnableEras</td>
<td>Specifies how to enable ERAs. See “Filtering the ERAs Considered During Rating and Discounting”.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>EnableGlobalChargeSharing</td>
<td>Enables global charge sharing. See “Enabling Global Charge Sharing Searches during Discounting” in BRM Managing Accounts Receivable.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>EnableTailormadeCache</td>
<td>Specifies if tailor-made products must be maintained in the rating cache. See “Enabling and Disabling the Caching of Customized Products”.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>OverrideCreditLimit</td>
<td>Enables system-wide credit limit override. See ‘Configuring the System-Wide Override Credit Limit’ in BRM Managing Customers.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>ProductsDiscountsThreshold</td>
<td>Specifies the maximum number of products or discounts that can be cached by the real-time rating engine. See “Configuring the Maximum Number of Products and Discounts Cached”.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
<tr>
<td>RestrictResourceValidityToOffer</td>
<td>Restrict resource validity end time to the end time of the product or discount that grants the resource. See ‘Configuring Real-Time Rating to Restrict Resource Validity End Time’ in BRM Setting Up Pricing and Rating.</td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
</tr>
</tbody>
</table>

Selective Account Loading business_params Entries

Table 39–7 lists the Selective Account Loading business_params entries.

Table 39–7 Selective Account Loading business_params Entries

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>CacheResidenciesForBatchPipeline</td>
<td>Enables Pipeline Manager to selectively load accounts. See “Configuring Pipeline Manager for Selective Account Loading”.</td>
<td>Cached by the CM and Pipeline Manager. Restart the CM and Pipeline Manager after changing this entry.</td>
</tr>
</tbody>
</table>

Subscription business_params Entries

Table 39–8 lists the Subscription business_params entries.
**Table 39–8 Subscription business_params Entries**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>AllowBackdateNoRerate</td>
<td>Enables backdating beyond the number of billing cycles specified in the <code>num_billing_cycles</code> entry without requesting to automatically rerate.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
<td></td>
</tr>
<tr>
<td>TransferScheduledActions</td>
<td>Specifies whether to transfer pending scheduled actions associated with an existing subscription when transferring the subscription to a different account.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>See the discussion of transferring a subscription service in <code>BRM Managing Customers</code>.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
<td></td>
</tr>
<tr>
<td>UsePrioritySubscriptionFees</td>
<td>Specifies whether to use product priority while applying cycle fee for products.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>See &quot;Enabling Product Priority While Applying Cycle Fee&quot; in <code>BRM Configuring and Running Billing</code>.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
<td></td>
</tr>
</tbody>
</table>

**System Administration business_params Entries**

Table 39–9 lists the System Administration `business_params` entries.

**Table 39–9 Systems Administration business_params Entries**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaxLoginAttempts</td>
<td>Specifies the maximum number of invalid login attempts to be allowed with an incorrect password.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>See &quot;Configuring the Maximum Number of Invalid Login Attempts&quot;.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cached by the CM. Restart the CM after changing this entry.</td>
<td></td>
</tr>
</tbody>
</table>