Oracle® Communications Convergent Charging Controller

Notification Gateway Technical Guide

Release 12.0.2

December 2018

ORACLE

Copyright

Contents

About This Document

Scope

The scope of this document includes all the information required to install and configure the Oracle Communications Convergent Charging Controller Notification Gateway application.

Audience

This guide was written primarily for system administrators and persons installing and configuring the Notification Gateway application. The documentation assumes that the person using this guide has a good technical knowledge of the system.

Prerequisites

Although there are no prerequisites for using this guide, familiarity with the target platform would be an advantage.

A solid understanding of Unix and a familiarity with IN concepts are an essential prerequisite for safely using the information contained in this guide. Attempting to install, remove, configure or otherwise alter the described system without the appropriate background skills, could cause damage to the system; including temporary or permanent incorrect operation, loss of service, and may render your system beyond recovery.

This manual describes system tasks that should only be carried out by suitably trained operators.

Related documents

The following documents are related to this document:

- *ACS User's Guide*
- *ACS Technical Guide*
- *CPE User's Guide*
- *OSD User's and Technical Guide*

Document Conventions

Typographical Conventions

The following terms and typographical conventions are used in the Oracle Communications Convergent Charging Controller documentation.

Formatting Convention	Type of Information
Special Bold	Items you must select, such as names of tabs. Names of database tables and fields.
Italics	Name of a document, chapter, topic or other publication. Emphasis within text.
Button	The name of a button to click or a key to press. **Example:** To close the window, either click **Close**, or press **Esc**.
Key+Key	Key combinations for which the user must press and hold down one key and then press another. Example: **Ctrl+P** or **Alt+F4**.
`Monospace`	Examples of code or standard output.
`Monospace Bold`	Text that you must enter.
variable	Used to indicate variables or text that should be replaced with an actual value.
menu option > menu option >	Used to indicate the cascading menu option to be selected. Example: **Operator Functions > Report Functions**
hypertext link	Used to indicate a hypertext link.

Specialized terms and acronyms are defined in the glossary at the end of this guide.

Chapter 1

System Overview

Overview

Introduction

This chapter provides a high-level overview of the Notification Gateway. It explains the basic functionality of the system and lists the main components.

It is not intended to advise on any specific Oracle Communications Convergent Charging Controller network or service implications of the product.

In this Chapter

This chapter contains the following topics.

About the Notification Gateway

The Convergent Charging Controller Notification Gateway (NGW) receives notifications from Oracle Communications Billing and Revenue Management (BRM) Elastic Charging Engine (ECE) and transforms those notifications into messages that Convergent Charging Controller can pass to customers. For example, the following occurs when a customer reaches a credit threshold limit and needs to top-up their account:

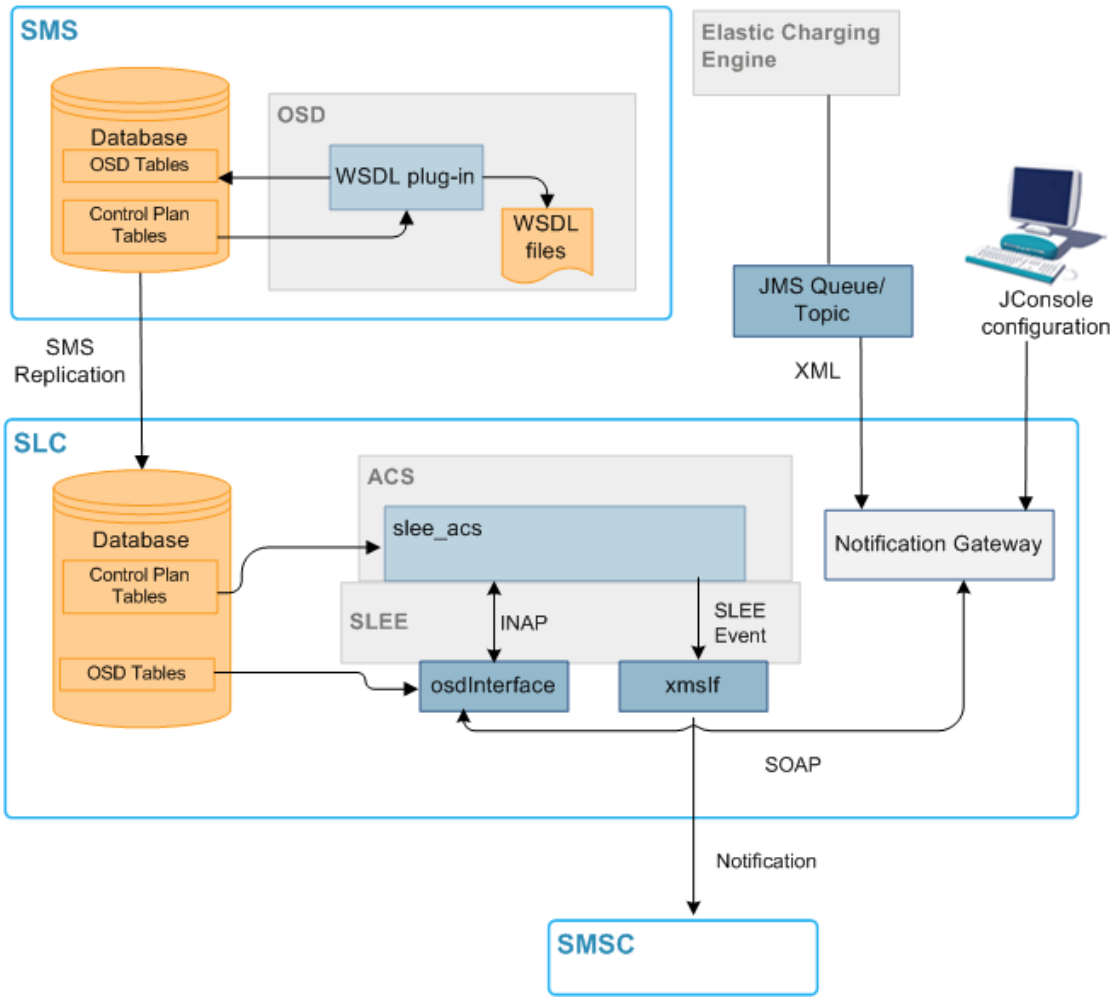
1. ECE sends a notification to a WebLogic Server JMS queue.
2. The NGW connected to the queue receives the message and transforms it into Convergent Charging Controller data.
3. The NGW triggers a control plan to send a message to the customer. The message can be sent by using SMS.
4. The NGW removes the notification from the queue.

To handle large numbers of messages, you can configure multiple NGWs. When you configure a NGW, do the following:

- Configure the NGW to connect to a JMS queue on which ECE publishes notifications.
- Configure which types of messages to send to customers
- Configure how to map notifications into Convergent Charging Controller format.

Notification Gateway Architecture

The following figure shows the system architecture for the Notification Gateway.



Notification Gateway Components

The following table describes the Notification Gateway components.

Component	Description
/IN/services_packages/NGW/etc/config.xml	The Notification Gateway configuration file
/IN/services_packages/NGW/bin/ngw	The Notification Gateway interface
/IN/service_packages/NGW/xsl	The transform file directory

Notification Types Supported by Convergent Charging Controller

The following table lists the notification types that are sent by ECE and can be processed by the Notification Gateway. ECE may send other notification types, but those are not currently supported by the Notification Gateway.

Notification Type	Description
Advice of Charge	Sent when a charge is applied to a subscriber's account.

Notification Type	Description
Aggregated Threshold Breach	Sent when a subscriber's balance breaches multiple thresholds. For example, if thresholds are configured for $80 and $78 and a subscriber's balance passes through both or these values on the same call.
Credit Ceiling	Sent when a subscriber's balance breaches a credit ceiling. ECE includes information about the breach in the notification message, such as the balance element and the balance ID of the balance that was breached.
Credit Floor	Sent when a subscriber's balance breaches a credit floor. ECE includes information about the breach in the notification message, such as the balance element and the balance ID of the balance that was breached.
First Use Validity and Expiry (if applicable)	Sent when a subscriber first uses a balance. ECE includes information about the balance expiry date in the message. The balance expiry date is derived using an offset from when the balance is first used; for example, if the balance had an initial first use top up of $50 and an expiry of 20 days, then on first use, the expiry date is set to the date of the first call plus 20 days.
Offering Validity Initialization	Sent when a subscriber buys or changes a charging offering. It contains the action type, such as the charge and the new offering validity dates.
Threshold Breach	Sent when a subscriber's account breaches a credit threshold value. It includes details about the threshold that was breached.
Top-up	Sent when a top-up is performed on a subscriber's balance.

Java Requirements

The NGW requires that Java SE 8 is installed on SLC nodes. You can download the latest version of Java SE 8 from the following location:

http://java.oracle.com

You should download the version for Solaris SPARC 64-bit.

You must set one of the following environment variables in the **/etc/profile** file to the Java home path:

- JAVA8_HOME
- JAVA_HOME

For example, to set the home path in the JAVA8_HOME environment variable to **/usr/jre/latest**, do the following:

As the root user, edit the **/etc/profile** file to include the following lines:

```
JAVA8_HOME=/usr/jre/latest
export JAVA8_HOME
```

JMS Configuration in ECE

ECE uses the Java Message Service (JMS) to publish external notifications to the notification queue (JMS topic) on the JMS server. The Notification Gateway processes those notifications and sends them on to customers.

For information about configuring ECE to publish external notifications to JMS topics, see *BRM Elastic Charging Engine Implementation Guide*.

Notification Gateway Control Plans

A control plan controls session flow through a series of connected feature nodes, where each feature node in the control plan performs a specific function. You specify the session flow logic in the control plan configuration. You configure control plans in the Convergent Charging Controller Control Plan Editor (CPE). See *Control Plan Editor User's Guide* for more information.

The Notification Gateway includes a number of example control plans. Each Notification Gateway control plan defines how to process a specific notification type.

Open Services Development (OSD)

Convergent Charging Controller Open Services Development (OSD) enables service providers to submit Web Services Design Language (WSDL) files. The Notification Gateway uses OSD WSDL files to trigger the control plans that send notifications to customers. To create WSDL files, you configure operations in OSD and associate the operations with control plans. When you save and compile OSD control plans in the Control Plan Editor, you can select to generate a WSDL file for the associated operation.

See *Open Services Development User's and Technical Guide* for more information about OSD.

Note: The OSD WSDL files define operations that are invoked by the NGW when making requests for those operations to the osdInterface process. osdInterface communicates with slee_acs, which is the component that actually triggers the control plans.

Notification Gateway Transform Files

BRM ECE publishes external notifications to JMS topics in XML format. The Notification Gateway transforms the notifications into SOAP messages and sends them to the OSD osdInterface for processing. The Notification Gateway includes **XSL** files that define how to transform the XML notifications into SOAP messages.

Chapter 2

Configuring Convergent Charging Controller for Notification Gateway

Overview

Introduction

This chapter explains how to configure Convergent Charging Controller applications to enable the Convergent Charging Controller Notification Gateway (NGW) to work.

In this chapter

This chapter contains the following topics.

Configuring Convergent Charging Controller for Notification Gateway

To configure Convergent Charging Controller for the Notification Gateway, complete these procedures in the order listed:

Step	Action
1	Configure the external subscriber service on the SLC node. See *Configuring the External Subscriber Service* (on page 5).
2	Configure Charging Control Services (CCS) balance type mappings. See Configuring Convergent Charging Controller Balance Type Mappings.
3	Configure the ACS notification templates for the different notification types. See *Configuring Notification Templates in ACS* (on page 7).
4	Configure the Open Services Development operations for the Notification Gateway. See *Configuring Open Services Development (OSD) Operations* (on page 8).
5	If required, import the Notification Gateway sample control plans. See *Importing Notification Gateway Control Plans* (on page 9). Alternatively, you can create control plans to process and send notifications, for which the supplied sample control plan can be used as a basis. Control plans must be valid OSD control plans.

Configuring the External Subscriber Service

The external subscriber service maps external balances to Convergent Charging Controller balances. The Notification Gateway uses the external subscriber service to handle balance mapping between ECE balances and Convergent Charging Controller balances. The balance mapping enables the Notification Gateway to send the balance information received in ECE notifications to the subscriber.

To configure the external subscriber service:

Step	Action
1	Stop the SLEE on the SLC node.
2	Open the **acs.conf** file on the SLC node in a text editor.
3	Specify the following entry in the acsChassis section: **`ServiceEntry(Ext_Sub,ccsSvcLibrary.so)`** where `Ext_Sub` is the service handle for the external subscriber service.
4	Save and close the **acs.conf** file.
5	Define the `Ext_Sub` service in the **SLEE.cfg** configuration file on the SLC node by doing the following: a. Open the **SLEE.cfg** file in a text editor. b. Specify SERVICE and SERVICEKEY entries for the `Ext_Sub` service: **`SERVICE=Ext_Sub 1 slee_acs Ext_Sub`** **`SERVICEKEY=INTEGER`** *`int`* **`Ext_Sub`** where *int* is an unused service key number. The standard value for `Ext_Sub` is 137. Use this value if no other SERVICEKEY of 137 is defined in **SLEE.cfg**. c. Save and close the **SLEE.cfg** file.
6	Open the **eserv.config** file in a text editor.

7	In the CCS, ccsServiceLibrary section, configure the `extSubDomainTypeId` parameter:
	`ccsServiceLibrary = {`
	`  extSubDomainTypeId = int`
	`}`
	where `extSubDomainTypeId` specifies the ID of the domain type that the `Ext_Sub` service will use for mapping external balances to Convergent Charging Controller balances. The default value is 2 (for the Diameter domain).

8	In the CCS, ccsnotificationPlugin section, configure the `extSubServiceKey` parameter:
	`notificationPlugin = {`
	`  extSubServiceKey = int`
	`}`
	where `extSubServiceKey` is the service key value that you set for the `Ext_Sub` service in the **SLEE.cfg** file (Step 5).

9	Save and close the **eserv.config** file.
10	Restart the SLEE.

Configuring Convergent Charging Controller Balance Type Mappings

Balance type mappings enable the `Ext_Sub` service to convert the balance types in ECE notifications to equivalent Convergent Charging Controller balance types. This balance information can then be substituted in Convergent Charging Controller notifications.

You map an ECE balance type to a Convergent Charging Controller balance type by specifying the following:

- The Convergent Charging Controller balance type that is included in Convergent Charging Controller notifications
- The ECE balance ID
- The conversion scale to use when converting balance values
- The ECE domain

You must configure a balance type mapping for each ECE balance type that will be used in ECE notifications processed by the Notification Gateway.

To configure a balance type mapping:

Step	Action
1	Log in to the SMS UI.
2	From the **Services** menu, select **Prepaid Charging > Service Management**. The Service Management screen is displayed.
3	From the **Service Provider** list, select the service provider.
4	Click the **Balance Type Mapping** tab.
5	Click **New**. The New Balance Type Mapping dialog box is displayed.
6	From the **Balance Type** list, select the name of the Convergent Charging Controller balance type.
7	In the **Third-Party Resource** field, enter the equivalent ECE balance ID.
8	In the **Third-Party Resource Scale** field, enter the conversion scale factor to apply between Convergent Charging Controller and ECE balances. For example, to convert Convergent Charging Controller dollar balances held in cents to ECE dollar balances held in dollars, set this field to 100. In this case, Convergent Charging Controller balances are divided by 100 before they are sent to ECE. That is, an Convergent Charging Controller balance of 500 cents is divided by 100 to convert it into an ECE balance of 5 dollars.
9	From the **Domain Type** list, select the ECE charging domain. The ID for the selected domain type must correspond to the external domain type defined in the `extSubDomainTypeId` parameter. See *Configuring the External Subscriber Service* (on page 5).
10	Click **Save**.

For more information about configuring balance type mappings, see *CCS User's Guide*.

Configuring Notification Templates in ACS

The Notification Gateway delivers notifications to customers based on information it receives from a third party such as the Oracle Communications Billing and Revenue Management (BRM) Elastic Charging Engine (ECE). You configure the content and format of the different types of notifications sent to customers in the Advanced Control Services (ACS) Configuration screens in the ACS UI.

You configure a specific notification type for each type of notification that will be output by the Notification Gateway; for example, configure a notification type named Topup Notification for topup notifications. After you configure a notification type, you configure the notification templates that are associated with it. The notification template configuration enables you to specify the language to use for the notification, the text to output in the notification, and any variable elements that you want to include, such as balance information. You should include one or more of the following variable elements in your notification templates:

- Balance
- Charge
- Recharge

See the discussion of ACS configuration in *ACS User's Guide* for more information about configuring ACS notifications.

Configuring Open Services Development (OSD) Operations

Open Services Development (OSD) allows service providers to define OSD operations that are then expressed as Web Services Design Language (WSDL) files, defining SOAP requests that can then be sent to osdInterface on the SLC. OSD WSDL files are created by configuring OSD operations and then associating the operations with ACS control plans. The operations are defined in the OSD UI on the **Operation Sets** tab and **Operations** tab, and the association with an ACS control plan is made when the control plan is saved in the ACS Control Plan Editor.

For processing notifications from the Notification Gateway, you configure an OSD operation set and associated operation for each notification type.

You must ensure the following details are configured in the OSD UI:

- The service provider that uses the Notification Gateway.
- The client ASP for the Notification Gateway.

See *OSD User's and Technical Guide* for more information about configuring service providers and client ASPs in the OSD UI.

To configure OSD notification type operations for a service provider:

Step	Action
1	Open the **Open Services Development** window in the SMS UI.
2	Select the **Service Providers** tab.
3	Select the service provider for the Notification Gateway from the **Service Provider** list.
4	Configure the SLC ports for the Notification Gateway. Ensure that the interface name is set to `osdInterface`.
5	Select the **Operation Sets** tab.
6	Do the following for each type of notification: a. Add an operation set, giving the operation set the name of the notification type; for example ADVICE_OF_CHARGE_EVENT. b. Set the **Service to Invoke** for the operation set to `Ext_Sub`. c. Save the operation set.
7	Create an operation with the same name as the notification. a. Click the **Operations** tab. The screen includes the **Service Provide**r and **Operation Set to Use** tabs. b. In the **Control Plan** field, select **None Specified**. c. Select the **Enabled** check box. d. Save the operation.
8	Open the Control Plan Editor.

Step	Action
9	Do the following for each type of notification: a. Create a control plan that uses the **Send Notification** feature node in its configuration. Alternatively, you can import the Notification Gateway sample control plans. See *Importing Notification Gateway Control Plans* (on page 9). b. Ensure that the **Send Notification** feature node is configured to send the ACS notification template for that type of notification. c. Save the control plan. d. In the **Save** dialog, select the previously configured OSD operation for that type of notification in the **Generate WSDL for operation** list. For example, select the ADVICE_OF_CHARGE_EVENT operation when saving the ADVICE_OF_CHARGE_EVENT control plan. A WSDL file for the selected operation is created in the **/IN/html/wsdls/***service_provider* directory.

10	Check that the WSDL files for each notification type have been generated. Each generated WSDL file (including hostname and directory path) must correspond to the SERVICE/URI entries for that notification type in the Oracle Communications Convergent Charging Controller section of **config.xml**, as in the following example:
	`<SERVICE>`
	`<URI>http://NCC-SMS-HOST/wsdls/PROVIDER/ADVICE_OF_CHARGE_EVENT.wsdl</URI>`
	`<OPERATION>ADVICE_OF_CHARGE_EVENT</OPERATION>`
	`</SERVICE>`
	See *Notification Gateway Configuration File* (on page 15) for more information.

11	Return to the **Operations** tab in the OSD UI.
12	Do the following for each notification type: a. Refresh the operation details by finding and displaying the operation on the **Operations** tab. b. Check that the name of the associated control plan has been populated in the **Control Plan** field. c. Select the **Enabled** field. d. Click **Save**.
12	Select the **Client ASPs** tab in the OSD UI.
13	Find and display the client ASP for the Notification Gateway on the **Client ASPs** tab.
14	Add each of the Notification Gateway operations for the service provider to the **Allowed Operations** list.
15	Click **Save**.

Importing Notification Gateway Control Plans

The Notification Gateway includes the following sample control plan files that you can import into the ACS Control Plan Editor (CPE). Each control plan defines how to process a specific notification type:

- ADVICE_OF_CHARGE_EVENT
- AGGREGATED_THRESHOLD_BREACH_EVENT
- CREDIT_CEILING_BREACH_NOTIFICATION_EVENT
- CREDIT_FLOOR_BREACH_NOTIFICATION_EVENT

- EXTERNAL_TOP_UP_NOTIFICATION_EVENT
- THRESHOLD_BREACH_EVENT
- FIRST_USE_VALIDITY
- OFFERING_VALIDITY_INITIALIZATION

Follow these steps to import a sample control plan into the CPE.

Step	Action
1	Open the ACS CPE.
2	From the **File** menu, select **Import from file**. The **Open** dialog box appears.
3	Navigate to the folder containing the control plan file that you want to import. The Notification Gateway control plan files are installed on SLC nodes in the following directory: **/IN/services_packages/NGW/cpl**
4	Select the control plan file to import and then click **Open**. The CPE imports the control plan and displays it in the CPE canvas in the Control Plan Editor window.
5	Ensure that the control plan configuration includes the feature node you want to use to send notifications; for example, you can configure a control plan to send notifications by including the **Send Notification** feature node in the control plan. You specify the notification template to use in the feature node configuration. See *Feature Nodes Reference Guide* for more information about Convergent Charging Controller feature nodes.
6	Save the control plan and generate the OSD operation **.wsdl** file for the notification type at the same time. To generate the OSD operation **.wsdl** file, in the **Save** dialog select the associated OSD operation from the **Generate WSDL for operation** list. For example, to generate the **ADVICE_OF_CHARGE.wsdl** file, select the ADVICE_OF_CHARGE operation from the list. The CPE creates the operation **.wsdl** file in the following directory on the SMS node: **/IN/html/wsdls/***service_provider* where *service_provider* is the name of the service provider for whom the OSD operations have been configured. See Convergent Charging Controller CPE Help for more information about saving control plans.

Chapter 3

Notification Gateway Configuration

Overview

Introduction

This chapter explains how to configure the Oracle Communications Convergent Charging Controller application.

In this chapter

This chapter contains the following topics.

About Notification Gateway Configuration

To configure the Notification Gateway, complete these procedures in the order listed:

Step	Action
1	Configure secure connection to the Oracle Database for the Notification Gateway. See *Configuring Notification Gateway for Secure Connections to the Database* (on page 11).
2	Set the httpAuth username and password in OSD. See *Setting the httpAuth Username and Password in OSD* (on page 12)
3	Configure the Notification Gateway to receive notifications from ECE. See *Configuring Notification Gateway to Receive ECE Notifications* (on page 12).
4	Configure T3s protocol connections between the Notification Gateway and the ECE WebLogic Server instance. See Configuring T3s Connections to the ECE WebLogic Server Instance.
5	If required, configure the Notification Gateway for high availability. See *Configuring Connections for High Availability* (on page 14).
6	Configure Notification Gateway message options. See *Configuring Notification Gateway Message Options* (on page 14).
7	Configure Notification Gateway message type services. See *Configuring the Notification Gateway Message Type Services* (on page 14).
8	Configure how notifications from ECE will be mapped to Convergent Charging Controller formats. See *Configuring How to Map Notifications into Convergent Charging Controller Format* (on page 15).

Configuring Notification Gateway for Secure Connections to the Database

Secure connections to the database is set up automatically when the Convergent Charging Controller patch is installed on the SMS node. After the Convergent Charging Controller patch installation, set the following in the Convergent Charging Controller section of the **config.xml** file:

```
CCC
```

```
<DBUSER>/@SCP</DBUSER>
```

This applies the database username and password stored in the credentials vault.

See *Service Management System Technical Guide* for more information about secure connections to the database.

About the Oracle Wallet for Notification Gateway

The Oracle wallet for the Notification Gateway is created automatically during the Convergent Charging Controller patch installation. You need not create it manually.

Setting the httpAuth Username and Password in OSD

The httpAuth user enables the Notification Gateway to access OSD remotely. You set the user credentials (username and password) for the httpAuth user on a service provider basis, on the **Notification Gateway** tab in the OSD UI. The username and password are stored in a secure credentials vault on the SMS.

The **Notification Gateway** tab is available in the OSD UI only if the `jnlp.ECEExtensions` Java application property is present and set to true in the **/IN/html/sms.jnlp** configuration file. See *SMS Technical Guide* for more information.

Note: You can override user credentials by setting the [SERVICE/USER] and [SERVICE/PASS] parameters in the Notification Gateway **config.xml** file. You should set these parameters only if you do not want to store user credentials in the Convergent Charging Controller secure credentials vault.

Follow these steps to set the user credentials for the httpAuth user for a selected service provider.

Step	Action
1	Open the OSD UI and select the **Notification Gateway** tab in the Open Services Development window.
2	Select the **Service Provider** from the drop down list.
3	Enter the name of the authorized user of the Notification Gateway in the **User Name** field.
4	Enter a new password for the user in the **Change Password** field.
5	Re-enter the password in the **Confirm Password** field.
6	Click **Save**. **Result:** The user credentials (username and password) are stored in the Convergent Charging Controller secure credentials vault on the SMS.

Configuring Notification Gateway to Receive ECE Notifications

Follow these steps to configure NGW to receive ECE notifications.

Step	Action
1	On the SLC node go to the directory containing the NGW configuration file, **config.xml**: `cd /IN/service_packages/NGW/etc`
2	Edit the **config.xml** file by using a text editor.

Step	Action
3	In the ECE, JMS section, configure the JMS provider URL for the ECE weblogic server by editing the following line: `<PROVIDER_URL>t3://ece_host:port</PROVIDER_URL>` where: • *ece_host* is the ECE hostname. You must change the ECE-WEBLOGIC-HOST default value to the ECE hostname for the JMS notifications weblogic instance. • *port* is the port used for receiving ECE notifications. The default port is 7001.
4	Add ECE hostname and IP address to the **/etc/hosts** file on the SLC.
5	Save and close the **config.xml** file.

Configuring T3s Connections to the ECE WebLogic Server Instance

Follow these steps to set up T3s secure protocol connections between the Notification Gateway process on the SLC and the ECE WebLogic Server instance.

Step	Action
1	Copy the **client.jks** file from ECE to a local SLC directory: `scp User@Hostname:/scratch/User/opt/WEBLOGIC_KEYSTORE/client.jks .` where: • *User* is the user that installed the ECE software. • *Hostname* is the hostname of the ECE machine.
2	List the certificates in your Java keystore by entering the following command: `keytool -v -list -keystore ~/certificate/client.jks` The keytool utility should return the value **weblogic**.
3	Export the trusted certificate from the JKS file to the **weblogic.cert** file. For example: `keytool -exportcert -alias weblogic -file ./weblogic.cert -keystore ./client.jks`
4	Log in as the sudo user.
5	Set the JAVA_HOME environment variable to the same value as the acs_oper user.
6	Import the trusted certificate into the keystore for SLC by entering the following command: `keytool -importcert -alias weblogic -file ./weblogic.cert -keystore $JAVA_HOME/jre/lib/security/cacerts`

Follow these steps to return to a T3 protocol connection in the future:

Step	Action
1	Open the **/IN/service_packages/NGW/etc/config.xml** file in a text editor.

2	Replace the following line:
	`<PROVIDER_URL>t3s://DomainName:7002</PROVIDER_URL>`
	with the following:
	`<PROVIDER_URL>t3://DomainName:7001</PROVIDER_URL>`
	Where *DomainName* is the fully qualified domain name of your ECE host.
	Note: Port numbers 7002 and 7001 are default values. Your ECE WebLogic Server port numbers may be different.

Step	Action
3	Export the trusted certificate from the JKS file to the **weblogic.cert** file. For example: `keytool -exportcert -alias weblogic -file ./weblogic.cert -keystore ./client.jks`
4	Log in as the sudo user.
5	Set the JAVA_HOME environment variable to the same value as the acs_oper user.
6	Import the trusted certificate into the keystore for SLC by entering the following command: `keytool -importcert -alias weblogic -file ./weblogic.cert -keystore $JAVA_HOME/jre/lib/security/cacerts`

Configuring Connections for High Availability

For high availability, modify the <PROVIDER_URL> entry in the NGW configuration file, **config.xml**, to include multiple ECE hostnames. For example,

```
<PROVIDER_URL>t3://den00bjy.us.oracle.com:7003,den00bke.us.oracle.com:7003,den00hbg.
us.oracle.com:7003</PROVIDER_URL>
```

This allows the NGW to connect to the defined ECE JMS queues and in the case where one queue goes out of service, NGW can get the notifications from an alternative queue.

Configuring Notification Gateway Message Options

Follow these steps to configure JMS message options for NGW.

Step	Action
1	On the SLC node, go to the directory containing the NGW configuration file (**config.xml**): `cd /IN/service_packages/NGW/etc`
2	Open the **config.xml** file in a text editor.
3	Set the parameters in the ECE JMS section as required. See *ECE JMS Message Configuration Section* (on page 16) for more information about the available parameters.
4	Save and close the **config.xml** file.

Configuring the Notification Gateway Message Type Services

This section explains the procedure to configure the Notification Gateway message type services.

Notification Gateway Message Types

The ECE notification types that can be processed by the Notification Gateway are included in the following list:

- CREDIT_FLOOR_BREACH_NOTIFICATION_EVENT
- AGGREGATED_THRESHOLD_BREACH_EVENT
- OFFERING_VALIDITY_INITIALIZATION_EVENT
- THRESHOLD_BREACH_EVENT
- FIRST_USAGE_VALIDITY_INIT_EVENT
- ADVICE_OF_CHARGE_EVENT
- EXTERNAL_TOP_UP_NOTIFICATION_EVENT

- CREDIT_CEILING_BREACH_NOTIFICATION_EVENT

Each notification type must be defined in a <SERVICE> entry in the Convergent Charging Controller section of the **config.xml** file. See *Notification Gateway Configuration File* (on page 15) for more information

For example:

```
<SERVICE>

<URI>http://slc03tig/wsdls/ECE_SP/CREDIT_FLOOR_BREACH_NOTIFICATION_EVENT.wsdl</URI>
          <OPERATION>CREDIT_FLOOR_BREACH_NOTIFICATION_EVENT</OPERATION>
      </SERVICE>
```

where:

- The <URI> entry specifies the location of the notification associated OSD WSDL file and includes the SMS hostname, wsdls directory, service provider name, and OSD operation wsdl.
- The <OPERATION> entry specifies the OSD operation name. The operation name matches the notification name.

Configuring How to Map Notifications into Convergent Charging Controller Format

Follow these steps to configure XML SOAP transformation for NGW.

Step	Action
1	On the SLC node go to the directory containing the NGW configuration file, **config.xml**: `cd /IN/service_packages/NGW/etc`
2	Edit the **config.xml** file by using a text editor.
3	In the TRANSFORM section, set: • DIR to the full directory path for the Notification Gateway **.xsl** files • TRANSFORM_POOL to the required thread pool size
4	For each notification type, set FILE to the file name of the corresponding **.xsl** file.
5	Save and close the **config.xml** file. See *Notification Gateway Configuration File* (on page 15) for more information about the **config.xml** file.

About the Notification Gateway Configuration File

Notification Gateway Configuration File

You specify configuration for the Notification Gateway in the **config.xml** file, located in the following directory:

/IN/service_packages/NGW/etc

The **config.xml** file contains the following sections:

- ECE JMS
- CCC
- TRANSFORM

ECE JMS Message Configuration Section

The following table describes the parameters in the ECE JMS section of the **config.xml** configuration file. The parameters are listed in order of appearance in the configuration file.

Field	Default	Description
PROVIDER_URL	`ECE-WEBLOGIC-HOST:7001`	String. Defines the ECE hostname and port in the JMS provider URL for the ECE weblogic server.
INITIAL_CONTEXT_FACTORY	`weblogic.jndi.WLInitialContextFactory`	String. The JMS context factory on the ECE weblogic server.
BINDING	`ECE/NotificationFactory`	String. The JMS binding on the ECE weblogic server.
TOPIC	`ECE/NotificationTopic`	String. The JMS topic where messages are written by ECE.
CLIENT_ID	`NCC.NGW.SHARED`	String. The client ID to use to request a durabale subscription to the JMS topic. You must change the default value if another client is using the same name.
VERSION_HEADER	`NOTIFICATION_VERSION`	String. The JMS user header containing the XML message version. The NGW matches XML messages containing a version header only if the JMS header name field matches the VERSION_HEADER setting and there is an XSL template file that uses the name format: NOTIFICATION_TYPE.VERSION.xsl By default the NGW matches only messages that do not contain a version header.
MSG_DUMP_POOL	`2`	Integer. The thread pool size for write-to-file message threads.
MSG_DUMP_DIR	`/IN/service_packages/NGW/tmp/`	String. The directory path for the directory where message dump files are written.
KEEP_NEW_MSG	False	Boolean. If set to true, saves all the new incoming JMS messages.
MAX_NEW_MSG	True	Boolean. If KEEP_NEW_MSG is set to true, indicates the maximum number of new messages that will be saved.
KEEP_RESENT_MSG	False	Boolean. If KEEP_RESENT_MSG is set to true, saves all the resent incoming JMS messages.
MAX_RESENT_MSG	1000	Integer. If KEEP_RESENT_MSG set to true, indicates the maximum number of resent messages that will be saved.
KEEP_REJECTED_MSG	False	Boolean. If KEEP_REJECTED_MSG is set to true, saves all the rejected incoming JMS messages.
MAX_COMMS_ERROR_MSG	True	Boolean. if KEEP_REJECTED_MSG set to true, indicates the maximum number of resent messages that will be saved.
KEEP_COMMS_ERROR_MSG	True	Boolean. Saves all the JMS messages which had an outgoing SOAP failure due to communications failure. Manual intervention via jconsole is required to resend the messages after the error is fixed.
MAX_COMMS_ERROR_MSG	1000	Boolean. if KEEP_COMMS_ERROR_MSG set to true, indicates the maximum number of comms error messages that will be saved. The default value can be modified.

Field	Default	Description
KEEP_DOC_ERROR_MSG	True	Boolean. Saves all the JMS messages which had an outgoing SOAP failure due to communications failure. Manual intervention via jconsole is required to resend the messages after the error is fixed.
MAX_DOC_ERROR_MSG	1000	
THROTTLE_ENABLED	True	Enables a delay in JMS message acknowledgment
THROTTLE_DELAY_TIME_MS	10	Indicates the delay in JMS message acknowledgement.

Convergent Charging Controller Configuration Section

The following table describes the parameters in the CCC section of the **config.xml** configuration file. The parameters are listed in order of appearance in the configuration file.

Field	Default	Description
DBUSER	enter /@SCP	Uses the secure vault 'username/password' as the user connection data
DBHOST	N/A	Hostname of server running the NGW. For example, slctjw.us.oracle.com
DBID	N/A	Leave blank
SERVICE_INVOKE_POOL	4	Number of simultaneous Java threads allocated to the NGW connection to OSD
SERVICE	N/A	Creates one service entry for each notification type
URI	N/A	Storage path to the notification associated OSD WSDL file that displays SMS hostname, wsdls directory, Service Provider Name, and OSD operation wsdl
OPERATION	N/A	Indicates the OSD operation name
GROUPS	N/A	Port groups for a particular service.
GROUP	active="false"	Attribute "name" provides the jconsole WebService Group display name. Duplicate group names are merged into a single group. Attribute "active" determines the status of the constituent ports at startup time.
PORT	N/A	PORT name should match names in the service WSDL. A group may contain any subset of WSDL defined ports.

TRANSFORM Configuration Section

The following table describes the parameters in the TRANSFORM section of the **config.xml** configuration file. The parameters are listed in order of appearance in the configuration file.

Field	Default	Description
TRANSFORM_POOL	4	The thread pool size for the XSL transform threads.
DIR	N/A	The directory path of the Notification Gateway **.xsl** files

Field	Default	Description
FILE	N/A	The name of the **.xsl** file for a specific notification type. One FILE entry per notification type.

Sample config.xml File

The following sample **config.xml** file shows the default Notification Gateway configuration at installation.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<NCC_ECC_Notification_Gateway>
    <ECE>
        <JMS>
            <PROVIDER_URL>t3://ECE-WEBLOGIC-HOST:7001</PROVIDER_URL>
            <INITIAL_CONTEXT_FACTORY>weblogic.jndi.WLInitialContextFactory</INITIAL_C
            ONTEXT_FACTORY>
            <BINDING>ECE/NotificationFactory</BINDING>
            <TOPIC>ECE/NotificationTopic</TOPIC>

            <CLIENT_ID>NCC.NGW.SHARED</CLIENT_ID>
            <VERSION_HEADER>NOTIFICATION_VERSION</VERSION_HEADER>

            <MSG_DUMP_POOL>2</MSG_DUMP_POOL>
            <MSG_DUMP_DIR>/IN/service_packages/NGW/tmp/</MSG_DUMP_DIR>

                <KEEP_NEW_MSG>false</KEEP_NEW_MSG>
                <KEEP_RESENT_MSG>false</KEEP_RESENT_MSG>
            <KEEP_REJECTED_MSG>true</KEEP_REJECTED_MSG>
            <KEEP_COMMS_ERROR_MSG>true</KEEP_COMMS_ERROR_MSG>
            <KEEP_DOC_ERROR_MSG>true</KEEP_DOC_ERROR_MSG>

                <MAX_NEW_MSG>100000</MAX_NEW_MSG>
            <MAX_RESENT_MSG>1000</MAX_RESENT_MSG>
            <MAX_REJECTED_MSG>1000</MAX_REJECTED_MSG>
            <MAX_COMMS_ERROR_MSG>1000</MAX_COMMS_ERROR_MSG>
            <MAX_DOC_ERROR_MSG>1000</MAX_DOC_ERROR_MSG>

            <THROTTLE_ENABLED>true</THROTTLE_ENABLED>
            <THROTTLE_DELAY_TIME_MS>10</THROTTLE_DELAY_TIME_MS>
        </JMS>
    </ECE>
    Oracle Communications Convergent Charging Controller
        <DBUSER>/@SCP</DBUSER>
        <DBHOST>NCC-DATABASE-HOST</DBHOST>
        <DBID>PORT:SERVICE</DBID>
        <SERVICE_INVOKE_POOL>4</SERVICE_INVOKE_POOL>
        <SERVICE>
            <URI>http://NCC-SMS-HOST/wsdls/PROVIDER/ADVICE_OF_CHARGE_EVENT.wsdl</URI>
            <OPERATION>ADVICE_OF_CHARGE_EVENT</OPERATION>
            <GROUPS>
              <GROUP name="GroupA" active="true">
                 <PORT>ADVICE_OF_CHARGE_EVENTPort3</PORT>
                 <PORT>ADVICE_OF_CHARGE_EVENTPort4</PORT>
                 <PORT>ADVICE_OF_CHARGE_EVENTPort5</PORT>
              </GROUP>
              <GROUP name="GroupB" active="false">
                 <PORT>ADVICE_OF_CHARGE_EVENTPort1</PORT>
                 <PORT>ADVICE_OF_CHARGE_EVENTPort2</PORT>
              </GROUP>
            </GROUPS>
        </SERVICE>
        <SERVICE>
```

```
            <URI>http://NCC-SMS-
            HOST/wsdls/PROVIDER/EXTERNAL_TOP_UP_NOTIFICATION_EVENT.wsdl</URI>
            <OPERATION>EXTERNAL_TOP_UP_NOTIFICATION_EVENT</OPERATION>
            <GROUPS>
              <GROUP name="GroupA">
                <PORT>EXTERNAL_TOP_UP_NOTIFICATION_EVENTPort3</PORT>
                <PORT>EXTERNAL_TOP_UP_NOTIFICATION_EVENTPort4</PORT>
                <PORT>EXTERNAL_TOP_UP_NOTIFICATION_EVENTPort5</PORT>
              </GROUP>
            </GROUPS>
        </SERVICE>
        <SERVICE>
            <URI>http://NCC-SMS-HOST/wsdls/PROVIDER/THRESHOLD_BREACH_EVENT.wsdl</URI>
            <OPERATION>THRESHOLD_BREACH_EVENT</OPERATION>
        </SERVICE>
        <SERVICE>
            <URI>http://NCC-SMS-
            HOST/wsdls/PROVIDER/CREDIT_FLOOR_BREACH_NOTIFICATION_EVENT.wsdl</URI>
            <OPERATION>CREDIT_FLOOR_BREACH_NOTIFICATION_EVENT</OPERATION>
        </SERVICE>
        <SERVICE>
            <URI>http://NCC-SMS-
            HOST/wsdls/PROVIDER/CREDIT_CEILING_BREACH_NOTIFICATION_EVENT.wsdl</URI>
            <OPERATION>CREDIT_CEILING_BREACH_NOTIFICATION_EVENT</OPERATION>
        </SERVICE>
        <SERVICE>
            <URI>http://NCC-SMS-
            HOST/wsdls/PROVIDER/AGGREGATED_THRESHOLD_BREACH_EVENT.wsdl</URI>
            <OPERATION>AGGREGATED_THRESHOLD_BREACH_EVENT</OPERATION>
        </SERVICE>
    </NCC>
    <TRANSFORM>
        <TRANSFORM_POOL>4</TRANSFORM_POOL>
        <DIR location="/IN/service_packages/NGW/xsl/">
            <FILE>EXTERNAL_TOP_UP_NOTIFICATION_EVENT.xsl</FILE>
            <FILE>THRESHOLD_BREACH_EVENT.xsl</FILE>
            <FILE>CREDIT_FLOOR_BREACH_NOTIFICATION_EVENT.xsl</FILE>
            <FILE>CREDIT_CEILING_BREACH_NOTIFICATION_EVENT.xsl</FILE>
            <FILE>ADVICE_OF_CHARGE_EVENT.xsl</FILE>
            <FILE>AGGREGATED_THRESHOLD_BREACH_EVENT.xsl</FILE>
        </DIR>
    </TRANSFORM>
</NCC_ECC_Notification_Gateway>
```

Chapter 4

Starting and Stopping the Notification Gateway

Overview

Introduction

This chapter explains how to start and stop the Oracle Communications Convergent Charging Controller Notification Gateway (NGW) `ngw` process.

In this chapter

This chapter contains the following topics.

About the ngw Process

The Notification Gateway `ngw` process accepts ECE notification XML messages from an ECE JMS topic and transforms the messages into SOAP requests. `ngw` then sends the SOAP requests on to the Convergent Charging Controller `osdInterface` for processing.

`ngw` runs automatically on Service Logic Controller (SLC) nodes. It is located in the following directory:

/IN/service_packages/NGW/bin

Starting the Notification Gateway

Follow these steps to start `ngw` on the SLC.

Step	Action
1	As the user `root`, edit **/etc/inittab** on the SLC.

2	Edit the following entry for `ngw1`:
	`ngw1:34:`**`off:`**`/IN/service_packages/SMS/bin/cmnSU - acs_oper -c "exec /IN/service_packages/NGW/bin/NGWStartup.sh >> /IN/service_packages/NGW/tmp/NGW.log 2>&1" > /dev/null 2>&1 0<&1`
	
	Currently this entry is set to **`off`**. Change **`off`** to **`respawn`**:
	
	`ngw1:34:`**`respawn:`**`/IN/service_packages/SMS/bin/cmnSU - acs_oper -c "exec /IN/service_packages/NGW/bin/NGWStartup.sh >> /IN/service_packages/NGW/tmp/NGW.log 2>&1" > /dev/null 2>&1 0<&1`

Step	Action
3	Restart the `ngw` process by entering this command: **`init q`**

Stopping the Notification Gateway

Follow these steps to stop the `ngw` process on the SLC node.

Step	Action
1	Log in to the SLC as the user `root`.
2	Edit the **/etc/inittab** file to stop the Convergent Charging Controller `ngw` process.

3	Edit the following entry for `ngw1:`
	`ngw1:34:`**`respawn:`**`/IN/service_packages/SMS/bin/cmnSU - acs_oper -c "exec /IN/service_packages/NGW/bin/NGWStartup.sh >> /IN/service_packages/NGW/tmp/NGW.log 2>&1" > /dev/null 2>&1 0<&1`
	
	Currently this entry is set to **`respawn`**. Change **`respawn`** to **`off`**:
	
	`ngw1:34:`**`off:`**`/IN/service_packages/SMS/bin/cmnSU - acs_oper -c "exec /IN/service_packages/NGW/bin/NGWStartup.sh >> /IN/service_packages/NGW/tmp/NGW.log 2>&1" > /dev/null 2>&1 0<&1`

4	Enter the following command: **`init q`** **Result:** The commented out **inittab** process will stop.

Chapter 5

Configuring Notification Gateway Run-Time Options

Overview

Introduction

This chapter explains the run-time configuration options for Oracle Communications Convergent Charging Controller Notification Gateway (NGW).

In this chapter

This chapter contains the following topics.

Notification Gateway Run-Time Options

You can configure the following run-time options in JMS for the Notification Gateway:

- Global configuration. See *Updating Global Configuration at Run-Time* (on page 23)
- JMS subscription configuration. See *Updating JMS Subscription Configuration at Run-Time* (on page 25)
- Notification Type web services. See *Updating Web Notification Service Configuration at Run-Time* (on page 29)
- Notification XML transform files. See XML SOAP Transformation Parameters

Updating Global Configuration at Run-Time

This section describes the NGW global configuration options that you can set at run-time.

Global Parameters

NGW supports the following global run-time parameters:

`LoggerLevel`

Description: Set the log4j log level

Type: String

Permissible Values: NONE, INFO, DEBUG, TRACE

Default: DEBUG

`RuntimeConfigXml`

Description: XML representation of currently active configuration. You can view the value for this parameter but you cannot change it at run-time.

Getter/Setter: G

Type: String

Permissible Values: Valid XML

Default: config.xml settings

`writeConfig`

Description: Writes the currently active configuration to file

Type: Void

Permissible Values N/A

Return Type: String

Return Value: Success status

Setting the Log Level

Follow these steps to set the log level run-time option.

Step	Action
1	Access the Notification Configuration MBeans by performing the following: a. Log onto the driver machine. b. Start a JMX editor that allows you to edit the MBean attributes. If you use JConsole, enter, **`$ jconsole &`** c. Do one of the following: – If the driver machine and the server machine are the same, select **Local Process**. – If the driver machine is synchronized onto a remote server machine, select **Remote Process**. d. Enter either the IP address or the host name and the port number of the Notification Gateway node you have enabled for JMX-management in your topology. e. Click **Insecure Connection** and select the **MBeans** tab. f. Expand the **NotificationsGateway** navigation tree.
2	Expand **`Config`**.
3	Expand **`Global`**, **`Attributes`**. The list of available parameters is displayed. You can modify only run-time parameters that are highlighted blue.
4	Specify the level of information to display in log files in the **`LoggerLevel`** parameter. Specify one of: `NONE`, `INFO`, `DEBUG`, `TRACE`, or `ERROR`.
5	Press the **Enter** key.

Step	Action
6	To save configuration changes in the **config.xml** file, do the following: a. Expand **Global**, **Operations**. b. Click **writeConfig**. Your configuration changes are saved and will be loaded if the Notification Gateway process is stopped and then restarted.

Updating JMS Subscription Configuration at Run-Time

This section describes the NGW JMS subscription configuration options that you can set at run-time.

JMS Subscriptions Parameters

NGW supports the following JMS subscription run-time parameters:

JmsRuntimeConfigXML

Description: XML representation of currently active configuration
Getter/Setter: G
Type: String
Permissible Values: Valid XML
Default: **config.xml** settings for ECE, JMS section

_topic

Description: The JMS topic to which we are a subscriber.
Type: G, String
Permissible Values: Notifications topic name configured on the ECE weblogic server
Default: **config.xml** TOPIC setting

_throttle_enabled

Description: Enable JMS acknowledgment throttling
Getter/Setter: GS
Type: Boolean
Permissible Values: True, False
Default: **config.xml** THROTTLE_ENABLED setting

_throttle_delay_time_ms

Description: JMS acknowledgment throttling in milliseconds
Getter/Setter: GS
Type: Integer
Permissible Values: Any positive value in milliseconds
Default: **config.xml** THROTTLE_DELAY_TIME_MS setting

_provider_url

Description: The JMS Provider URL for the ECE weblogic server

Getter/Setter: G

Type: String

Permissible Values: N/A

Default: **config.xml** PROVIDER_URL setting

_msg_dump_pool

Description: Thread pool size for message write-to-file threads.

Getter/Setter: GS

Type: Integer

Permissible Values: N/A

Default: **config.xml** MSG_DUMP_POOL setting

_msg_dump_dir

Description: Directory where messages dump files will be written

Getter/Setter: GS

Permissible Values: Any valid directory, writable by the notifications gateway process user on the SLC where the Notification Gateway is running.

Default: **config.xml** MSG_DUMP_DIR setting

_max_resent_msg

Description: Maximum number of resent messages retained in hash map. A flag in the JMS header indicates if the message has been resent.

Getter/Setter GS

Type: Integer

Permissible Values: Any positive integer

Default: **config.xml** MAX_RESENT_MSG setting

_max_rejected_msg

Description: Maximum number of rejected messages retained in hash map. Rejected messages are those accepted by the message selector, but with an unhandled version.

Getter/Setter: GS

Type: Integer

Permissible Values: Any positive integer

Default: **config.xml** MAX_REJECTED_MSG setting

_max_new_msg

Description: Maximum number of new messages retained in hash map. New messages are those sent by Weblogic for the first time. The resent flag in the JMS header is set to false.

Getter/Setter: GS

Type: Integer

Permissible Values: Any positive integer

Default: **config.xml** MAX_NEW_MSG setting

_max_doc_error_msg

Description: Maximum number of messages with a document error retained in hash map. A document error can be result of:

- An XML parsing error
- Invalid XSL transform
- SOAP fault (i.e., message is rejected by OSD interface)

Getter/Setter: GS

Type: Integer

Permissible Values: Any positive integer

Default: **config.xml** MAX_DOC_ERROR_MSG setting

_max_comms_error_msg

Description: Maximum number of messages with a communications error retained in hash map. A communications error can be the result of:

- No response from any WSDL-specified OSD port for the service
- All WSDL-specified service ports marked locally out-of-service by NGW

Getter/Setter: GS

Type: Integer

Permissible Values: Any positive integer

Default: **config.xml** MAX_COMMS_ERROR_MSG setting

_keep_resent_msg

Description: Instruct notification gateway to keep resent messages

Getter/Setter: GS

Type: Boolean

Permissible Values: true, false

Default: **config.xml** KEEP_RESENT_MSG setting

_keep_rejected_msg

Description: Instructs the notification gateway to keep rejected messages

Getter/Setter: GS

Type: Boolean

Permissible Values: true, false

Default: **config.xml** KEEP_REJECTED_MSG setting

_keep_new_msg

Description: Instructs the notification gateway to keep new messages

Getter/Setter: GS

Type: Boolean
Permissible Values: true, false
Default: **config.xml** KEEP_NEW_MSG setting

_keep_doc_error_msg

Description: Instructs the notification gateway to keep document error messages
Getter/Setter: GS
Type: Boolean
Permissible Values: true, false
Default: **config.xml** KEEP_DOC_ERROR_MSG setting

_keep_comms_error_msg

Description: Instructs the notification gateway to keep communications error messages
Getter/Setter: GS
Type: Boolean
Permissible Values: true, false
Default: **config.xml** KEEP_COMMS_ERROR_MSG setting

_jms_version_header

Description: Version string to match in JMS message header
Getter/Setter: GS
Type: String
Permissible Values: Any string
Default: **config.xml** VERSION_HEADER setting

_jms_client_id

Description: Clent ID used to request a durable subscription to the JMS topic on the weblogic instance.
Getter/Setter: G
Type: String
Permissible Values: Any string not already in use by another JMS subscriber
Default: **config.xml** CLIENT_ID setting

_initial_context_factory

Description: The JMS context factory on the ECE weblogic server
Getter/Setter: G
Type: String
Permissible Values: N/A
Default: **config.xml** INITIAL_CONTEXT_FACTORY setting

`_binding`

Description: The JMS context factory on the ECE weblogic server.

Getter/Setter: G

Type: String

Permissible Values: N/A

Default: **config.xml** BINDING setting

Setting JMS Subscription Parameters

Follow these steps to set JMS subscription run-time options.

Step	Action
1	Access the Notification Configuration MBeans by performing the following: a. Log onto the driver machine. b. Start a JMX editor that allows you to edit the MBean attributes. If you use JConsole, enter, **`$ jconsole &`** c. Do one of the following: – If the driver machine and the server machine are the same, select **Local Process**. – If the driver machine is synchronized onto a remote server machine, select **Remote Process**. d. Enter either the IP address or the host name and the port number of the Notification Gateway node you have enabled for JMX-management in your topology. e. Click **Insecure Connection** and select the **MBeans** tab. f. Expand the **NotificationsGateway** navigation tree.
2	Expand **`Config`**, then **`JMSSubscription`**.
3	Expand **`Attributes`**. The list of available parameters is displayed. You can modify only run-time parameters that are highlighted blue.
4	Specify the values for the parameters you want to change. See *JMS Subscriptions Parameters* (on page 25) for details of the available parameters.
5	Press the **Enter** key.
6	To save configuration changes in the **config.xml** file, do the following: a. Expand **`Operations`**. b. Click **writeConfig**. Your configuration changes are saved and will be loaded if the Notification Gateway process is stopped and then restarted.

Updating Web Notification Service Configuration at Run-Time

This section describes the NGW web notification service configuration options that you can set at run-time.

WebServices Parameters

NGW supports the following WebServices run-time parameters:

_services_str

Description: Currently configured Web services represented in the form:
`username:password@http://hostname.domain/service.wsdl`

Note: The password field will be censored.

Getter/Setter: G

Type: String array

Permissible Values: URI strings

Default: **config.xml** DBUSER and SERVICE URI settings

_service_invoke_pool

Description: Thread pool size for service connection and invocation threads

Getter/Setter: G

Type: Integer

Permissible Values: Positive integer

Default: config.xml SERVICE_INVOKE_POOL setting

_number_services

Description: Number of configured web services

Getter/Setter: G

Type: Integer

Permissible Values: Any positive integer

Default: None

WsRuntimeConfigXML

Description: XML representation of currently active configuration

Getter/Setter: G

Type: String

Permissible Values: Valid XML

Default: config.xml settings for NCC_ECC_Notification_Gateway --> ECE --> JMS section

WebServices Operations

NGW supports the following JMS web services run-time operations:

addService

Description: Adds a Web Service

Input Type: String

Permissible Values: The following, for a valid Web Service:
1. Username
2. Password
3. WSDL Operation
4. WSDL URL

Return Type: String

Return Value: Success status

`removeService`

Description: Removes a Web Service

Input Type: String

Permissible Values: Valid Web Service operation name for a currently configured service.

Return Type: String

Return Value: Success status

About WebService Operations

You can perform the following web service operations at run-time:

- View the list of configured notification services. See *Viewing a List of Configured Notification Services* (on page 31).
- Add or remove a notification service. See *Adding or Removing a Notification Service* (on page 31).
- Update a web service configuration for a notification type. See *Updating a WebService Configuration* (on page 32).
- Set the service status for ports. See *Setting the Service Status for Ports* (on page 33).

Viewing a List of Configured Notification Services

Follow these steps to view the list of configured notification services.

Step	Action
1	Expand the **NotificationsGateway** navigation tree on the **MBeans** tab in JConsole.
2	Click **Configuration**.
3	Click **WebService**.
4	Click **Attributes**.
	The `_services_str` parameter displays all the configured notification services.
5	Double click the value to view the detailed data.

Adding or Removing a Notification Service

Follow these steps to add or remove a notification service.

Step	Action
1	Expand the **NotificationsGateway** navigation tree on the **MBeans** tab in JConsole.
2	Click **Configuration**.
3	Click **WebService**.

Step	Action
4	Click **Operations**.
5	To add a notification service: a. Select the **addService** operation. b. Populate the four string fields as follows: – **p1**=username – **p2**=password – **p3**=OSD operation name – **p4** = OSD WSDL URL; for example, **http://slc03tig//wsdls/ECE_SP/ADVICE_OF_CHARGE_EVENT.wsdl** for the ADVICE_OF_CHARGE_EVENT notification service c. Click **addService**. The notification will be added.
6	To remove a notification service: a. Select the **removeService** operation. b. Enter the WSDL name in the **p1** string field. For example, ADVICE_OF_CHARGE_EVENT.wsdl **Note:** The WSDL name must match existing WSDL name stored in the WebServices attribute, `_services_str`. c. Click **removeService**. The notification will be removed.

Updating a WebService Configuration

Individual WebServices (for example, WebService.1, WebService.2 and so on) can be configured or updated in NGW.

Follow these steps to update the WebService configuration.

Step	Action
1	Expand the **NotificationsGateway** navigation tree on the **MBeans** tab in JConsole.
2	Click **Configuration**.
3	Click any Web service option.
4	If you want to edit a Web service value, select **Attributes**. You can edit the fields colored blue by double clicking the value.
5	Select **Operations** to configure or update the following: • setAllPortsServicesStatus: Sets all the OSD ports for a notification service in or out of service. Ports that are in and out of service are displayed in the Attributes screen under PortsInServiceLocal and PortsOutOfServicceLocal fields. – If **p1** = true = in service – If **p1** = false = out of service • setPortServiceStatus: Sets individual port in or out of service, where **p1** is the number for the port you want to set. – If **p2** = true = in service – If **p2** = false = out of service • createServiceWithAttributes: Creates new service attributes associated with a new OSD operation. If you change an existing operation name or WSDL URL values for the service, you can recreate the WebService with the new Attribute values.
6	Click **Refresh**.

Creating a Notification Service

Follow these steps to create a notification service.

Step	Action
1	Expand the **NotificationsGateway** navigation tree on the **MBeans** tab in JConsole.
2	Click **Configuration**.
3	Click any Web service option.
4	Select **Operations**.
5	Select **createServiceWithAttributes**.
6	Specify the attributes for the new service that will be associated with a new OSD operation. If you change an existing operation name or WSDL URL values for the service, you can recreate the Web service with the new attribute values.
7	Click **Refresh**.

Setting the Service Status for Ports

Follow these steps to update the service status for ports.

Step	Action
1	Expand the **NotificationsGateway** navigation tree on the **MBeans** tab in JConsole.
2	Click **Configuration**.
3	Click any Web service option.
4	Select **Operations**. You can configure the following parameters: • setAllPortsServicesStatus. Set **p1** to true or false to set the status of all the OSD ports for a notification service in or out of service. You can view which ports are in or out of service in the Attributes screen in the **PortsInServiceLocal** and **PortsOutOfServiceLocal** fields. – If **p1** = true = in service – If **p1** = false = out of service • setPortServiceStatus: Sets an individual port in or out of service where **p1** = port number – If **p2** = true = in service – If **p2** = false = out of service
5	Click **Refresh**.

Updating WebService Groups at Run-Time

This section describes the NGW WebService Group configuration options that you can set at run-time.

Note: The WebService Group Active status acts as a wrapper on the Instance `setPortServiceStatus()` operations, triggering that operation for all ports in the group. Manually invoking `setPortServiceStatus()` in the WebService Instance will override the WebService Group setting for that port. In other words, WebService Instance `PortsInServiceLocal` and `PortsOutOfServiceLocal` are the authority on the true port status.

`Active`

Description: Batch processing of WebService Instance port status
Getter/Setter: GS
Type: String
Permissible Values:
Default:

`Name`

Description: Name of the WebService Group
Getter/Setter: G
Type: String
Permissible Values:
Default:

`PortsInGroup`

Description: List of WebService Instance ports belonging to the group.
Getter/Setter: G
Type: String
Permissible Values:
Default:

Configuring Notification Type (XML Transform) at Run-Time Options

This section describes the NGW notification type (XML Transform) configuration options that you can set at run-time.

Notification Type Parameters

NGW supports the following Notification Type run-time parameters:

`_versions`

Description: Lists the XSL transform versions cached by NGW. Displays the notification types and version strings.
Getter/Setter: G
Type: String array
Permissible Values: Notification types and versions corresponding to a transform file with the following naming convention:

NOTIFICATION_TYPE.[VERSION.].xsl

(where the square brackets denote the optional version field)

Default: **config.xml** entries: TRANSFORM ->DIR->FILE

`_transform_pool`

Description: Thread pool size for XSL Transform threads.

Getter/Setter: G

Type: Integer

Permissible Values: Positive integer

Default: **config.xml** TRANSFORM_POOL setting

`_directory`

Description: Directory where XLS transform files are stored.

On start-up (and when adding new transforms at runtime) this is the location in which the transform files must reside.

Getter/Setter: G

Type: String

Permissible Values: Valid accessible directory on the NGW SLC

Default: **config.xml** TRANSFORM -> DIR location setting

`_XslRuntimeConfigXML`

Description: XML representation of currently activve configuration

Getter/Setter: G

Type: String

Permissible Values: Valid XML

Default: **config.xml** settings for NCC_ECE_Notfication_Gateway -> TRANSFORM section

Notification Type Operations

NGW supports the following notification type run-time operations:

`addNotificationType`

Description: Adds a new XSL transform specification

Input Type String, String

Permissible Values: The following, for a valid XSL transform file with the naming convention,

NOTIFICATION_TYPE.[VERSION].xsl

residing in the directory given by the _directory attribute:

- Notification type
- Notification version

Return Type: String

Return Value: Success status

deleteNotificationType

Description: Removes a cached XSL transform specification.

Input Type String, String

Permissible Values: The following, for a valid XSL transform file with the naming convention, NOTIFICATION_TYPE.[VERSION].xsl residing in the directory given by the _directory attribute:

- Notification type
- Notification version

Return Type: String

Return Value: Success status

reloadNotificationType

Description: Reloads a cached XSL transform specification.

Input Type String, String

Permissible Values: The following, for a valid XSL transform file with the naming convention, NOTIFICATION_TYPE.[VERSION].xsl residing in the directory given by the _directory attribute:

- Notification type
- Notification version

Return Type: String

Return Value: Success status

Viewing a List of Notification Type Versions

Follow these steps to view the list of configured notification type versions.

Step	Action
1	Expand the **NotificationsGateway** navigation tree on the **MBeans** tab in JConsole.
2	Click **Configuration**.
3	Click **XMLTRANSFORM**.
4	Click **Attributes** and select **`_versions`**. The **`_versions`** parameter displays all the configured notification services.
5	Double click the value for the **`_versions`** field to view the detailed data.

Adding or Deleting a Notification Type

Follow these steps to add or remove a notification type specification.

Step	Action
1	Expand the **NotificationsGateway** navigation tree on the **MBeans** tab in JConsole.
2	Click **Configuration**.
3	Click **XMLTransform**.
4	Click **Operations**.

Step	Action
5	To add a notification type: a. Select the **addNotificationType** operation. b. Populate the two string fields as follows: – **p1**=name – **p2**=version, for example, 1 c. Click **addNotificationType**. The notification type will be added. Different versions of notifications may require different transform files. Therefore, when you add a new notification type to the Transform configuration, state the version.
6	To delete a notification service: a. Select the **deleteNotificationType** operation. b. Enter the notification name in the **p1** string field and the version in the **p2** field. c. Click **deleteNotificationType**. The notification type will be removed.

Reloading Notification Types

Follow these steps to reload a notification type specification.

Step	Action
1	Expand the **NotificationsGateway** navigation tree on the **MBeans** tab in JConsole.
2	Click **Configuration**.
3	Click **XMLTransform**.
4	Click **Operations**.
5	To reload a notification type: a. Select the **reloadNotificationType** operation. b. Populate the two string fields as follows: – **p1**=name – **p2**=version, for example, 1 c. Click **reloadNotificationType**. This action reloads a single transform file using the specified notification name and version.
6	To reload all the notification services: a. Select the **reloadAllNotificationType** operation. b. Click **reloadNotificationType**.This action makes the transform code clear its cache, re-read all the transform files, and then reloads all the transform files.

Testing SOAP Transformation (XML Transform) Configuration at Run-Time

This section provides the details to test the SOAP Transformation (XML Transform) configuration at run-time.

XML SOAP Transformation Parameters

NGW supports the following XML SOAP transformation run-time parameters:

`TransformTimeAverage_ms`

Description: A moving average of the time taken to perform an XML transformation. This is a moving average window of 100 transformations.

Getter/Setter: G

Type: Integer

Permissible Values: Any integer >= 0

Default: N/A

`TransformStats`

Description: Running totals XML Transformations per notification type.

Getter/Setter: G

Type: HashMap<String,Integer>

Permissible Values: Format is: NOTIFICATION_TYPE=<count>

Default: N/A

XML SOAP Transformation Operations

NGW supports the following XML SOAP transformation run-time operations:

`transformXML`

Description: Performs a test XML transformation.

Input Type String, String, String

Permissible Values: The following, for a valid XSL transform file with the naming convention, NOTIFICATION_TYPE.[VERSION].xsl residing in directory given by the _directory attribute:

- Notification type
- Notification version
- Notification XML

Return Type: String

Return Value: One of the following:

- SOAP message
- Failure status

`transformXMLFile`

Description: Tests an XML transformation file.

Input Type String

Permissible Values: A valid XML x`file with the naming convention, NOTIFICATION_TYPE.[VERSION].xml residing in directory given by the _directory attribute:

- Notification type
- Notification version
- Notification XML

Return Type: String

Return Value: One of the following:

- SOAP message
- Failure status

Performing a Test XML Transformation

Both the XML Transform operations, transformXML and transformXMLFile allow you to try running a XML file through the transform process to see if it gets transformed successfully.

For example, if a notification failed the original transformation processing and the operator found an error in the transform file, the operator modifies the transform file and then re-runs the Notification through the modified transform.

There are two operations:

- transformXMLFile allows the operator to select the notification file to be processed by defining a stored file containing the notification in XML format.
- transformXML allows the user to cut and paste the actual XML file into the **p3** field.

Follow these steps to perform a test XML transform:

Step	Action
1	Expand the **NotificationsGateway** navigation tree on the **MBeans** tab in JConsole.
2	Click **XML Transform**.
3	Click **XML -> SOAP transformation**.
4	Click **Operations**.
5	Select transformXML and specify the following : • **p1** = notification name • **p2** = version • **p3** = XML file
6	Store the XML file locally on the NGW server by selecting transformXMLFile and entering the path name to the XML file in the **p1** field; for example, **/IN/service_packages/NGW/tmp/THRESHOLD_BREACH_EVENT.1.0.0.0.0.xml**
7	Click **transformXMLFile**.

Chapter 6

Viewing Notification Gateway Run-Time Statistics

This section provides the details to view the notification gateway run-time statistics.

Overview

Introduction

This chapter describes the Oracle Communications Convergent Charging Controller Notification Gateway run-time statistics.

In this chapter

This chapter contains the following topics.

About Notification Gateway Statistics

About Notification Gateway Run-Time Statistics

Notification gateway displays message statistics at the following locations in JMS:

- **/NotificationsGateway/JMSSubscription/MessageStats/Attributes**
- **/NotificationsGateway/XMLTransform/Attributes**

About Web Services Statistics

This section describes the available WebServices statistics.

`InvokeAttempts`

Description:	Running total of service invoke attempts. A service is invoked each time a SOAP message is sent.
Getter/Setter:	G
Type:	String Array
Permissible Values:	Format is: {WSDL}Operation=<count>
Default:	N/A

InvokeFailures

Description: Running total of service invoke failures. An invocation failure occurs when:

- A service is unreachable
- A service returns a SOAP failure

Getter/Setter: G

Type: String Array

Permissible Values: Format is:
{WSDL}Operation=<count>

Default: N/A

InvokeSuccess

Description: Running total of service Invoke success. An invocation success occurs when a service returns a SOAP success.

Getter/Setter: G

Type: String Array

Permissible Values: Format is:
{WSDL}Operation=<count>

Default: N/A

SOAPFaultCount

Description: Running total of SOAP faults. A SOAP fault occurs when the SLC OSD interface fails to parse the SOAP message.

Getter/Setter: G

Type: Integer

Permissible Values: Any integer >= 0

Default: N/A

SOAPResponseTimeAverage_ms

Description: A moving average of the time taken for the SLC OSD interface to respond to a SOAP message after service invocation.

This is a moving average window of 100 invocations.

Getter/Setter: G

Type: Integer

Permissible Values: Any integer >= 0

Default: N/A

SOAPSuccessCount

Description: Running total of successful SOAP invocations. A successful invocation occurs when the SLC OSD interface returns a SOAP success in response to a SOAP invocation.

Getter/Setter: G

Type: Integer

Permissible Values: Any integer >= 0

Default: N/A

`WsExceptionCount`

Description: Running total of Web service exception messages. A Web service exception occurs when:

- NGW fails to transform the JMS message into SOAP
- All service ports are unavailable

Getter/Setter: G

Type: Integer

Permissible Values: Any integer >= 0

Default: N/A

About JMS Message Statistics

This section describes the available JMSSubscription message statistics.

JMS Message Statistics	Description
MsgConsumed_runningTotal	Running total of messages consumed by the NGW (consumed means that the NGW has returned an 'ACK' to ECE and now owns the Notification).
MsgInNewTimeAverage_ms	Indicates average time in ms to process new message from ECE.
MsgIn New_runningTotal	Indicates running total of messages received from the ECE JMS Queue.
MsgInrejected_runningTotal	Indicates running total of messages received from the ECE JMS Queue that are rejected by the NGW e.g. bad format.
MsgInResent_runningTotal	Indicates running total of messages received that have been resent from the ECE JMS Queue.
MsgOutCommsError_runningTotal	Indicates running total of messages that have failed to be sent to OSD due to communications error e.g. no ports available
MsgOutDocError_runningTotal	Indicates running total of messages that have failed to be sent to OSD due to format error.
StoredMsgInNew_currentTotal	Indicates stored message total
StoredMsgInRejected_currentTotal	Indicates stored message total
StoredMsgInResent_currentTotal	Indicates stored message total
StoredMsgOutCommsError_currentTotal	Indicates stored message total
StoredMsgOutDocError_currentTotal	Indicates stored message total

Note: Configuration of which messages are to be stored and the maximum numbers to be stored is done in **NotificationsGateway/Configuration/JMSSubscription/Attributes**

JMSSubscriptions MessageStats Parameters

This section describes the available JMSSubscription MessageStats Parameters.

MessageSelector

Description: The JMS message selector string.

NGW uses the selector string to subscribe to the JMS Topic. The string is automatically generated from the XSL transform files specified in **config.xml** and cannot be regenerated at runtime (meaning that you cannot subscribe to a new notification type at runtime).

Getter/Setter: G

Type: String

Permissible Values: OR separated NOTIFICATION_TYPE String assignments

Default: Generated from **config.xml** TRANSFORM → DIR → FILE values

MsgConsumed_runningTotal

Description: Running total of consumed JMS messages.

A message (and all it's antecedents) is considered “consumed” when the NGW calls the JMS acknowledge() method on the message. The acknowledge() method is only called after:

1. The XML has been transformed into SOAP.
2. The SOAP has been sent to the SLC OSD interface.
3. A SOAP response has been received.
4. The throttling period specified by the `_throttle_delay_time_ms` attribute.

Getter/Setter: G

Type: Integer

Permissible Values: Positive integer

Default: N/A

MsgInNewTimeAverage_ms

Description: A measure of the time elapsed between ECE Weblogic JMS message creation and NGW message reception.

Weblogic inserts a timestamp in the JMS message header. On message reception, NGW calculates the time elapsed and maintains a 100 message moving average.

Getter/Setter: G

Type: Integer

Permissible Values: Positive integer

Default: N/A

MsgInNew_runningTotal

Description: Running total of new JMS messages. A new message is one in which the **redelivered** field in the JMS message header is set to false.

Getter/Setter: G

Type: Integer

Permissible Values: Positive integer

Default: N/A

MsgInRejected_runningTotal

Description: Running total of rejected JMS messages.

Rejected messages are those accepted by the message selector, but which have an unhandled version string.

Getter/Setter: G

Type: Integer

Permissible Values: Positive integer

Default: N/A

MsgInResent_runningTotal

Description: Running total of resent JMS messages. A resent message is one in which the **redelivered** field in the JMS message header is set to true.

Getter/Setter: G

Type: Integer

Permissible Values: Positive integer

Default: N/A

MsgOutCommsError_runningTotal

Description: Running total of JMS messages which were subject to a communications error. A communications error can be the result of:

- No response from any WSDL-specified OSD port for the service
- All WSDL-specified service ports marked locally out-of-service by NGW

Getter/Setter: G

Type: Integer

Permissible Values: Positive integer

Default: N/A

MsgOutDocError_runningTotal

Description: Running total of JMS messages which were subject to a document error. A document error can be the result of:

- An XML parsing error
- Invalid XSL transform specification
- SOAP Fault, i.e., message is rejected by OSD Interface on Convergent Charging Controller

Getter/Setter: G

Type: Integer

Permissible Values: Positive integer

Default: N/A

StoreMsgInNew_currentTotal

Description: Current total of stored new JMS messages. A new message is one in which the **redelivered** field in the JMS message header is set to false.

Getter/Setter: G

Type: Integer

Permissible Values: Positive integer

Default: N/A

StoredMsgInRejected_currentTotal

Description: Current total of stored rejected JMS messages. Rejected messages are those accepted by the message selector, but which have an unhandled version string.

Getter/Setter: G

Type: Integer

Permissible Values: Positive integer

Default: N/A

StoredMsgInResent_currentTotal

Description: Current total of stored resent JMS messages. A resent message is one in which the **redelivered** field in the JMS message header is set to true.

Getter/Setter: G

Type: Integer

Permissible Values: Positive integer

Default: N/A

StoredMsgOutCommsError_currentTotal

Description: Current total of stored JMS messages which were subject to a communications error. A communications error can be the result of:

- No response from any WSDL-specified OSD port for the service
- All WSDL-specified service ports marked locally out-of-service by NGW

Getter/Setter: G

Type: Integer

Permissible Values: Positive integer

Default: N/A

StoredMsgOutDocError_currentTotal

Description: Current total of stored JMS messages which were subject to a document error. A document error can be the result of:

- An invalid XSL transform specification
- A SOAP fault, such as when a message is rejected by the OSD interface

Getter/Setter: G

Type: Integer

Permissible Values: Positive integer

Default: N/A

JMSSusbscription MessageStats Operations

This section describes the available JMSSubscription MessageStats operations.

dumpStoredMessages

Description: Dumps the contents of one of the in-memory message stores. The messages are written to a file in the directory configured in the `_msg_dump_dir` attribute.

Type: String

Permissible Values Available message types:

- der – Document Errors
- cer – Communications Errors
- new – New
- res – Resent
- rej – Rejected

Return Type: String

Return Value: Success status

retryMsgInRejected

Description: Retries rejected messages by sending all messages currently residing in the rejected message store.

Permissible Values N/A

Return Type: String

Return Value: Success status. http://order-order.com/#_@/ddK1IPV4TNTLFA

retryMsgOutCommsError

Description: Retries communications error messages by sending all messages currently residing in the communications error message store.

Permissible Values N/A

Return Type: String

Return Value: Success status.

retryMsgOutDocError

Description: Retries document error messages by sending all messages currently residing in the document error message store.

Permissible Values N/A

Return Type: String

Return Value: Success status.

`performSubscriptionOperation`

Description: Performs the specified JMS subscription operation.

Type: String

Permissible Values Available Operations:

- init – Initialise JMS connection
- start – Start subscription
- close – Close subscription
- recover – Recover session
- [unsubscribe – Unsubscribe from topic]
- finish – Close JMS connection

Return Type: String

Return Value: Success status.

`showLatestStoredMessage`

Description: Displays the last stored message of one of the in-memory message stores.

Type: String

Permissible Values Available Operations:

- init – Initialise JMS connection
- start – Start subscription
- close – Close
- recover – Recover session
- [unsubscribe – Unsubscribe from topic]
- finish – Close JMS connection

Return Type: String Array

Return Value: One of the following:

- JMS Message
- Failure status

Viewing JMS Message Statistics

Follow these steps to view JMS Message Statistics in NGW:

Step	Action
1	Expand the **NotificationsGateway** navigation tree on the **MBeans** tab in JConsole.
2	Click **JMSSubscription**.
3	Expand **MessageStats**.
4	Click **Attributes** to view the list of JMS message statistics and their values. You cannot modify them. They are read only.

Viewing XML Transform Statistics

Follow these steps to view the XML Transform Statistics:

Step	Action
1	Expand the **NotificationsGateway** navigation tree on the **MBeans** tab in JConsole.
2	Click **XMLTransform**.
3	Expand **XML->SOAP transformation**.

Step	Action
4	Click **Attributes** to view the list of XML transform statistics and their values. You cannot modify them. They are read only.

Chapter 7

Notification Gateway Errors

Overview

Introduction

This chapter describes the Oracle Communications Convergent Charging Controller Notification Gateway errors that might occur while configuring the Notification Gateway.

In this chapter

This chapter contains the following topics.

About Notification Gateway Errors

NGW Error List

This section lists the NGW errors and their descriptions.

General Configuration Errors

The following list describes general configuration errors.

Error	Java Exception	Error Log Message	Description
[0]	Exception	...	Catch-all default message for unknown errors.
[1]	TransformerException, TransformerFactoryConfigurationError	"Exception:" <variable text>	Error creating runtime configuration XML document.
[2]	IOException	"File not saved: Exception:"<variable text>	Configuration XML document file not saved.
[3]	No java exception	"Invalid # of " ... "children specified in " <variable text>	Invalid **config.xml** structure
[4]	No java exception	ERROR: Could not create service	Error creating the web service from config.xml settings.
[5]	Error creating the web service from config.xml settings	Invalid port number in DBID parameter	DBID in **config.xml** is not a valid number.
[6]	N/A	Could not connect to database.	Unable to connect to database for wallet validation.
[7]	SQLException	...	SQL Error encountered whilst retrieving credentials from database.
[8]	N/A	No xlsDirectory specified in TRANSFORM section in <variable text>	No transform directory specified in **config.xml**

Error	Java Exception	Error Log Message	Description
[9]	N/A	"No FILEs specified in DIR section in " <variable text>	No transform files specified in **config.xml**

Run-Time Transform Configuration Errors

The following list describes the run-time Transform configuration errors:

Error	Java Exception	Error Log Message	Description
[1]	Exception	MBean registration failed:	Error registering MBean – JConsole access will not be available.
[2]	TransformerConfigurationException	TransformerConfigurationException in configure(): <variable text>	Could not create XSL template from transform specified in file.
[3]	N/A	Invalid fileName: "... " Format should be: NOTIFICATION_TYPE[.VERSION].XSL	XSL filename is incorrectly formatted.
[4]	Exception	"Exception in parseFilename() :" ...	Unknown exception parsing XSL file.
[5]	TransformerConfigurationException	TransformerConfigurationException in cacheInsert(): "...	Error while attempting to cache transform in memory.

Run-Time WebServices Configuration Errors

The following list describes the run-time WebServices configuration errors:

Error	Java Exception	Error Log Message	Description
[1]	Exception	MBean registration failed:	Error registering MBean – JConsole access will not be available.
[2]	Exception	"Exception in removeService(): " ... <variable text>	Service could not be deregistered.
[3]	N/A	No running services found!	No web services are active.
[5]	TransformerConfigurationException	TransformerConfigurationException in cacheInsert(): "...	Error while attempting to cache transform in memory.

Run-Time JMS Configuration Errors

The following list describes the JMS configuration errors:

Error	Java Exception	Error Log Message	Description
[1]	Exception	MBean registration failed:	Error registering MBean – JConsole access will not be available.

Run-Time Transform Errors

The following list describes the run-time Transform errors:

Error	Java Exception	Error Log Message	Description
[1]	Exception	MBean registration failed:	Error registering MBean – JConsole access will not be available.
[2]	TransformerException	TransformerException in transformXML(): "...	A transformation exception occurred during XML transform.
[3]	Exception	"Exception in transformXML(): "...	An unknown error occurred during XML transform.
[4]	TransformerException	TransformerException in transformXMLFile(): "...	A transformation exception occurred during XML file transform.

Run-Time WebService Errors

The following list describes the run-time WebService errors

Step	Java Exception	Error Log Message	Description
[1]	Exception	MBean registration failed:	Error registering MBean – JConsole access will not be available.
[2]	N/A	"Service:" ... "has no ports in service."	The indicated web service has no currently active ports. The indicated web service has no currently active ports.
[3]	javax.xml.ws.soap.SOAPFaultException	"SOAP exception:" ...	A SOAP exception occurred when attempting to invoke the service.
[4]	javax.xml.stream.XMLStreamException	"XMLStreamReaderException:" ...	An XML parsing exception occurred when trying to invoke the service.
[5]	javax.xml.ws.WebServiceException	"WebServiceException:" ...	A communications exception occurred when trying to invoke the service.
[6]	TransformerException	TransformerException	A XML parsing exception occurred when parsing the SOAP response from the service.
[7]	Exception	"Exception in WebService(): "...	The service has an invalid WSDL URL.
[8]	N/A	"Error: Could not create service: url or operation not defined."	The service URL and/or Operation has not been defined.
[9]	Exception	"createService() failed: "...	Unknown exception when trying to create the service.
[10]	Exception	"Exception in setPortServiceStatus(): "...	"Exception in setPortServiceStatus(): "...

Run-Time JMS Errors

The following list describes the run-time JMS errors.

Error	Java Exception	Error Log Message	Description
[1]	Exception	MBean registration failed:	Error registering MBean – JConsole access will not be available.
[2]	JMSException	"JMS exception in messageToString(): “...	Could not retrieve XML string from JMS message.
[3]	IOSException	"Exception in dumpMessages(): “...	Could not write JMS messages to file.
[4]	JMSException	"JMS exception in doStats(): " ... "Could not get timestamp."	Could not retrieve timestamp from JMS message.
[5]	N/A	"onException(): “... " : " ... "ErrorCode: " ...	Exception during communication with WebLogic server.
[6]	JMSException	"JMSException in processWsResult(): “... " : " ...	Could not retrieve message ID from JMS message.
[7]	N/A	"Encountered SOAP failure sending notification."	A SOAP error occurred sending the SOAP message to the SLC.
[8]	N/A	Encountered communications failure sending notification.	A communications error occurred sending the SOAP message to the SLC.
[9]	JMSException	"JMSException in onMessage(): " ... + “:”	An unknown error occurred whilst processing the JMS message.
[10]	NamingException	"JNDI url lookup failed in init(): “...	Could not lookup JMS provider URL.
[11]	Exception	"Exception in init(): "	Unknown exception initializing JMS Topic reader.
[12]	NamingException	"Naming exception:" ...	Could not connect to JMS topic.
[13]	NamingException	"JNDI API lookup failed: "...	Could not perform JMS binding.
[14]	Exception	"connect(): "...	Unknown error connecting to JMS topic.
[15]	JMSException	"JMSException in startSubscriber(): "...	Could not start JMS subscription.
[16]	Exception	"Exception in startSubscriber(): "...	Could not start JMS subscription.
[17]	JMSException	"Exception occurred:" ...	Could not close JMS subscription
[18]	JMSException	"Exception occurred:" ...	Could not unsubscribe from JMS topic.
[19]	JMSException	"Exception occurred:" ...	Could not close connection to WebLogic.
[20]	JMSException	"Exception occurred:" ...	Could not recover the session.

Chapter 8

Installed Components

Overview

Introduction

This chapter provides information about the installed components for the Oracle Communications Convergent Charging Controller application described in this guide. It also lists the files installed by the application that you can check for, to ensure that the application installed successfully.

In this Chapter

This chapter contains the following topics.

Installation Overview

Introduction

For information about the following requirements and tasks, see *Installation Guide*:

- Convergent Charging Controller system requirements
- Pre-installation tasks
- Installing and removing Convergent Charging Controller packages

Notification Gateway Package

Installing Oracle Communications Convergent Charging Controller Notification Gateway installs the following package on the Service Logic Controller (SLC) node:

- ngwScp

Notification Gateway Directories and Files

The NGW installation on the SLC creates the following directories:

- **/IN/services_packages/NGW/bin**
- **/IN/services_packages/NGW/cpl**
- **/IN/services_packages/NGW/etc**
- **/IN/services_packages/NGW/tmp**

Installing NGW installs the following interface:

- **/IN/services_packages/NGW/bin/ngw**

Installing NGW installs the following configuration file

- **/IN/services_packages/NGW/etc/config.xml**

Installing NGW installs the following sample control plan files in the **/IN/services_packages/NGW/cpl** directory:

- ADVICE_OF_CHARGE_EVENT
- AGGREGATED_THRESHOLD_BREACH_EVENT
- CREDIT_CEILING_BREACH_NOTIFICATION_EVENT
- CREDIT_FLOOR_BREACH_NOTIFICATION_EVENT
- EXTERNAL_TOP_UP_NOTIFICATION_EVENT
- THRESHOLD_BREACH_EVENT
- FIRST_USE_VALIDITY
- OFFERING_VALIDITY_INITIALIZATION

Glossary of Terms

AAA

Authentication, Authorization, and Accounting. Specified in Diameter RFC 3588.

ACS

Advanced Control Services configuration platform.

API

Application Programming Interface

ASP

- Application Service Provider, or
- Application Server Process. An IP based instance of an AS. An ASP implements a SCTP connection between 2 platforms.

CCS

1) Charging Control Services component.

2) Common Channel Signalling. A signalling system used in telephone networks that separates signalling information from user data.

Connection

Transport level link between two peers, providing for multiple sessions.

Convergent

Also “convergent billing”. Describes the scenario where post-paid and pre-paid calls are handed by the same service platform and the same billing system. Under strict converged billing, post-paid subscribers are essentially treated as “limited credit pre-paid”.

CPE

Control Plan Editor (previously Call Plan Editor) - software used to define the logic and data associated with a call -for example, "if the subscriber calls 0800 *nnnnnn* from a phone at location *xxx* then put the call through to *bb bbb bbbb*".

Diameter

A feature rich AAA protocol. Utilises SCTP and TCP transports.

DP

Detection Point

DTMF

Dual Tone Multi-Frequency - system used by touch tone telephones where one high and one low frequency, or tone, is assigned to each touch tone button on the phone.

HTML

HyperText Markup Language, a small application of SGML used on the World Wide Web.

It defines a very simple class of report-style documents, with section headings, paragraphs, lists, tables, and illustrations, with a few informational and presentational items, and some hypertext and multimedia.

IN

Intelligent Network

IP

1) Internet Protocol

2) Intelligent Peripheral - This is a node in an Intelligent Network containing a Specialized Resource Function (SRF).

IP address

Internet Protocol Address - network address of a card on a computer.

MS

Mobile Station

SCP

Service Control Point. Also known as SLC.

SCTP

Stream Control Transmission Protocol. A transport-layer protocol analogous to the TCP or User Datagram Protocol (UDP). SCTP provides some similar services as TCP (reliable, in-sequence transport of messages with congestion control) but adds high availability.

Service Provider

See Telco.

SGML

Standard Generalized Markup Language. The international standard for defining descriptions of the structure of different types of electronic document.

SLC

Service Logic Controller (formerly UAS).

SLEE

Service Logic Execution Environment

SMS

Depending on context, can be:

- Service Management System hardware platform

- Short Message Service
- Service Management System platform
- Convergent Charging Controller Service Management System application

SOAP

Simple Object Access Protocol. An XML-based messaging protocol.

SQL

Structured Query Language is a database query language.

SRF

Specialized Resource Function – This is a node on an IN which can connect to both the SSP and the SLC and delivers additional special resources into the call, mostly related to voice data, for example play voice announcements or collect DTMF tones from the user. Can be present on an SSP or an Intelligent Peripheral (IP).

SSP

Service Switching Point

TCP

Transmission Control Protocol. This is a reliable octet streaming protocol used by the majority of applications on the Internet. It provides a connection-oriented, full-duplex, point to point service between hosts.

Telco

Telecommunications Provider. This is the company that provides the telephone service to customers.

Telecommunications Provider

See Telco.

URI

Uniform Resource Identifier.

URL

Uniform Resource Locator. A standard way of specifying the location of an object, typically a web page, on the Internet.

WSDL

Web Services Description Language.

XML

eXtensible Markup Language. It is designed to improve the functionality of the Web by providing more flexible and adaptable information identification.

It is called extensible because it is not a fixed format like HTML. XML is a `metalanguage' — a language for describing other languages—which lets you design your own customized markup languages for limitless different types of documents. XML can do this because it's written in SGML.

Index

T

U

V

W

X