Administrating Telco Service Manager

V4.0
Document ID: TSGN-04-4.0-01
Date Published: 9.10.03

edocs, Inc., One Apple Hill Drive, Suite 301, Natick, MA 01760

Preface

In This Section

Using this Manual

Welcome to Administrating Telco Service Manager (TSM).

This manual covers the different tasks that administrators need to carry out when working with account management applications built using TSM.

Before You Get Started

You should be familiar with the following:

- Your application architecture
- Programming Java and Java Server pages
- Designing or working with databases
- eXtended Markup Language (XML)

Who Should Read this Manual

This manual is primarily for administrators of deployed Account Management solutions. However, there are other topics covered in this manual that may interest other members of the project development team.

- Administrators

  You will find the information you need to manage a deployed solution. The information in this manual starts with information about the basic administration tasks. You will also find detailed information about other administration tasks, such as purging information, fine-tuning specific application settings, or interpreting logs to pinpoint problems.

- Developers

  You will find information about running and administrating components of your development environment. You may especially be interested in the basic administration tasks and the detailed administration information dealing with various components. You may also want to consult the Administration Tool Reference for the comprehensive list of administration tools and their commands.

- Project Architect

  You will find information about the different components which may require administration or intervention by administrators. There are some features that may be of interest when designing your solution. For instance, you need to be aware of how requests are managed and how the solution handles persistent action managers. You may also be interested in the various ways of auditing the application which are explained in detail. You should also look at the sections covering distributed architectures.

- Project Manager

  You will find information about the configuration and processes of the runtime environment you need to take into account when planning the development of your solution. There is important information about how the solution handles requests. It also explains how to reload reference information during runtime. You can also look into the auditing and logging features of the solution.

How this Manual is Organized

This manual contains the following chapters:

- **Overview of Administrating**

  This chapter covers the basics of managing Account Management solutions:

  - The role of an administrator
  - Different components and third-party software these components use
  - Configuration files and their location

- **Basic Administration Tasks**

  This chapter covers the basic administration tasks:

  - Starting and stopping the Synchronizer connector
  - Starting and stooping the OSS connector
  - Starting and stooping the Approval Sequencer

- **Administrating Account Applications**

  This chapter covers the following administration tasks:

  - Managing the Synchronizer connector
  - Managing requests
  - Managing trouble tickets
  - Managing persistent action managers
  - Reloading reference data
  - Monitoring account applications
  - Auditing account applications

- **Managing Distributed Architectures**

  This chapter covers administrating solutions in distributed architectures:

- Configuring shared directories
- Managing sessions

- **Administration Tool Reference**

  This appendix is an administration tool reference guide. It covers the location, configuration and use of the administration tools

  The administration tools covered include:

  - Synchronizer connector tools
  - OSS connector tools
  - Approval Sequencer tools
  - CID administration tool

What Typographical Changes and Symbols Mean

This manual uses the following conventions:

TYPEFACE	MEANING	EXAMPLE
Italics	Manuals, topics or other important items	Refer to *Developing Connectors*.
Small Capitals	Software and Component names	Your application uses a database called the CID.
Fixed Width	File names, commands, paths, and on screen commands	Go to `//home/my file`

Finding the Information You Need

The product suite comes with comprehensive documentation set that covers all aspects of building Account Management solutions. You should always read the release bulletin for late-breaking information.

Getting Started

If you are new to the edocs Telco Solutions, you should start by reading *Introducing Telco Service Manager*. This manual contains an overview of the various components along with a list of the available features. It introduces various concepts and components you must be familiar with before moving on to more specific documentation. Once you have finished, you can read the manual which covers different aspects of working with the application. At the beginning of each manual, you will find an introductory chapter which covers concepts and tasks.

Designing Your Solution

While reading *Introducing Telco Service Manager*, you should think about how the different components can address your Account Management Solution's needs.

You can refer to *Developing Telco Service Manager* for information about extending the object model, application security, and other design issues. The *CID Reference Guide* also gives you the information about how the information in your solution is managed and stored.

Installing Your Telco Service Manager

You should start by reading the Release Bulletin. For detailed installation and configuring information, refer to *Installing Telco Service Manager*. This manual covers installing TSM on one or more computers. It also contains the information you need to configure the different components you install. You might also refer to *Developing Telco Service Manager* and *Developing Connectors for Telco Service Manager* as these manuals contain information on customizing applications and working with other software.

Building Account Management Solutions

If you are designing and programming *Telco Service Manager*, you have several different sources of information. If you are programming the user interface of the solution, you should read *Developing User Interfaces for Telco Service Manager*. You also refer to the BLM Specification for detailed information about programming the user interface. For configuring the various components, you refer to *Installing Telco Service Manager*and sections in other documents which deal with the component to configure.

If you are working with the business logic of your solution, you should read *Developing Telco Service Manager*. You can also refer to the *BLM Reference Guide* for more information about the design and structure of the BLM object model. For information about how this information is stored, you should refer to the *CID Reference Guide* along with the CID Reference documentation for your database. In order to develop your application, you most likely will need to install and run the Loopback Connector. This component mimics back-end applications for development purposes. For information about installing and running this component, refer to *Using the Loopback Connector with Telco Service Manager*.

Integrating Account Management Solutions

If you are involved in configuring your solution to work with Operation Support Software (OSS), you should read *Developing Connectors with Telco Service Manager*. This manual helps you understand the integration architecture and shows you how to build connectors to connect to today's market-leading OSS software. You can also read *Using the Loopback Connector with Telco Service Manager* for information about a connector built for development purposes. Other manuals you can refer to for information about configuring your application include *Introducing Telco Service Manager* and *Developing Telco Service Manager*.

Managing Telco Service Manager (TSM)

If you are responsible for managing TSM, you should read the *Installing Telco Service Manager* for information about configuring various components and information about working with different application servers. *Administrating Telco Service Manager* covers what you need to know about managing your solution at runtime. For information about OSS systems, you should read *Developing Connectors with Telco Service Manager*.

If You Need Help

Technical support is available to customers who have valid maintenance and support contracts with edocs. Technical support engineers can help you install, configure, and maintain your edocs application.

To reach the U.S. Service Center, located in Natick, MA (Monday through Friday 8:00am to 8:00pm EST):

- Telephone: 508.652.8400
- Toll Free: 877.336.3362
- E-support: support.edocs.com (This requires a one-time online registration)
- E-mail: support@edocs.com

When you report a problem, please be prepared to provide us the following information:

- What is your name and role in your organization?
- What is your company's name?
- What is your phone number and best times to call you?
- What is your e-mail address?
- In which edocs product did a problem occur?
- What is your Operating System version?
- What were you doing when the problem occurred?
- How did the system respond to the error?
- If the system generated a screen message, please send us that screen message.
- If the system wrote information to a log file, please send us that log file.

If the system crashed or hung, please tell us.

Contents

CHAPTER 1

Overview of Administrating

In This Section

About Administrating

TSM applications are designed to run without user intervention or maintenance. To ensure that your system runs smoothly and meets the availability and performance demands your users expect, you have to perform some maintenance and administrative tasks.

Along with this set of tasks, you can also monitor the behavior of your application to make sure it is running without problems. You can use the message logger and the user event logger to pinpoint problems or analyze user actions.

Administrating your TSM application also includes managing some of the system settings and properties. For example, establishing and maintaining system security and data integrity are all part of managing TSM.

Administrating TSM involves:

- **Basic Administration Tasks**

  Starting and stopping connectors, testing connections, and so on

- **Administrating TSM**

  Monitoring your TSM application and other software, administrating the CID, managing message and notification queues, and checking logs

- **Managing Distributed Architectures**

  Managing session serialization

When managing your TSM application, you must remember that it is part of a large and complex system that works with several different systems and software. Not only do you need to be familiar with how this software works, you need to understand how your TSM application interacts with these systems and software.

Overview of the Application Architecture

Your TSM application has the following components:

- Personalization Manager
- CSS Engine
- Customer Interaction Datastore (CID)
- SmartLink Framework

Each of these components works with and communicate with different software applications. The software packages that these components work with include:

- **Web servers**

  The Web server is the main entrance to MyWeb channel. The Web server hosts Web sites and their different files as well as managing the communication between Web browsers and the application server. Web servers can also host Personalization Manager files.

- **Application servers**

  The application server creates and manages dynamic content of applications. It manages the communication between the Web server (and other channels) and databases. The application server serves as the foundation of the CSS Engine and can also host Personalization Manager and JSPF files.

- **Databases**

  Databases contain the information your application manages. The CID contains the core application information.

- **Operational Support Systems (OSS)**

  The OSS contain the information your TSM application uses to enable users to manage their accounts. The information managed by TSM in the CID comes from OSS. The SmartLink Framework manages the transfer of information between the CID and OSS.

About Configuration Files and their Location

During installation, there are two identical sets of TSM configuration files. This is done to respect the requirements of the J2EE Web Application aRchive (WAR) file specifications and to help you easily deploy your TSM application.

However, administrating may involve modifying some of the configuration files.

The configuration files are found in:

- `<home_dir>/classes/nmycfg`

  This directory contains the reference set of core configuration files. These files are the default set of configuration files and are used when developing TSM applications.

- `<home_dir>/channels/WEB-INF/classes/nmycfg`

  This directory contains the Channel configuration files. These files are the configuration files that are deployed when using the J2EE WAR file.

The location of deployed Channel configuration files depends on how your application server handles and deploys WAR files. To indicate the location of deployed Channel configuration files, this manual uses `<app_dir>`.

For more information about WAR and configuration files, refer to *Installing TSM* and *Developing Telco Service Manager*.

CHAPTER 2

Basic Administration Tasks

In This Section

About the Basic Tasks

In order to help you, this chapter covers the basic tasks of administrating your TSM application.

These tasks include:

- Starting and stopping the Synchronizer connector
- Starting and stopping the OSS connector
- Starting and stopping the Approval Sequencer

You do not start and stop your TSM application. As the TSM is an application hosted by your application server, starting and stopping your TSM application is starting and stopping your application server. For more information, refer to your application server's documentation.

Starting and Stopping the Synchronizer Connector

The Synchronizer connector manages the communication between the CID and OSS systems. This agent manages the requests in the CID and their transmission to the OSS.

You use a set of administration tools to start and stop the Synchronizer connector. The administration tools are:

- `agentstart`
- `agentadm`

These administration tools are located in `<home_dir>/bin`.

For more information about configuring and using connectors, refer to *Developing Connectors*.

To start the Synchronizer connector

1. Go to `<home_dir>/bin`.
2. Run `agentstart`. Use the syntax:

   ```
   agentstart <connector_name>
   ```

   The connector is loaded and it starts the processes it needs.

To stop the Synchronizer connector

1. Go to `<home_dir>/bin`.
2. Run `agentadm`. Use the syntax:

   ```
   agentadm <host> <port> shutdown
   ```

   When finished, it displays a message.

Starting and Stopping the OSS Connector

The OSS connector manages the communication between the middleware and the OSS.

You use a set of administration tools to start and stop the OSS connector. The administration tools are:

- `ossstart`
- `ossadm`

These administration tools are located in `<home_dir>/bin`.

For more information about configuring and using connectors, refer to *Developing Connectors*.

To start the OSS connector

1. Go to `<home_dir>/bin`.
2. Run `ossstart`. Use the syntax:

   ```
   ossstart <connector_name>
   ```

   The connector is loaded and it starts the processes it needs.

To stop the OSS connector

1. Go to <home_dir>/bin.
2. Run `ossadm`. Use the syntax:

   ```
   ossadm <host> <port> <connector> shutdown
   ```

   When finished, it displays a message.

Starting and Stopping the Approval Sequencer

The Approval Sequencer manages the approval processes in the BLM.

You use a set of administration tools to start and stop the Approval Sequencer. The administration tools are:

- `approvalsequencer`
- `approvalsequenceradm`

These administration tools are located in `<home_dir>/bin`.

These tools use the `agent.properties` configuration file to set the properties of the Approval Sequencer. This file is located in `<home_dir>/config/approvalsequencer`.

For more information about working with the Approval Sequencer, refer to *Developing Telco Service Manager*.

To start the Approval Sequencer

1. Go to `<home_dir>/bin`.
2. Run `approvalsequencer`. The Approval Sequencer loads its configuration file and starts the processes it needs. When finished, it displays a message.

To stop the Approval Sequencer

1. Go to `<home_dir>/bin`.
2. Run `approvalsequenceradm`. Use the syntax:

   ```
   approvalsequenceradm <host> <port> shutdown
   ```

   When finished, it displays a message.

CHAPTER 3

Administrating TSM

In This Section

About Administrating TSM

When managing your TSM application, you may have to work with several different components and administration tools. For instance, you may have to monitor your TSM application, your database and other software, administrate the CID, manage message and notification queues, and check logs.

This section explains the different administration tasks and the tools you use to carry them out.

Administrating involves:

- Managing the Synchronizer
- Managing requests
- Managing trouble tickets
- Managing persistent action managers
- Reloading reference data
- Monitoring your application with the logger
- Auditing application activity using user events

This section does not show you how to install and configure the various administration tools. For more information on installing and configuring your tools, refer to *Installing Telco Service Manager* and the Appendixes in this manual.

Managing the Synchronizer Connector

The Synchronizer connector takes the requests from the CID and create messages to be sent to the OSS connector.

After getting a request from the CID, the Synchronizer connector translates the request into a message. This connector puts this message in a directory for the OSS connector.

Managing the Synchronizer connector involves:

- Checking the Synchronizer connector
- Purging the Synchronizer queue

Checking the Synchronizer Connector

You use the `agentadm` administration tool to manage the Synchronizer connector.

This administration tool is located in `<home_dir>/bin`.

You use this tool to:

- Check the connector status
- Check the Runtime mode
- Display the current settings

For more information about the Synchronizer connector, refer to *Developing Connectors*.

Synchronizer Connector Runtime Modes

This table describes the different running modes of the Synchronizer Connector.

Synchronizer Connector Modes

MODE	DESCRIPTION
Normal	The default mode in which the connector scans the CID at a regular, fixed time intervals, extracts the requests and sends them to the middleware backbone
Paused	The connector stops scanning the CID for requests, and waits for a restart or recover command before it resumes
Recover	The connector makes a final attempt to send requests (with the status 'TransportFailed') that have not been sent due to a transport problem. When finished, it goes back to the normal mode. The connector goes into this mode when all transport queues are down

To check the status

1. Go to `<home_dir>/bin`.
2. Run `agentadm`. Use the syntax:

   ```
   agentadm <host> <port> getstatus
   ```

   The administration tool displays the current status of the connector.

To check the runtime mode of the Agent

1. Go to `<home_dir>/bin`.
2. Run `agentadm`. Use the syntax:

   ```
   agentadm <host> <port> getmode
   ```

   The administration tool displays the current runtime mode of the connector.

To display the current settings

1. Go to <home_dir>/bin.
2. 2. Run agentadm. Use the syntax:

   ```
   agentadm <host> <port> info
   ```

   The administration tool displays the current settings of the connector.

Purging the Synchronizer Queue

You can empty the Synchronizer message queue when required.

Purging the Synchronizer queue also depends on your system's middleware and configuration. For instance, if your application uses MQSeries, you must use and follow the MQSeries procedure to purge queues.

By default, the Synchronizer uses a file queue for messages. The files that correspond to messages are put in a directory for processing by the OSS connector. This directory corresponds to the Synchronizer's `sync2oss` queue. To purge this queue, you delete the messages in this directory.

To purge the sync2oss queue

1. Go to `<home_dir>/<var>/data/sync2oss`.
2. Delete the message files.

If you use a middleware package to exchange messages with the OSS, refer to your product documentation for more information about purging messages.

Managing Requests

Managing requests involves:

- Viewing the pending requests
- Viewing a request detail
- Cancelling requests
- Resubmitting requests
- Purging requests

You use the administration tool to manage requests in the CID.

About Requests

When a user makes a change that impacts one of the legacy systems (such as add or remove service), the request is stored in the `REQUEST` table. The parameters of the request are stored in the `REQUEST_PARAM` table.

Requests can be:

- **Elementary**

  This type of request is a single request.
- **Composite**

  This type of request is a set of requests that have to be submitted for consistency.

When you change or cancel a composite request, you change all of the requests.

Pending requests are requests with one of the following statuses:

- `NOT YET SUBMITTED`
- `SUBMISSION IN PROGRESS`
- `TO BE APPROVED`
- `TRANSPORT FAILED`
- `ACKNOWLEDGED`
- `SUBMITTED`

This table describes the different request status codes and their names.

Request Status Codes

REQUEST STATUS CODE	REQUEST STATUS NAME
1	TO BE APPROVED
2	NOT YET SUBMITTED

REQUEST STATUS CODE	REQUEST STATUS NAME
3	SUBMITTED
4	DONE
5	FAILED
7	DENIED
8	SUBMISSION IN PROGRESS
10	TRANSPORT FAILED
12	ACKNOWLEDGED

Viewing the Requests

You can view a list of requests in the CID. You can also view the detail of a specific request.

To view the pending requests

1. Go to `<home_dir>/bin`.
2. Run the CID administration tool. Use the syntax:

   ```
   cidAdminTool admin_requests <CID> <CID_ADMIN login> <CID_ADMIN password>  [-quiet] [-datablocksize:####]
   ```

   where `<CID>`:

   - Oracle: `<instance alias>`
   - DB2: `<database alias>`
   - SQL Server: `<database host> [:<port>]` If no port is specified, the tool uses the default SQL server port

   The CID administration tool displays a menu.
3. Choose `1) View pending requests` then press Enter. The CID administration tool asks you to enter the number of days the requests have been pending.
4. Do one of the following:
   - Enter `0` to display all of the pending requests
   - Enter the number of days the requests have been pending
5. Press Enter. The CID administration tool displays a list of pending requests.

To view a request detail

1. Go to `<home_dir>/bin`.
2. Run the CID administration tool. Use the syntax:

```
cidAdminTool admin_requests <CID> <CID_ADMIN login> <CID_ADMIN
password>  [-quiet] [-datablocksize:####]
```

where <CID>:

- Oracle: <instance alias>
- DB2: <database alias>
- SQL Server: <database host> [:<port>] If no port is specified, the tool uses the default SQL server port

The CID administration tool displays a menu.

3 Choose 2) View a request in detail then press Enter. The CID administration tool asks you to enter the ID of the request to display.

4 Enter the ID of the request then press Enter. The CID administration tool displays the following pending request information:

- Request external id
- Generated by
- Creation date
- Action requested
- Status
- Status change date
- Reason for failure

Cancelling Requests

You can cancel specific requests or you can cancel all of the requests. Cancelling a request means you change its status to FAILED. To remove the request from the CID, you need to purge the FAILED requests.

To cancel a request

1 Go to <home_dir>/bin.

2 Run the CID administration tool. Use the syntax:

```
cidAdminTool admin_requests <CID> <CID_ADMIN login> <CID_ADMIN
password> [-quiet] [-datablocksize:####]
```

where <CID>:

- Oracle: <instance alias>
- DB2: <database alias>
- SQL Server: <database host> [:<port>] If no port is specified, the tool uses the default SQL server port

The CID administration tool displays a menu.

3 Choose `3) Set a pending request to failed` then press Enter. The CID administration tool asks you to enter the ID of the request to cancel by setting its status to FAILED.

4 Enter the ID of the request then press Enter.

5 At the prompt, press Y to confirm.

The CID administration tool cancels the request and displays a confirmation message.

To cancel all requests

1 Go to `<home_dir>/bin`.

2 Run the CID administration tool. Use the syntax:

```
cidAdminTool admin_requests <CID> <CID_ADMIN login> <CID_ADMIN password> [-quiet] [-datablocksize:####]
```

where `<CID>`:

- Oracle: `<instance alias>`
- DB2: `<database alias>`
- SQL Server: `<database host> [:<port>]` If no port is specified, the tool uses the default SQL server port

The CID administration tool displays a menu.

3 Choose `4) Set all pending requests to failed` then press Enter. The CID administration tool asks you to enter the number of days the requests have been pending that you want to set to `FAILED`.

4 Do one of the following:

- Enter `0` to set all of the pending requests
- Enter the number of days the requests have been pending

The CID administration tool displays a list of pending requests.

5 At the prompt, press Y to confirm.

The CID administration tool cancels the requests and displays a confirmation message.

Resubmitting Requests

You can resubmit specific requests. In general, you do not need to resubmit requests. This can happen when unforeseen problems or application failures occur. For instance, if the Synchronizer connector fails and restarts, you may have requests in the CID with the status `SUBMISSION IN PROGRESS`. You have to resubmit these request by changing their status to `NOT YET SUBMITTED`.

Only requests having the `REQUEST_STATUS_CODE` of `8 (SUBMISSION IN PROGRESS)` can be resubmitted.

To resubmit a request

1. Go to <home_dir>/bin.
2. Run the CID administration tool. Use the syntax:

   ```
   cidAdminTool admin_requests <CID> <CID_ADMIN login> <CID_ADMIN password> [-quiet] [-datablocksize:####]
   ```

   where `<CID>`:

   - Oracle: `<instance alias>`
   - DB2: `<database alias>`
   - SQL Server: `<database host> [:<port>]` If no port is specified, the tool uses the default SQL server port

   The CID administration tool displays a menu.
3. Choose `5) Reset a request being submitted to not yet submitted` then press Enter. The CID administration tool asks you to enter the ID of the request to resubmit.
4. Enter the ID of the request then press Enter.
5. At the prompt, press Y to confirm.
6. The CID administration tool changes the request status and displays a confirmation message.

Purging Requests

You can remove requests from the CID that are not pending requests. You can remove request having the following statuses:

- `TO BE APPROVED`
- `DONE`
- `FAILED`

To purge requests

1. Go to <home_dir>/bin.
2. Run the CID administration tool. Use the syntax:

   ```
   cidAdminTool purge_requests <CID> <CID_ADMIN login> <CID_ADMIN password> <days> [<status>,<status>,...] [-quiet] [-datablocksize:####]
   ```

   where <CID>:

   - Oracle: <instance alias>
   - DB2: <database alias>
   - SQL Server: <database host> [:<port>] If no port is specified, the tool uses the default SQL server port

   The CID administration tool removes all of the requests that:

   - Have the specified statuses
   - Are older than the specified number of days

   The CID administration tool displays a confirmation message.

Managing Trouble Tickets

Managing trouble tickets involves:

- Purging trouble tickets in the CID

You use the administration tool to manage trouble tickets in the CID.

About Trouble Tickets

Trouble tickets are located in the CID and have an associated status.

Purging Trouble Tickets

You can remove the trouble tickets from the CID.

To purge trouble tickets

1. Go to `<home_dir>/bin`.
2. Run the CID administration tool. Use the syntax:

   ```
   cidAdminTool purge_trouble_tickets <CID> <CID_ADMIN login>
   <CID_ADMIN password> <days> <statuscode> <datablocksize> [-
   quiet]
   ```

   where `<CID>`:

   - Oracle: `<instance alias>`
   - DB2: `<database alias>`
   - SQL Server: `<database host> [:<port>]` If no port is specified, the tool uses the default SQL server port

   The CID administration tool removes all of the trouble tickets that:

   - Have the specified statuses
   - Are older than the specified number of days

   The CID administration tool displays a confirmation message.

Managing Persistent Action Managers

Managing persistent action managers involves:

- Purging persistent action managers from the CID

You use the administration tool to manage persistent action managers in the CID.

About Persistent Action Managers

Shopping Carts correspond to BLM Action Managers. These action managers do exactly what their name implies, they manage the actions in the current context. An action manager is considered a shopping cart when it holds a set of actions to submit at the same time. The types of action managers are:

- Action Managers
- Persistent Action Managers

The context of your application determines which type of action manager you use. In general, you use Action Managers to manage the actions in a specific context or workflow. However these action managers and their contents cannot be saved. If your application requires saving the contents of an action manager, you use Persistent Action Managers. The saved Persistent Action Managers can be backup copies of the shopping cart or used as a shopping cart template.

When saving persistent action managers as Shopping Carts and templates, you should keep in mind the following:

- There is no limit to the number of saved persistent action managers
- There is no history of saved persistent action managers

Shopping Cart templates have the following:

- Name
- Description
- Category
- Additional information to be used as criteria to retrieve it

The shopping carts are saved in the same table as the backup shopping carts. This information is in the Shopping Cart Package tables in the CID:

- `PERSISTENT_ACTIONMGR_CATEGORY` Table
- `PERSISTENT_ACTIONMGR` Table

Persistent Action Manager Categories

This table describes the default Persistent Action Manager categories and their codes.

Persistent Action Manager Categories

CODE	NAME	NOTES
1	Backup	Backup category to save Persistent Action Managers during a session
2	Contract	For contract templates

Purging Persistent Action Managers

You can remove persistent action mangers from the CID. When purging the persistent action managers from the CID, you specify the category of persistent action managers to purge. Make sure you specify the correct category when purging the CID.

To purge persistent action managers

1. Go to `<home_dir>/bin`.
2. Run the CID administration tool. Use the syntax:

   ```
   cidAdminTool purge_persistent_action_managers <CID> <CID_ADMIN login> <CID_ADMIN password> <days> <category> [-quiet] [-datablocksize:####]
   ```

   where `<CID>`:

   - Oracle: `<instance alias>`
   - DB2: `<database alias>`
   - SQL Server: `<database host> [:<port>]` If no port is specified, the tool uses the default SQL server port

   The CID administration tool removes all of the persistent action managers that:

   - Are older than the specified number of days
   - Have the specified category

     The category corresponds to the value in the `PAM_CATEGORY_CODE` column of the `PERSISTENT_ACTIONMGR_CATEGORY` table.

   The CID administration tool displays a confirmation message.

Reloading Reference Data

In general, your reference data should not change frequently. But on occasion, you may need to change this data. You can use the reference data reload feature to program your application to reload reference data without having to take your application off line.

You cannot use this feature to remove reference data and it cannot be used to add new types of reference data. This feature is for reloading modified and new reference information only. If you modify the data structure or remove reference information, you must stop and restart your application server.

Reloading reference data involves:

- Specifying the caching policy of BLM objects
- Creating the batch files to reload the data
- Programming JSPs to use this feature

About the BLM Cache

The BLM cache is a global cache for all user sessions. All of the cached BLM objects are one of following types:

- `GLOBAL`

  The object is cached the entire life of the BLM host process.

- `RELOADABLE`

  The object is cached and can be updated using the reference data reload feature.

- `HTTP_REQUEST`

  The object is cached and updated for each HTTP request.

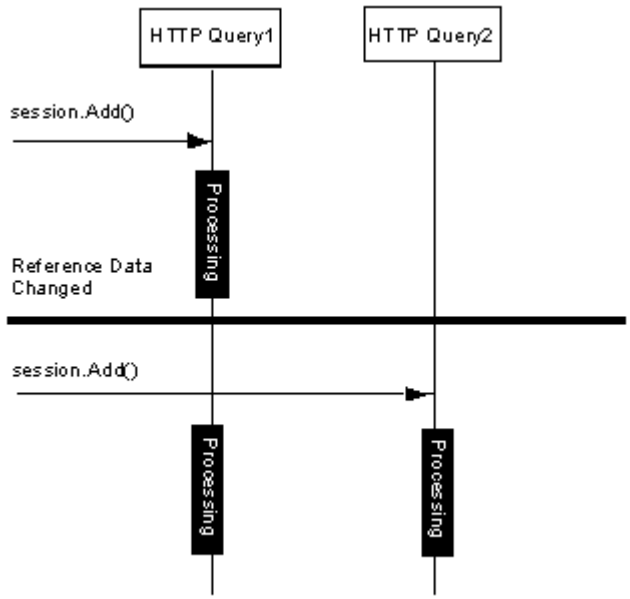
The `policy.properties` configuration file contains the list of BLM objects and their assigned type. This file is located in `<home_dir>/classes/nmycfg/blm/util`.

The `VERSIONS` table contains information about the reference data. This table contains the version of the reference data and its associated timestamp.

Every time a JSP calls the `session.Add()` method, the BLM checks the `VERSIONS` table to see if the timestamp has changed. If the reference data has changed, the BLM updates all of the `RELOADABLE` objects in the cache and starts the session.

Limits to Using the Reference Data Reload Feature

Concurrent user sessions may cause some problems and you should design your application to take the following sequence of events:



In this sequence of events, the HTTP Query2 causes your TSM application to update the BLM cache because the reference data changed. In this case, the HTTP Query 1 cannot be notified and the BLM cache might not contain data needed by the query. This is especially important if your data reload contains modified reference information. This situation does not cause problems when you reload only new reference information, because the threads do not need to know about new information.

In order to avoid this situation, you can:

- Use this feature only to load modified or new information in existing reference tables
- Block access to the impacted service during reloads (redirect to another JSP, display a message)
- Limit the frequency of reloads
- Reload your data during off-peak hours

1 Create a batch file to extract and load information into the CID.

2 At the end of the transaction, do the following in the `VERSIONS` table:

- Insert your reference data version reference
- Insert the timestamp using the `SYSDATE` method

```
update versions set ITEM_VERSION="your reference" where
ITEM_CODE='REFERENCE_DATA'
```

```
update versions set ITEM_TIMESTAMP=sysdate where
ITEM_CODE='REFERENCE_DATA'
```

Do not insert or change the information in the `STRUCTURE` data. This data is for internal use only and must not be modified.

To program a JSP using the reference data reload feature

1. Open and validate the session
2. Use the `ObjectRefMgr.getReferenceDataTimestamp()` method to return the time stamp of the current reference data.
3. Store the timestamp in the `HTTP` session.
4. When the thread handles an `HTTP` request within the current session, retrieve the timestamp and compare it with the stored timestamp.
   - If the timestamps differ, the reference data has changed and the BLM cache has been updated. You can change the workflow (display a message, reset the workflow, and so on).
   - If the timestamps are identical, the application continues normally.

For more information about programming JSPs, refer to *Building User Interfaces*.

Monitoring TSM

You can use the system logger to create logs while your TSM application is running. These logs are very helpful when you have to monitor system activity and are helpful in solving problems and pinpointing system or architecture bottlenecks.

The common logger features are available for the different product components and each component can have its own specific logger configuration and output. You can log events that occur in the:

- Presentation Layer - includes the Personalization Manager and CSS Engine
- Synchronizer - The Synchronizer connector, SmartLink Framework, CSS Engine
- Approval Sequencer
- CID2CBU loader
- Connectors - Loopback and Template Connectors

About Monitoring Applications using Logs

Using logs to monitor your application involves the following:

- The capability to define meaningful alerts based on logs the application generates.
- The assumption that your application works fine until an alert has been raised.
- Your TSM application does not only deal with core components.

This also means that you use a supervision platform or tool to gather information and generate alerts. The enhanced system logger helps you make your TSM application an integral part of a comprehensive monitoring system.

There are also different professionals who supervise applications and who can use these logs to monitor applications:

- **Developers**

  They can define what should be logged and monitored. They also determine the type of alert to generate and define possible corrective actions.

- **Administrators**

  They are responsible for monitoring the application and define and integrate system logs and the monitoring system.

For more information about the standard log messages and their meaning, refer to the *Logger Message Reference Documentation*.

Logger Events

ATTRIBUTE	DESCRIPTION
Date/Time	ISO-8601 date string to identify the time
Thread id	To identify the thread that hosts the event generator
Unique id	To identify the event
Session id	To identify the user session
Type	To classify the impact of event
Severity	To classify the issue level
Module	To identify the event technical source
Code	To classify and describe what occurs
Description	To classify and describe what occurs – linked to code
Debug information	To provide additional technical information

Event Types

EVENT TYPE	DESCRIPTION
INIT	Covers application/module initialization processes
STATE	Covers application/module state changes
EXCEPTION	Covers internal exceptions
SESSION	Covers user session life cycle
REQUEST	Covers request life cycles
MESSAGE	Covers lifecycle in the integration framework
OBJECT	Covers object handling
RESOURCE	Covers component events
DATA	Covers customer data handling in the CID
NONE	Unclassified events

Severity Levels

SEVERITY LEVEL	DESCRIPTION
FATAL	Events that have an impact on the availability of the application
ERROR	Events that cause a given application workflow to not work properly
WARN	Events that may impact the behavior of the application

SEVERITY LEVEL	DESCRIPTION
INFO	Events that record successful basic system actions for supervision
DEBUG	Level 0: no debug information. Level 3: for activation in a production environment within a working day. Level 5: for activation in a production environment for a limited period. Level 7: for activation in a very limited way with one or two concurrent users.

Event Modules

EVENT MODULE	DESCRIPTION
AGT	Any agent – synchronizer, connector, sequencer
BLM	Business Logic Manager
DAL	Data Access Layer
SmartLink Framework	SmartLink Framework and all of its sub modules
JSPF	JSP Framework
JSP	Java Server Pages
LOG	The logger platform
NIL	No module involved
UTL	Internal utility components

Event Code/description:

- Events are coded to ensure the description, for a given case, is always the same whatever the event source is.
- A code is associated with a description, which is the actual event message.
- The code is the reference to detect what occurs.

Overview of Using Logs

When using logs to monitor your application, you have to decide on the following:

- The goal for logs
- Which components to monitor
- The need for dynamic configuration
- The information to generate
- The output format

Determining the Goals

You have to determine what exactly you want to do with the logs. Once you decide this, you can easily determine what needs to be done in order to configure and set up the logger.

- If you work with a supervision platform that monitors a text-based source, you should use the circular file (rolling file) driver. We recommend this because even though there are backups, logs are always generated into the main output, with a constant file name.
- If you want to backup and route the generated logs regularly, you should use the time stamped file (daily file) driver. There are several backup capabilities.
- If you want to check what occurs in the application, you should use the standard output.

Determining the Components to Monitor

You can get events from several product layers:

- Presentation Layer

  This layer covers modules that run in the application server.
- Integration Layer
- System components

The components to monitor depend on your application features and architecture. For instance, if your application does not use the order validation feature, you do not use the Approval sequencer. Therefore you do not need to worry about monitoring its logs. The same is true with connectors. If you do not synchronize your backend systems with a connector, there is no need to work with connector logs.

Determining the Need for Dynamic Configuration

You can configure the logger dynamically. This means you can change the logger configuration without restarting the component. Using this feature involves a thread that reloads the logger configuration regularly.

You may need to use this feature in the following environments:

- Production environment

  In this environment, activating the dynamic reload is trade-off between the need to increase the amount of logged information when an issue occurs and having an additional thread running. This also depends on the supervision process and if it covers the capability to change the logger configuration and get more information.

- Application acceptance test / Pre-integration environment

  In these environments, dynamic configuration is helpful to speed up tasks. As issues may occur frequently, using this feature helps get the additional information you need to pinpoint a problem and find a solution quickly.

- Development

  When developing, dynamic configuration is helpful to continuously adapt the log to the components being worked on or tested.

Default Settings:

- The presentation layer logger does not use dynamic configuration.
- Standalone components (synchronizer, connector, approval sequencer) use dynamic configuration with a configuration reload every 5 minutes.

Determining the Level of Information

In order to obtain information you can use, you need to determine the type of information to log.

You use the following event attributes to make sure the logs contain the information you need:

- Event Type

  Basically, you could decide to log all event types, but it really depends on your supervision process.

  For instance, you may not want to focus on init/stop phases. In that case, there is no need to get 'INIT'-types events.

- Event Severity

  The logger configuration offers capability to define the minimum severity level you want to log.

  If you select 'ERROR', the log only contains events with severity 'ERROR' or 'FATAL'.

  For instance, you log for a monitoring platform and are trying to pinpoint some problems. In this case, you set the minimum severity level to 'WARN'.

Determining the Output Format

The format of your output depends of the way you plan on using the logs. For instance, you most likely use different output formats when working with a system console and a supervision platform.

When determining your format, keep in mind:

- The logger generates one line per event
- You can configure the pattern code
- You can arrange attributes the way you want and you can use any allowed character separator
- You can decide to log all attributes or not.

Configuring the Logger

You can customize the logger for each component. For example, your presentation layer logger can log only logins and critical errors and your Synchronizer logger tracks debug information.

For each component, you use the following configuration files:

- `logger.properties` to set the basic logger properties
- `log4j.properties` to set the dynamic configuration parameters

The location of the configuration files:

COMPONENT	PATH
Presentation Layer	`<home_dir>/classes/nmycfg/util` `<home_dir>/<channels>/WEB-INF/classes/nmycfg/util`
Synchronizer	`<home_dir>/config/synchronizers/synchronizer/util`
Approval sequencer	`<home_dir>/config/approvalsequencer/util`
Connector Template	`<home_dir>/config/connectors/loopback/nmycfg/util`
Loopback Connector	`<home_dir>/config/connectors/connectortemplate/nmycfg/util`

Although each component has its own set of configuration files, you configure the components the same way. The instructions in this section apply to configuring the logger for all components.

Setting the Basic Properties

You use the `logger.properties` configuration file to set the following basic properties of each logger:

- Location of the dynamic configuration file

  Path of the `log4j.properties` configuration file that contains information on the events to log and their format.

- Frequency to reload the parameter file

  The time in seconds to reload the `log4j.properties` configuration file.

To set the path of the parameter file

1. Go to the `/util` directory containing the `logger.properties` configuration file.
2. Open `logger.properties`.
3. For the `logger.log4j.properties` setting, enter the name of the `log4j.properties` configuration file.
4. Save your changes.

The two files must be in the same path.
The `log4j.properties` configuration file is found using the `CLASSPATH`.

To set the reload frequency

1. Go to the `/util` directory containing the `logger.properties` configuration file.
2. Open `logger.properties`.
3. Change the `logger.log4j.reloadDelay` setting to one of the following:
   - The number of seconds to wait before reloading the `log4j.properties configuration file`
   - 0 to deactivate reloading
4. Save your changes.

By default, the logger configuration disables the dynamic reloading of the configuration for Presentation Layer.
This is due to J2EE specifications that recommend not creating threads, except when explicitly required.

Specifying the Events to Log

You use the `log4j.properties` configuration file to set the types of events to log.

By specifying different types of events to log, the logs contain only information about the different types of events you want to track.

For each event type, you specify:

- The minimum severity level
- Debug level when required
- The target for the generated events (output)

To specify the types of events to log

1. Go to the `/util` directory containing the `log4j.properties` configuration file.
2. Open `log4j.properties`.
3. Under `EVENTS`, enter the event to log. Use the syntax:

```
<Event Type> = <Severity Level>, [Debug Level], <Output Driver #1>, <Output Driver #2>,...
```

`<Event Type>` is one of the following event type patterns:

EVENT TYPE	DESCRIPTION
All types	log4j.category.nmy
INIT	log4j.category.nmy.INIT
STATE	log4j.category.nmy.STATE
EXCEPTION	log4j.category.nmy.EXCEPTION
SESSION	log4j.category.nmy.SESSION
REQUEST	log4j.category.nmy.REQUEST
MESSAGE	log4j.category.nmy.MESSAGE
OBJECT	log4j.category.nmy.OBJECT
RESOURCE	log4j.category.nmy.RESOURCE
DATA	log4j.category.nmy.DATA
NONE	log4j.category.nmy.NONE

`<Severity Level>` is the minimum security level you want the event type to be logged with:

SEVERITY LEVEL	LEVELS OF EVENTS LOGGED
FATAL	FATAL

SEVERITY LEVEL	LEVELS OF EVENTS LOGGED
ERROR	FATAL, ERROR
WARN	FATAL, ERROR, WARN
INFO	FATAL, ERROR, WARN, INFO
DEBUG	FATAL, ERROR, WARN, INFO, DEBUG

When the `<Severity Level>` is DEBUG, you can enter the `[Debug Level]`:

NAME	NOTES
0	Minimum debug information
3	Events with debug level 3
5	Events with debug level 3 or 5
7	Events with debug level 3, 5, or 7

`<Output Driver>` is the output driver to use:

DRIVER	CODE
Standard output driver	CON
Rolling file output driver	ROL
Daily file output driver	DAY

You enter as many lines in the `EVENTS` section as there are event types to track. To disable an event type, enter a line `<Event Type>=INFO,DISABLED`

Configuring the Logger Output

You use the `log4j.properties` configuration file to set the properties of the output drivers.

For each output driver, you specify:

- Target of the driver
- Driver specific properties
- Format of the log

About the Standard Output Driver

This output driver sends log information to your application's standard output (stdout.) This log information is included in all other standard application output.

To configure a standard output driver

1. Go to the `/util` directory containing the `log4j.properties` configuration file.
2. Open `log4j.properties`.
3. Go to the `#STANDARD OUTPUT DRIVER SETTINGS` section.
4. Set `log4j.appender.CON.Target` to one of the following:
   - `System.out` to redirect output to the standard output
   - `System.error` to redirect output to the error output.
5. Set the `log4j.appender.CON.layout.ConversionPattern` to specify the log format.
6. Save your changes.

```
log4j.appender.CON=org.apache.log4j.ConsoleAppender
log4j.appender.CON.Target=System.out
log4j.appender.CON.layout=com.netonomy.util.logger.LoggerLayout
log4j.appender.CON.layout.ConversionPattern=+{DATE_TIME};{EVENT_ID};{SEVERITY};{MODULE};{THREAD_ID};{EVENT_TYPE
};{ERROR_CODE};{DESCRIPTION}
```

About the Rolling File Output Driver

This output driver saves log information in a text file.

When using this driver, you specify the following:

- The name and location of the log file
- Maximum size of the text file.

  When this limit is reached, the logger saves the file as `<log_filename>.1` and starts a new log file. When there is more than one saved logger file, the logger increments the filename of the saved log files. This means that the `<log_filename>.1` is the latest saved log file.
- Number of backup copies.

  Determines the number of backup copies the logger keeps. When this limit is reached, the logger deletes the oldest backup copy.

Make sure you specify the correct file size and the number of backup copies.

If you do not, you may lose some log information as the logger automatically deletes the oldest log files.

To configure a rolling file output driver

1. Go to the `/util` directory containing the `log4j.properties` configuration file.
2. Open `log4j.properties`.

3 Go to the `#ROLLING FILE OUTPUT DRIVER SETTINGS` section.

4 Modify the following settings:

SETTING	DESCRIPTION
`log4j.appender.ROL.File`	Full path and filename of the log file
`log4j.appender.ROL.MaxFileSize`	Maximum size of the log file. Use the syntax: <size><unit> Units can be either: KB MB GB For a log file with a maximum size of 4MB, enter: log4j.appender.ROL.MaxFileSize=4MB

5 Save your changes.

Saved copies are named `<log_filename>.<number>` and incremented when the logger creates a new copy.
This means that `<log_filename>.1` is the latest saved log file.

Default Rolling File Output Settings

By default, the Rolling file output driver has the following configuration:

- MaxFileSize: 4 MB
- MaxBackupIndex: 5
- File: depends on the product layer

COMPONENT	ROLLING FILE DRIVER OUTPUT FILE DEFAULT LOCATION
Presentation Layer	`<home_dir>/var/logs/nmy_application.log`
Synchronizer	`<home_dir>/var/logs/nmy_synchronizer.log`
Connector Template	`<home_dir>/var/logs/nmy_connectortemplate.log`
Loopback Connector	`<home_dir>/var/logs/nmy_loopback.log`
Approval Sequencer	`<home_dir>/var/logs/nmy_sequencer.log`

Example of Rolling File Output Driver Settings

In this example, the logger:

- Saves logs in the `nmy_application.log` file
- Keeps a maximum number of 5 backup copies:
  - `nmy_application.log.1`

- nmy_application.log.2
- nmy_application.log.3
- nmy_application.log.4
- nmy_application.log.5

- Has a maximum file size of 4MB

```
log4j.appender.ROL=org.apache.log4j.RollingFileAppender
log4j.appender.ROL.File=!NMY_VAR_DIR!/logs/nmy_application.log
log4j.appender.ROL.MaxFileSize=4MB
log4j.appender.ROL.MaxBackupIndex=5
log4j.appender.ROL.layout=com.netonomy.util.logger.LoggerLayout
log4j.appender.ROL.layout.ConversionPattern=+{DATE_TIME};{EVENT_ID};{SEVERITY};{MODULE};{THREAD_ID};{EVENT_TYPE
};{ERROR_CODE};{DESCRIPTION}
```

About the Time Stamp Log File Output Driver

This output driver saves log information in a text file.

When using this driver, you specify the following:

- The name and location of the log file
- The interval before creating a backup file. When this interval occurs, the logger saves the file as <log_filename>.<date> and starts a new log file.

There is no limit to the number of backup copies of the log file.
You must manage the backup log files, as the logger does not automatically delete them.

To configure a time stamp log file driver

1. Go to the /util directory containing the log4j.properties configuration file.
2. Open log4j.properties.
3. Go to the #DAILY FILE OUTPUT DRIVER SETTINGS section.
4. Set log4j.appender.DAY.File to the full path and log file name.
5. Set log4j.appender.DAY.DatePattern to one of the following:

SETTING	DESCRIPTION	BACKUP FILE EXTENSION
'.'yyyy-MM	Monthly logs that begin at the start of the month	<log_filename>.YYYY-MM
'.'yyyy-ww	Weekly logs that start at the beginning of each week	<log_filename>.YYYY-WW
'.'yyyy-MM-dd	Daily logs that start at midnight	<log_filename>.YYYY-MM-DD
'.'yyyy-MM-dd-h	Daily logs that start at noon and midnight	<log_filename>.YYYY.MM-DD-AM/PM

SETTING	DESCRIPTION	BACKUP FILE EXTENSION
'.'yyyy-MM-dd-HH	Hourly logs that start on the hour	<log_filename>.YYYY-MM-DD-HH
'.'yyyy-MM-dd-HH-mm	Minute logs that start every minute	<log_filename>.YYYY-MM-DD-HH-MM

6 Set the `log4j.appender.CON.layout.ConversionPattern` to specify the log format.

7 Save your changes.

Default Time Stamp Log File Output Settings

By default, the Time Stamp Log File output driver has the following configuration:

- Backup every day at midnight - `DatePattern='.'yyyy-MM`
- File: depends on the product layer

COMPONENT	ROLLING FILE DRIVER OUTPUT FILE DEFAULT LOCATION
Presentation Layer	<home_dir>/var/logs/nmy_daily_application.log
Synchronizer	<home_dir>/var/logs/nmy_daily_synchronizer.log
Connector Template	<home_dir>/var/logs/nmy_daily_connectortemplate.log
Loopback Connector	<home_dir>/var/logs/nmy_daily_loopback.log
Approval Sequencer	<home_dir>/var/logs/nmy_daily_sequencer.log

In this example, the logger:

- Saves logs in the `nmy_daily_application.log` file
- Daily log that starts at midnight

```
log4j.appender.DAY=org.apache.log4j.DailyRollingFileAppender
log4j.appender.DAY.File=!NMY_VAR_DIR!/logs/nmy_daily_application.log
log4j.appender.DAY.DatePattern='.'yyyy-MM-dd
log4j.appender.DAY.layout=com.netonomy.util.logger.LoggerLayout
log4j.appender.DAY.layout.ConversionPattern=+{DATE_TIME};{EVENT_ID};{SEVERITY};{MODULE};{THREAD_ID};{EVENT_TYPE
};{ERROR_CODE};{DESCRIPTION}
```

About the Output Format

For each driver, you configure the output format. The output format is an arrangement of the event attributes that generates a line in the output.

Each attribute is assigned a pattern you use to define the layout. This means that the format is defined with a string that is a suite of patterns and separators.

Use the syntax:

```
<Driver ConversionPattern setting> = separator
[EVENT_ATTRIBUTE_PATTERN_1] separator
[EVENT_ATTRIBUTE_PATTERN_2]…
```

Each of the format definition elements is optional. You can log the event attributes you want.

You cannot use the following characters as layout separators:

- ! (exclamation point)
- : (colon)
- - (dash)

Use the syntax:

```
log4j.appender.<OUTPUT_DRIVER_CODE>.layout.ConversionPattern=+
{EVENT_OUTPUT_PATTERN1} separator {EVENT_OUTPUT_PATTERN2}...
```

For more information about the attribute patterns, refer to *Output Patterns* in this chapter.

The logger inserts the output as a single line of text in the log file.
If you log DEBUG information, the logger places a block of information in the log between the `<DEBUG_INFO>` and `</DEBUG_INFO>` tags.
Because of this, you should place DEBUG information at the end of your output format. This keeps all of the log information together then lists any associated debug information. Otherwise you may have log information split by a block of debug information.

To specify the output format

1. Go to the `/util` directory containing the `log4j.properties` configuration file.
2. Open `log4j.properties`.

3 Under the `OUTPUT DRIVER` section, set `log4j.appender.<OUTPUT_DRIVER_CODE>.ConversionPattern` to the format to use. Use the output format syntax.

4 Save your changes.

EVENT ATTRIBUTE PATTERN	DESCRIPTION	NOTES
{DATE_TIME}	Event date and time	The format complies with ISO 8601 standard
{SEVERITY}	Event severity level	FATAL ERROR WARN INFO DEBUG
{THREAD_ID}	Event thread ID	
{SESSION_ID}	Session ID	Session ID of the user. This is used to identify the user who caused the event to be logged. This ID is the BLM session ID. When there is no BLM session, the value is NONE.
{EVENT_TYPE}	Event type	This is the type of event you can specify
{MODULE}	Event source module	AGT - Any agent – synchronizer, connector, sequencer, CID2CBU loader BLM - Business Logic Manager CUS - Custom event DAL - Data Access Layer SmartLink Framework - SmartLink Framework and all of its sub-modules JSPF - JSP Framework JSP - Java Server Pages LOG - The logger platform NIL - Unqualified event UTL - Internal utility components
{EVENT_ID}	Event ID	Unique ID for the specific output driver
{ERROR_CODE}	Event error code	
{DESCRIPTION}	Event description	

EVENT ATTRIBUTE PATTERN	DESCRIPTION	NOTES
{DEBUG_INFO}	Event Debug info	Creates a block of information between the <DEBUG_INFO> and </DEBUG_INFO> tags. This should be used at the end of your debug pattern

Default Format Settings

For product layers and drivers, the default output format is:

```
+{DATE_TIME};{EVENT_ID};{SEVERITY};{MODULE};{THREAD_ID};{EVENT_T
YPE};{ERROR_CODE};{DESCRIPTION}
```

For the Presentation Layer, the default output format is:

```
+{DATE_TIME};{EVENT_ID};{SESSION_ID};{SEVERITY};{MODULE};{THREAD
_ID};{EVENT_TYPE};{ERROR_CODE};{DESCRIPTION}
```

Examples of Logger Files

The following examples show the logger configuration files and an example of a logger message.

Example of `logger.properties`

Location of the parameter file	`logger.log4j.properties=log4j.properties`
Frequency of parameter file reload	`logger.log4j.reloadDelay=0`

Example of `log4j.properties`

Internal settings
DO NOT MODIFY
	`# DO NOT MODIFY`
	`log4j.categoryFactory=com.netonomy.util.logger.LoggerCategoryFactory`
	`log4j.rootCategory=INFO,DISABLED`

Types of events to log
All of the events are logged in the ROL log file
Exceptions are logged in a DAY log file
Unqualified are displayed in the system console
	`# EVENTS`
	`log4j.category.nmy.INIT=INFO,0,ROL`
	`log4j.category.nmy.STATE=INFO,0,ROL`
	`log4j.category.nmy.EXCEPTION=INFO,0,ROL, DAY`
	`log4j.category.nmy.SESSION=INFO,0,ROL`
	`log4j.category.nmy.REQUEST=INFO,0,ROL`
	`log4j.category.nmy.MESSAGE=INFO,0,ROL`
	`log4j.category.nmy.OBJECT=INFO,0,ROL`
	`log4j.category.nmy.RESOURCE=INFO,0,ROL`
	`log4j.category.nmy.DATA=INFO,0,ROL`
	`log4j.category.nmy.NONE=INFO,0,ROL,CON`

Internal settings DO NOT MODIFY	# DO NOT MODIFY log4j.appender.DISABLED=com.netonomy.util.logger.NullAppender
Standard Output Settings	# STANDARD OUTPUT DRIVER SETTINGS log4j.appender.CON=org.apache.log4j.ConsoleAppender log4j.appender.CON.Target=System.out log4j.appender.CON.layout=com.netonomy.util.logger.LoggerLayout log4j.appender.CON.layout.ConversionPattern=+{DATE_TIME};{EVENT_ID};{SESSIO N_ID};{SEVERITY};{MODULE};{THREAD_ID};{EVENT_TYPE};{ERROR_CODE};{DESCRIPTIO N}
Rolling FIle Output Settings	# ROLLING FILE OUTPUT DRIVER SETTINGS log4j.appender.ROL=org.apache.log4j.RollingFileAppender log4j.appender.ROL.File=!NMY_VAR_DIR!/logs/nmy.log log4j.appender.ROL.MaxFileSize=4MB log4j.appender.ROL.MaxBackupIndex=5 log4j.appender.ROL.layout=com.netonomy.util.logger.LoggerLayout log4j.appender.ROL.layout.ConversionPattern=+{DATE_TIME};{EVENT_ID};{SESSIO N_ID};{SEVERITY};{MODULE};{THREAD_ID};{EVENT_TYPE};{ERROR_CODE};{DESCRIPTIO N}
Daily File Output Settings	# DAILY FILE OUTPUT DRIVER SETTINGS log4j.appender.DAY=org.apache.log4j.DailyRollingFileAppender log4j.appender.DAY.File=!NMY_VAR_DIR!/logs/nmy_daily_log.log log4j.appender.DAY.DatePattern='.'yyyy-MM-dd log4j.appender.DAY.layout=com.netonomy.util.logger.LoggerLayout log4j.appender.DAY.layout.ConversionPattern=+{DATE_TIME};{EVENT_ID};{SESSIO N_ID};{SEVERITY};{MODULE};{THREAD_ID};{EVENT_TYPE};{ERROR_CODE};{DESCRIPTIO N}

This example shows the initialization log.

```
+2002-03-06 17:39:23.830;EC6C8259B6ExecuteThread-310;INFO;LOG;ExecuteThread-31;INIT;0012000;Setting logger
configuration file to [path="classes/nmycfg/util/log4j.properties"].


+2002-03-06 17:39:23.850;EC6C8259B6ExecuteThread-311;INFO;LOG;ExecuteThread-31;INIT;0012001;Set dynamic reload
of logger configuration every [frequency=30] seconds.


+2002-03-06 17:39:29.889;EC6C8259B6ExecuteThread-312;INFO;DAL;ExecuteThread-31;INIT;0011001;Initializing
datasource [name="cidDatasource"], [driver="JNDI t3://localhost:7001"], [user="N/A"] succeeded.


+2002-03-06 17:39:52.232;EC6C8259B6ExecuteThread-313;INFO;DAL;ExecuteThread-31;INIT;0010101;Loading component
configuration [name="DAL"] succeeded.


+2002-03-06 17:40:18.760;EC6C8259B6ExecuteThread-314;INFO;BLM;ExecuteThread-31;INIT;0010101;Loading component
configuration [name="BLM"] succeeded.


+2002-03-06 17:40:21.635;EC6C8259B6ExecuteThread-315;INFO;JFN;ExecuteThread-31;INIT;0040001;Loading media
application file [path="<home_dir>/channels/WEB-INF/classes/nmycfg/jfn/MyWeb.xml"] succeeded.


+2002-03-06 17:40:22.255;EC6C8259B6ExecuteThread-316;WARN;BLM;ExecuteThread-31;STATE;0022102;Checking the
validity of the reference data cache. Cache out of date and cleared.
```

This example shows the log when an error occurs when loading a required file.

```
+2003-01-20 18:18:35.261;F2DC8E7ABEExecuteThread: '12' for queue: 'default'63;1060;INFO;BLM;ExecuteThread:
'12' for queue: 'default';INIT;0010101;Loading component configuration [name="nmycfg.blm.config"] succeeded.

+2003-01-20 18:21:21.568;F2DC8E7ABEExecuteThread: '12' for queue: 'default'64;1060;ERROR;DAL;ExecuteThread:
'12' for queue: 'default';DATA;0011404;Executing SQL statement failed [execute type="update"]: [DB error="1"].

+2003-01-20 18:25:40.728;F2DC9FB90CExecuteThread: '12' for queue: 'default'0;NONE;INFO;LOG;ExecuteThread: '12'
for queue: 'default';INIT;0012000;Setting logger configuration file to [path="nmycfg/util/log4j.properties"].

+2003-01-20 18:25:40.768;F2DC9FB90CExecuteThread: '12' for queue: 'default'1;NONE;INFO;LOG;ExecuteThread: '12'
for queue: 'default';INIT;0012002;Dynamic reload of logger configuration is not activated.

+2003-01-20 18:25:48.429;F2DC9FB90CExecuteThread: '12' for queue: 'default'2;NONE;FATAL;DAL;ExecuteThread:
'12' for queue: 'default';INIT;0011002;Initializing datasource [name="jdbc/cidDatasource"], [driver="JNDI "],
[user="N/A"] failed.

+2003-01-20 18:25:48.650;F2DC9FB90CExecuteThread: '12' for queue: 'default'3;NONE;FATAL;DAL;ExecuteThread:
'12' for queue: 'default';INIT;0030105;Instantiating Data Access Layer driver
[instance="nmycfg.dal.instances.instance_route"],
[driver="com.netonomy.dal.drivers.impl.sql.jndi.JNDIDatasourceInstance"] failed.

+2003-01-20 18:25:48.650;F2DC9FB90CExecuteThread: '12' for queue: 'default'4;NONE;FATAL;DAL;ExecuteThread:
'12' for queue: 'default';INIT;0010002;Initializing component [name="DAL Authentication Module"] failed.
```

Using Logs with a Supervision Platform

When using a supervision platform to manage or analyze logs, you can do the following to maximize the performance and quality of information in the logs:

- Adapt the output format for your platform
- Use certain event attributes
- Take advantage of event codes

Adapting the Output Format

Most supervision platforms offer the possibility to parse text files with pattern matching.

This feature is based on the following:

- In the generated file, there is one event per line
- The line structure is known

You can adapt the output format of log files to make pattern matching easier.

As this is also based on the principle that there is one event per line, you should not log `DEBUG_INFO`.

The default output format settings follow these recommendations.

Using Event Attributes

Some event attributes are very helpful to supervise:

- Event Id

  Because the ID is unique, you can use this information as a reference when the alert is to be forwarded to tracking systems.

- Session Id

  This ID allows you to track events which are generated by the same user. This way you can pinpoint and follow the events that are logged. If there is no associated user session id, you can still track the sequence of logged events by using the thread id.

- Event codes

Using Event Codes

When using event codes, the main attribute to check is severity. This is because severity gives you an idea as to the impact of the event on the application. But the real impact depends on your application. As most events are coded, you can use this to your advantage when checking logs.

For event codes:

- Each code is associated with a description (the event description attribute value).
- In some descriptions, there are some fixed parts (the description parameters).

Description parameters refer to the dynamic part of the description. But they are always marked with a fixed and documented tag. This part of the event code is the code interface and will not change even if the description does.

An example:

Your deployment relies on a file system that is extremely sensitive and you want to monitor events that deal with your application interacting with this file system.

There are some event codes that are dedicated to file system interaction.

For instance, the code 2004 is associated with the event description "`Cannot write to file [path="<the path>"].`"

In this case, the section `[path=""]` is fixed.

This means you can:

- Monitor logging of event code '2004'.
- If the event occurs, check the 'path' description parameter.

If the path that appears just after the `<[path=">` tag, you can generate a critical alert for your supervision system and have people work on the issue quickly.

For more information about the standard codes, refer to the *Logger Message Reference Documentation*.

Filtering by Source Component

You can also configure the logger to filter events by component. You can use this feature to help application development and debugging as you focus on event source components.

If you have customized a part of the application and want to check logs for testing or debugging, you can focus on the component you work with, activate full debug information, without getting a huge amount of events from components you are not interested in.

You filter events by specifying event types. In addition to event types, you can filter logged events by using the event component parameter.

You use the syntax:

```
<Event Type>.<Event Source> = <Severity Level>, [Debug Level],
<Output Driver #1>, <Output Driver #2>,...
```

For example, `log4j.category.nmy.INIT.BLM=INFO,0,ROL`

- Reduces the logs for INIT-type events to the ones issued from the BLM.
- Logs events using the Rolling File output, and there is no DEBUG event generated.

The event modules you can filter on are:

EVENT COMPONENT	DESCRIPTION
ADM	Administration
AGT	Any agent – synchronizer, connector, sequencer, CID2CBU loader
BLM	Business Logic Manager
CUS	Custom event
DAL	Data Access Layer
SmartLink Framework	SmartLink Framework and all of its sub-modules
JSPF	JSP Framework
JSP	Java Server Pages
LOG	The logger platform
NIL	Unqualified event
QRA	Query and Report Analysis
UTL	Internal utility components
WFS	Web File System

Examples of Filtering by Component

1 Get all types of event but only ones generated by the BLM – no debug information, rolling file

```
# EVENTS
log4j.category.nmy.INIT.BLM=INFO,0,ROL
log4j.category.nmy.STATE.BLM=INFO,0,ROL
log4j.category.nmy.EXCEPTION.BLM=INFO,0,ROL
log4j.category.nmy.SESSION.BLM=INFO,0,ROL
log4j.category.nmy.REQUEST.BLM=INFO,0,ROL
log4j.category.nmy.MESSAGE.BLM=INFO,0,ROL
log4j.category.nmy.OBJECT.BLM=INFO,0,ROL
log4j.category.nmy.DATA.BLM=INFO,0,ROL
log4j.category.nmy.NONE.BLM=INFO,0,ROL
...
```

2 Get only 'INIT' events, but block the ones generated by the DAL – debug level 3, rolling file

```
# EVENTS
log4j.category.nmy.INIT.BLM=INFO,0,ROL
log4j.category.nmy.INIT.JFN=INFO,0,ROL
log4j.category.nmy.INIT.JSP=INFO,0,ROL
log4j.category.nmy.INIT.AGT=INFO,0,ROL
log4j.category.nmy.INIT.ISF=INFO,0,ROL
log4j.category.nmy.INIT.LOG=INFO,0,ROL
log4j.category.nmy.INIT.UTL=INFO,0,ROL

log4j.category.nmy.STATE.BLM=INFO,DISABLED
log4j.category.nmy.EXCEPTION.BLM=INFO,DISABLED
log4j.category.nmy.SESSION.BLM=INFO,DISABLED
log4j.category.nmy.REQUEST.BLM=INFO,DISABLED
log4j.category.nmy.MESSAGE.BLM=INFO,DISABLED
log4j.category.nmy.OBJECT.BLM=INFO,DISABLED
log4j.category.nmy.DATA.BLM=INFO,DISABLED
log4j.category.nmy.NONE.BLM=INFO,DISABLED
...
```

3 Get all types of event, but block events generated by the JSPF and the JSP – no debug information, rolling file

```
# EVENTS
log4j.category.nmy.INIT.BLM=INFO,0,ROL
log4j.category.nmy.INIT.DAL=INFO,0,ROL
log4j.category.nmy.INIT.AGT=INFO,0,ROL
log4j.category.nmy.INIT.ISF=INFO,0,ROL
log4j.category.nmy.INIT.LOG=INFO,0,ROL
log4j.category.nmy.INIT.UTL=INFO,0,ROL

log4j.category.nmy.STATE.BLM=INFO,0,ROL
log4j.category.nmy.STATE.DAL=INFO,0,ROL
log4j.category.nmy.STATE.AGT=INFO,0,ROL
log4j.category.nmy.STATE.ISF=INFO,0,ROL
log4j.category.nmy.STATE.LOG=INFO,0,ROL
log4j.category.nmy.STATE.UTL=INFO,0,ROL

log4j.category.nmy.EXCEPTION.BLM=INFO,0,ROL
log4j.category.nmy.EXCEPTION.DAL=INFO,0,ROL
log4j.category.nmy.EXCEPTION.AGT=INFO,0,ROL
log4j.category.nmy.EXCEPTION.ISF=INFO,0,ROL
log4j.category.nmy.EXCEPTION.LOG=INFO,0,ROL
log4j.category.nmy.EXCEPTION.UTL=INFO,0,ROL
...
```

Note:

- You have to define filtering for all couples (Event Type, Event Module) that you are interested in.
- Blocking JSPF and JSP is achieved by not mentioning the modules at all.

Auditing

You can monitor the system User Events to track the actions of users. For instance, you can track the events of a user who adds a service to their contract, from the moment they click the submit button to the confirmation message received from the OSS.

User events belong to one of the following categories:

- Session events (login, logout, session expiration)
- Execution features (creation of requests)
- Custom user events
- DO events generated by an OSS

There is a set of predefined user event types in the CID. These user event types are stored in the `USER_TYPE_EVENT` table. User events are logged in the `USER_EVENT` table.

Using User Events involves:

- Activating user events
- Viewing user events
- Managing user events
- Creating custom user events
- Purging the User Event table

Activating User Events

By default, all user events are inactive. Before you can start using user events for an audit, you need to activate the user events you want to track.

User event settings are reference data. If you want to automatically reload this information after changing your settings, refer to *Reloading Reference Data* in *Developing Account Mangement Applications*.

To activate user events

1. Use your database tool to connect to the CID.
2. In the `USER_TYPE_EVENT` table, find the user event you want to activate.
3. In the `ACTIVATION_FLAG` column, enter 1.
4. Save your changes.
5. Do one of the following:

- Use the Reference Data Reload feature to refresh the reference data
- Restart your application server.

To disactivate user events

1. Use your database tool to connect to the CID.
2. In the `USER_TYPE_EVENT` table, find the user event you want to activate.
3. In the `ACTIVATION_FLAG` column, enter 0.
4. Save your changes.
5. Do one of the following:
   - Use the Reference Data Reload feature to refresh the reference data
   - Restart your application server.

Generating Reports

The cidAdminTool comes with a command you use to quickly generate logon statistics. If you need to create your own customized reports, you can use the methods in the `com.netonomy.blm.interfaces.event` package to display and create other reports.

To generate a login report

1. Go to `<home_dir>/bin`.
2. Run the `cidAdminTool` administration tool. Use the syntax:

   ```
   cidAdminTool generate_logon_stats <CID> <CID_ADMIN login>
   <CID_ADMIN password> <m|q|y> [-quiet]
   ```

   where `<CID>`:

   - Oracle: `<instance alias>`
   - DB2: `<instance alias>`
   - SQL Server: `<database host> [:<port>]` If no port is specified, the tool uses the default SQL server port
   - `m` generate stats for the last completed month
   - `q` generate stats for the last completed quarter
   - `y` generate stats for the last completed year

   The CID administration tool extracts the information and displays the location and filename of the extracted information. The report files are located in `<home_dir>/<var>/output/admin`.

Purging User Events

You can remove user events that are saved in the CID.

To purge user events

1 Go to `<home_dir>/bin`.

2 Run the CID administration tool. Use the syntax:

```
cidAdminTool purge_user_events <CID> <CID_ADMIN login>
<CID_ADMIN password> <days> [<code>,<code>,...] [-quiet] [-
datablocksize:####]
```

where `<CID>`:

- Oracle: `<instance alias>`
- DB2: `<database alias>`
- SQL Server: `<database host> [:<port>]` If no port is specified, the tool uses the default SQL server port

The CID administration tool removes all of the user events that:

- Have the specified statuses
- Are older than the specified number of days

The CID administration tool displays a confirmation message.

If you do not specify a user event code, the CID administration tool purges all of the user events.

CHAPTER 4

Managing Distributed Architectures

In This Section

About Distributed Architectures

Except for some development computers, you do not install all of the components on a single computer. Your application server, Web server, OSS systems and databases may be on different computers and even may be running different operating systems.

This section covers some of the things you have to keep in mind when working in distributed architectures.

Installing and Configuring for Distributed Architectures

Because components may be installed on several different computers, you need to make sure that they have all of the required access and permissions for shared directories.

You need to make sure that the components can access and has write permission for the following directories:

- `<home_dir>/var` and its subdirectories
- `<home_dir>/share` and its subdirectories

Your TSM application uses this directory to store logs, connector message queues, and so on.

Managing Sessions

Your TSM application supports session serialization to take advantage of the load balancing and failover features of your application server. You can use your application server's features to:

- Maintain information through HTTP session
- Persist this information in a database, file system, and so on
- Share the session context between application server instances

By default, all of the TSM objects support session serialization.

When customizing your application, you neet to make sure you implement the `java.io.Serializable` interface in order to support session serialization. For more information about implementing this interface, refer to your application server's documentation.

If you use a database to store sessions, you must store TSM sessions in one block. You cannot store them in multiple rows.

APPENDIX A

Administration Tool Reference

In This Section

About the Administration Tools

A complete set of administration tools helps you manage the different components of your TSM application.

The name of the files depends on your operating system. For Windows, the tools have a `.cmd` extension and for UNIX a `.sh` extension. The behavior of these tools is identical and they have the same command line options.

Make sure that the `CLASSPATH` and `LIB_PATH` in these files include the path to all required files. If not, open the required `.env` files and add the paths where required.

Synchronizer Connector Administration Tools

You use the following administration tools to administrate the Synchronizer Connector:

- `agentstart`
- `agentadm`

These tools are located in `<home_dir>/bin`.

agentstart Syntax

```
agentstart <connector_name>
```

PARAMETERS	DESCRIPTION
<connector_name>	Name of the synchronizer connector to start

agentadm Syntax

```
agentadm help | <host> <port> <command> [<parameters>]
```

PARAMETERS	DESCRIPTION
help	Displays help for the tool
<host>	Specifies the agent host
<port>	Specifies the agent administration port
<command>	Administration command

COMMAND	DESCRIPTION	PARAMETERS	RETURN VALUES
start	Resumes the execution of the connector. Use this command to restart the connector after a stop command.	none	-200 AGENT START
stop	Stops the connector. The connector no longer processes inbound and outbound messages	none	-120 AGENT STOP

COMMAND	DESCRIPTION	PARAMETERS	RETURN VALUES
shutdown	Terminates the connector	none	-100 AGENT SHUTDOWN
setparameter	Changes the settings of the connector remotely. Use the following syntax: setparameter <parameter>=<value>	AGENTNAPPING AGENTLOOPING ONE_EXECUTION NBRROW QFILL MAXNUMBERRETRY RETRYDELAY	-510 AGENT SET<parameter name>=<parameter value>
getparameter	Restarts the current value of the parameter	AGENTNAPPING AGENTLOOPING ONE_EXECUTION NBRROW QFILL MAXNUMBERRETRY RETRYDELAY Any parameter defined in the properties files	-610 AGENT GET<parameter name>=<parameter value>
getmode	Displays the current running mode	none	-210 AGENT NORMAL -220 AGENT REDUCE -230 AGENT PAUSE
getstatus	Displays the current status	none	130 AGENT NORMAL -140 AGENT REDUCE -150 AGENT PAUSE
force	Forces the connector to run in the specified mode	NORMAL REDUCE RECOVER	-410 AGENT SET MODE=NORMAL -410 AGENT SET MODE=REDUCE -410 AGENT SET MODE=RECOVER For MODE=RECOVER, an additional parameter must be provided: FILTER <FilterName>
info	Displays information about the connector	none	[AGENTNAME:xxx,NBTHREADS:xxx,FILL:xxx,NBROW:xxx,NAPPING:xxx,LOOPING:xxx,VERSION:xxx,BUILD:xxx,OS:xxx]
kill	Forces shutdown	none	none
list	Returns all the profiling information	none	none
purge	Purges the profiling information	none	none
stat	Returns the profiling information of the last profiling element	none	none
version	Returns the version of the connector	none	none

OSS Connector Administration Tools

You use the following administration tools to administrate the OSS connector:

- `ossstart`
- `ossadm`

These tools are located in `<home_dir>/bin`.

ossstart Syntax

```
ossstart help | <connector>
```

PARAMETERS	DESCRIPTION
<connector>	Name of the connector to start.

ossadm Syntax

```
ossadm help | <host> <port> <command>
```

PARAMETERS	DESCRIPTION
help	Displays help for the tool
<host>	Specifies the connector host
<port>	Specifies the connector port
<command>	Administration command

COMMAND	DESCRIPTION	PARAMETERS	RETURN VALUES
shutdown	Terminates the ossagent	none	none
kill	Forces shutdown	none	none

Approval Sequencer Administration Tools

You use the following administration tools to administrate the Approval Sequencer:

- `approvalsequencer`
- `approvalsequenceradm`

These tools are located in `<home_dir>/bin`.

approvalsequencer Syntax

```
approvalsequencer
```

The configuration parameters are in the `agent.properties` configuration file. This file is located in `<home_dir>/config/approvalsequencer`.

approvalsequenceradm Syntax

```
approvalsequenceradm help | <host> <port> <command>
```

PARAMETERS	DESCRIPTION
help	Displays help for the tool
<host>	Specifies the Approval Sequencer host
<port>	Specifies the Approval Sequencer administration port
<command>	Administration command

COMMAND	DESCRIPTION	PARAMETERS	RETURN VALUES
shutdown	Terminates the Approval Sequencer	none	none
kill	Forces shutdown	none	none

CID Administration Tool

You use the `cidAdminTool` to manage the CID.

This administration tool is located in `<home_dir>/bin`.

For very large databases, this tool comes with a datablock parameter for commands that manage tables containing large amounts of data (requests, trouble tickets, and so on.) This parameter lets you set the number of records to process before the tool commits changes. This way, you can handle large amounts of data in smaller blocks of information that are easier to manage and help avoid various issues while interacting with the database. You can also override this setting from the command line when administrating the CID.

However, by limiting the number of items the tool may have to submit several changes to the database. For instance, if your `REQUEST` table contains 3,000 requests to purge and your default datablock size is 1,000, the tool commits 3 changes. If an error occurs during this process, there is no way to carry out a global rollback of your changes.

Configuring the CID Administration Tool

After installation, you can change the settings of the CID administration tool.

You use the `cid_tools.properties` configuration file to set the properties of the CID administration tool. This file is located in `<home_dir>/lib/admin/cid`.

Configuring the CID Administration tool involves:

- Specifying the JDBC driver
- Specifying the database URL
- Specifying the encoding
- Specifying the data block size

When configuring the CID Administration tool, keep in mind that you can override the data block size when running the tool.

To modify the settings of the CID administration tool

1. Go to `<home_dir>/lib/admin/cid`.
2. Open `cid_tools.properties`.
3. Change the following settings:

SETTING	DESCRIPTION
DRIVER	Class name of the ODBC driver to use
URL	URL of the CID database
ENCODING	Encoding to use
DATABLOCKSIZE	Default number of items to handle before submitting the data. The CID administration tool processes the data in sets corresponding to the specified size until done. This value can be overridden when running the CID administration tool.

4 Save your changes.

Example of cid_tool.properties for Oracle

JDBC driver	# driver: This is the class name of the jdbc driver
	driver=oracle.jdbc.driver.OracleDriverdriver=oracle.jdbc.driver.OracleDriver

URL of the database	# url:This is url for the database (typically jdbc:oracle:oci8:@OracleCidName)
	url=jdbc:oracle:oci8:@CID

Internal settings
DO NOT MODIFY
If you need to modify these settings, you must reinstall and enter the new values during installation.
	# SQL Base Path
	SqlBasePath=<home_dir>/data/oracle/
	# LOG Path
	LogPath=<home_dir>/var/logs/admin/
	# OUTPUT PATH
	OutPath=<home_dir>/var/output/admin/
	# VERSION STRING
	Version=X.X.X.XXX

Encoding	# ENCODING
	Encoding=ISO-8859-1

Data Block Size	# DATABLOCKSIZE
	DataBlockSize=10000

cidAdminTool Syntax

```
cidAdminTool help | <command> <parameters> [<optional
parameters>] <help>
```

PARAMETERS	DESCRIPTION
help	Displays help for the tool
<command>	Administration command
<parameters>	Specifies the parameters for the command
[<optional parameters>]	Optional parameters
<command> help	Displays help for the command

COMMAND	DESCRIPTION	PARAMETERS
admin_notifications	Manages notifications stored in the CID notification queue	<CID> • Oracle: <instance alias> • DB2: <database alias> • SQL Server: <database host> [:<port>] If no port is specified, the tool uses the default SQL server port <CID_ADMIN login> login for the CID admin account <CID_ADMIN password> password for the CID admin account [-quiet] Do not display information [<datablocksize:####>] specifies the number of items to process before committing. Enter 0 to override default settings.
admin_requests	Manages requests stored in the CID request queue	<CID> • Oracle: <instance alias> • DB2: <database alias> • SQL Server: <database host> [:<port>] If no port is specified, the tool uses the default SQL server port <CID_ADMIN login> login for the CID admin account <CID_ADMIN password> password for the CID admin account [-quiet] Do not display information [<datablocksize:####>] specifies the number of items to process before committing. Enter 0 to override default settings.

COMMAND	DESCRIPTION	PARAMETERS
create_cid_users	Creates the CID_ADM and CID_USR users and their corresponding roles	<CID> • Oracle: <instance alias> • SQL Server: <database host> [:<port>];DatabaseName="<db_name>" If no port is specified, the tool uses the default SQL server port For SQL Server, note that the semicolon is a special character of UNIX and Windows shell. The CID parameter should be enclosed by quotation marks. For DB2, the CID users are system users and cannot be created using this tool. <login> DBA Login ID <password> DBA Login password < new CID_ADMIN login> login for the CID admin account <new CID_ADMIN password> password for the CID admin account <new CID_USER login> login for the CID user account <new CID_USER password> password for the CID user account [<quiet>] Do not display information
create_demo_cid_structure	Creates a CID demo database structure with reference data, but without the demo dataset	<CID> • Oracle: <instance alias> • DB2: <database alias> • SQL Server: <database host> [:<port>] If no port is specified, the tool uses the default SQL server port <CID_ADMIN login> login for the CID admin account <CID_ADMIN password> password for the CID admin account [-quiet] Do not display information

COMMAND	DESCRIPTION	PARAMETERS
create_demo_cid_test	Creates a CID demo database structure with reference data, and populates it with the demo dataset	<CID> • Oracle: <instance alias> • DB2: <database alias> • SQL Server: <database host> [:<port>] If no port is specified, the tool uses the default SQL server port <CID_ADMIN login> login for the CID admin account <CID_ADMIN password> password for the CID admin account [-quiet] Do not display information
drop_demo_cid_structure	Drops a CID demo database structure along with the populated data	<CID> • Oracle: <instance alias> • DB2: <database alias> • SQL Server: <database host> [:<port>] If no port is specified, the tool uses the default SQL server port <CID_ADMIN login> login for the CID admin account <CID_ADMIN password> password for the CID admin account [-quiet] Do not display information
generate_logon_stats	Extracts logon statistics from CID to a file	<CID> • Oracle: <instance alias> • DB2: <database alias> • SQL Server: <database host> [:<port>] If no port is specified, the tool uses the default SQL server port <CID_ADMIN login> login for the CID admin account <CID_ADMIN password> password for the CID admin account <m|q|y> m generate stats for the last completed month q generate stats for the last completed quarter y generate stats for the last completed year [-quiet] Do not display information

COMMAND	DESCRIPTION	PARAMETERS
install_cid	Creates the CID database structure and populates the required reference tables	<CID> • Oracle: <instance alias> • DB2: <database alias> • SQL Server: <database host> [:<port>] If no port is specified, the tool uses the default SQL server port <CID_ADMIN login> login for the CID admin account <CID_ADMIN password> password for the CID admin account <CID_USER login> login for the CID user account <CID_USER password> password for the CID user account [-quiet] Do not display information
purge_requests	Purges the requests stored in the CID request queue	<CID> • Oracle: <instance alias> • DB2: <database alias> • SQL Server: <database host> [:<port>] If no port is specified, the tool uses the default SQL server port <CID_ADMIN login> login for the CID admin account <CID_ADMIN password> password for the CID admin account <days> The age of the requests to keep in days. For example, entering 7 purges all of the requests that were not created during the last week. [<status>,<status>...] The status of the requests to purge [-quiet] Do not display information [<datablocksize:####>] specifies the number of items to process before committing. Enter 0 to override default settings.

COMMAND	DESCRIPTION	PARAMETERS
purge_persistent_action _managers	Purges the persistent action managers (shopping carts) stored in the CID	<CID> • Oracle: <instance alias> • DB2: <database alias> • SQL Server: <database host> [:<port>] If no port is specified, the tool uses the default SQL server port <CID_ADMIN login> login for the CID admin account <CID_ADMIN password> password for the CID admin account <days> The age of the persistent action managers to keep in days. For example, entering 7 purges all of the persistent action managers that were not created during the last week. <category> The category of persistent action managers to purge. By default: • 1 backup shopping carts • 2 contract templates [-quiet] Do not display information [<datablocksize:####>] specifies the number of items to process before committing. Enter 0 to override default settings.
purge_trouble_tickets	Purges the trouble tickets stored in the CID	<CID> • Oracle: <instance alias> • DB2: <database alias> • SQL Server: <database host> [:<port>] If no port is specified, the tool uses the default SQL server port <CID_ADMIN login> login for the CID admin account <CID_ADMIN password> password for the CID admin account <days> The age of the trouble tickets to purge in days [<status>,<status>...] The status of the trouble tickets to purge [-quiet] Do not display information [<datablocksize:####>] specifies the number of items to process before committing. Enter 0 to override default settings.

COMMAND	DESCRIPTION	PARAMETERS
purge_user_events	Purges the user events stored in the CID	<CID> ▪ Oracle: <instance alias> ▪ DB2: <database alias> ▪ SQL Server: <database host> [:<port>] If no port is specified, the tool uses the default SQL server port <CID_ADMIN login> login for the CID admin account <CID_ADMIN password> password for the CID admin account <days> The age of the events to keep in days. For example, entering 7 purges all of the events that were not created during the last week. [<event_code>[,<event_code>...] The code of the type(s) of events to purge. If none entered, all events are purged. The event code corresponds to the the USER_TYPE_EVENT_ID in the USER_TYPE_EVENT table. [-quiet] Do not display information [<datablocksize:####>] specifies the number of items to process before committing. Enter 0 to override default settings.
purge_notifications	Purges the notifications stored in the CID notification queue	<CID> • Oracle: <instance alias> • DB2: <database alias> • SQL Server: <database host> [:<port>] If no port is specified, the tool uses the default SQL server port <CID_ADMIN login> login for the CID admin account <CID_ADMIN password> password for the CID admin account <days> The age of the notifications to keep in days. For example, entering 7 purges all of the notifications that were not created during the last week. [<status>,<status>...] The status of the notifications to purge. If none specified, the tool purges FAILED and DONE notifications. [-quiet] Do not display information [<datablocksize:####>] specifies the number of items to process before committing. Enter 0 to override default settings.

Index

E

H

I

L

M

O

P

R

S

T

U

W