



Gateways

This chapter introduces gateways, their key features, and how they are used. It also describes the types of gateways, their architecture and components, and how to run them.

It contains the following sections:

- [“Introduction to Gateways”](#)
- [“Types of Gateways”](#)
- [“Gateway Architecture Overview”](#)
- [“Unidirectional and Bidirectional Gateway Architecture”](#)
- [“Gateway Configuration”](#)
- [“Running a Gateway”](#)
- [“Configuring Gateways Interactively”](#)
- [“Gateway Features”](#)
- [“Example Gateway Configuration”](#)
- [“Cisco Info Server Writer”](#)
- [“Conversion Table Utility”](#).

For more information about gateways, including commands common to all gateways, `nco_gate` command line options, and error messages, see [Appendix F, “Gateway Reference.”](#)

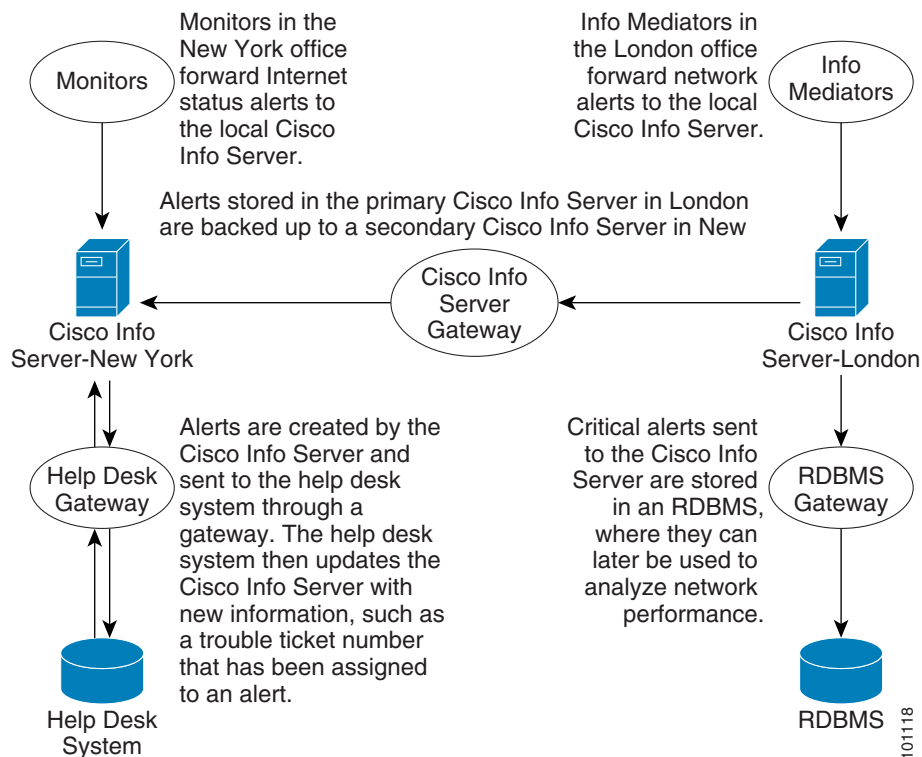
For more information about specific gateways, see the documentation available for each gateway on the Cisco Systems Support web site.

Introduction to Gateways

Cisco Info Center gateways enable you to exchange alerts between Cisco Info Servers and complementary third-party applications, such as databases and Help Desk systems.

You can use gateways to replicate alerts between different NOCs (Network Operations Centers) in your organization or to maintain a backup Cisco Info Server. Application gateways enable you to integrate different business functions. For example, you can integrate your Help Desk and NOC.

[Figure 6-1](#) shows an example gateway architecture.

Figure 6-1 Gateways in the Cisco Info Center Architecture

In this example, gateways are used for a variety of purposes:

- the Cisco Info Server gateway enables you to back up alerts between different Cisco Info Center installations
- the Relational Database Management System (RDBMS) gateway enables you to store critical alerts so you can analyze network performance
- the Help Desk gateway enables you to integrate the NOC and the Help Desk by converting trouble tickets to alerts and alerts to trouble tickets.

Once a gateway is correctly installed and configured, the transfer of alerts is transparent to operators. For example, Cisco Info Server alerts forwarded from one Cisco Info Server to another are processed and displayed in the Event List in the same way as events received from a Cisco Info Mediator.

Types of Gateways

The types of gateways are:

- Cisco Info Server
- Database
- Help Desk
- Other.

Cisco Info Server gateways are used to exchange events between Cisco Info Servers. This is useful when you want to create a distributed Cisco Info Center installation, or when you want to install a backup Cisco Info Server.

Database gateways are used to store events from a Cisco Info Server. This is useful when you want to keep a historical record of the events forwarded to the Cisco Info Server.

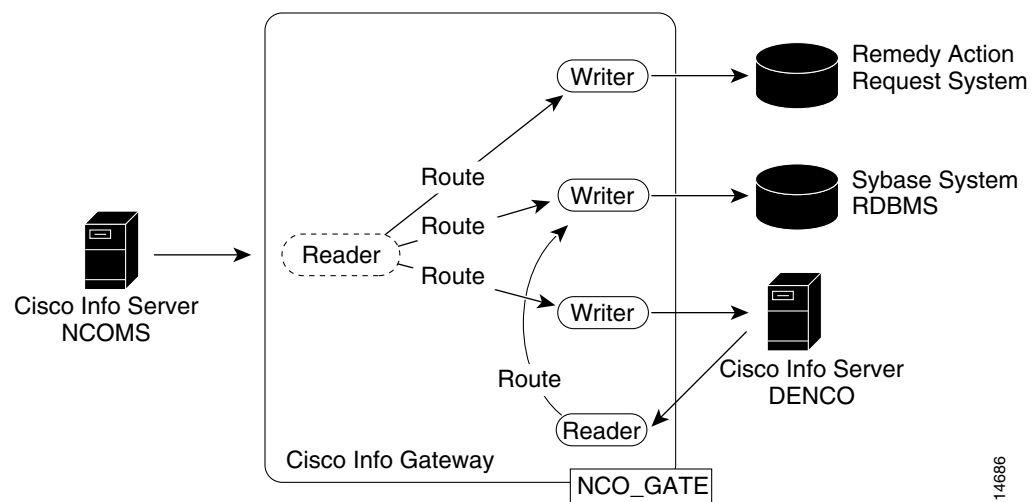
Help Desk gateways are used to integrate Cisco Info Center with a range of Help Desk systems. This is useful when you want to correlate the trouble-tickets raised by your customers with the networks and systems you are using to provide their services.

Other gateways are specialized applications that forward Cisco Info Server alerts to other applications or devices (for example, a flat file or socket).

Gateway Architecture Overview

In the example architecture in [Figure 6-2](#), a source Cisco Info Server (NCOMS) is generating trouble tickets for a Remedy ARS, sending alerts to a Sybase database for analysis, and replicating alerts in the DENCO Cisco Info Server. The DENCO Cisco Info Server is also sending alerts to the Sybase database for analysis.

Figure 6-2 Example Gateway Architecture



Gateways have Reader and Writer components. Readers extract alerts from the Cisco Info Server. Writers forward alerts to another Cisco Info Server or to other applications. There is only one type of Reader, but there are various types of Writers depending on the destination application.

Routes specify the destination to which a Reader forwards alerts. One Reader can have multiple routes to different Writers, and one Writer can have multiple routes from different Readers.

Gateway filters and mappings configure event flow. Filters define the types of events that can be passed through a gateway. Mappings define the format of these events.

Readers, Writers, routes, filters, and mappings are defined in the gateway configuration file, described in [Gateway Configuration, page 6-9](#).

Unidirectional and Bidirectional Gateway Architecture

Gateways can be unidirectional or bidirectional. In a unidirectional gateway, alerts flow from the source to the destination (for example, from a Cisco Info Server to a database). In a bidirectional gateway, alerts flow in both directions, ensuring the Cisco Info Server and the target application contain the same alerts.



Note

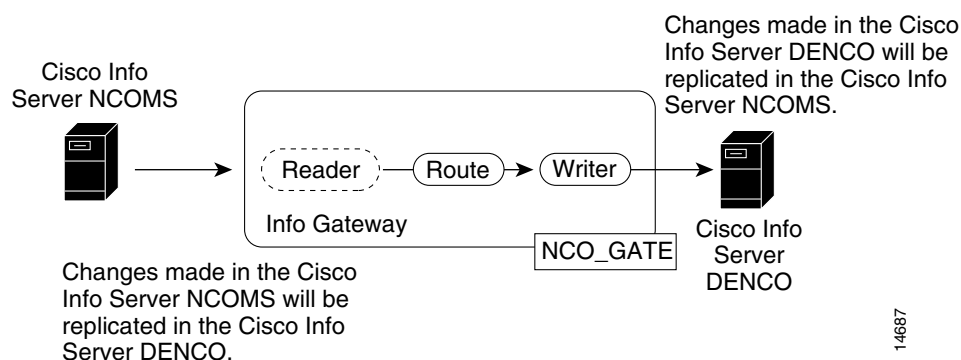
Only certain gateways can be bidirectional. See the documentation available for each gateway on the Cisco Systems Support Site for more information.

Unidirectional Gateways

A unidirectional gateway is a gateway that only allows alerts to flow in one direction. Changes made in the source Cisco Info Server are replicated in the destination Cisco Info Server or application, but changes made in the destination Cisco Info Server or application are not replicated in the source Cisco Info Server.

Figure 6-3 shows the configuration of a unidirectional Cisco Info Gateway.

Figure 6-3 Unidirectional Cisco Info Server Gateway



A unidirectional Cisco Info Gateway requires a Reader, a route, a mapping, and a Writer.

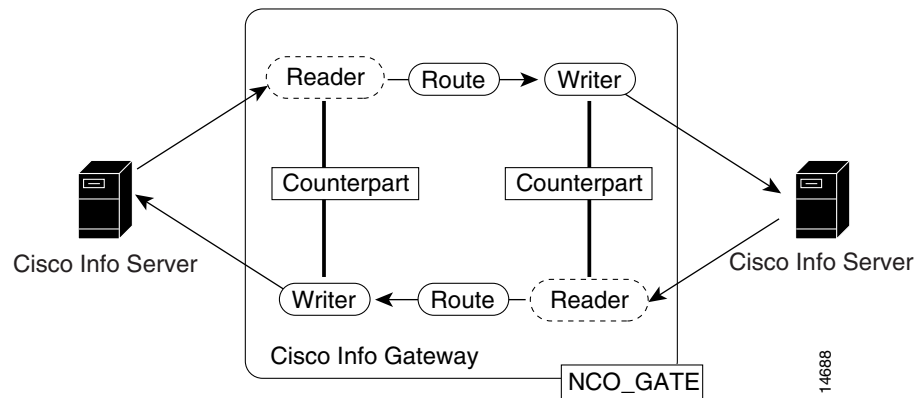
Bidirectional Gateways

In a bidirectional gateway configuration, changes made to the contents of a source Cisco Info Server are replicated in a destination Cisco Info Server or application, and the destination Cisco Info Server or application replicates its alerts in the source Cisco Info Server. This enables you, for example, to maintain a system with two Cisco Info Servers on different machines that can continue to run should one of them fail.

Bidirectional gateways have a similar configuration to unidirectional gateways, with an additional **COUNTERPART** attribute for the Writers. The **COUNTERPART** attribute defines a link between a gateway's Writer and Reader.

The gateway interprets a Reader and Writer connected by a counterpart as one half of a bidirectional gateway. Figure 6-4 shows an example bidirectional Cisco Info Server gateway configuration

Figure 6-4 Bidirectional Cisco Info Server Gateway



The Readers and routes shown in this example are configured in the same way as for a unidirectional gateway. However, the Writers are slightly different, because they define the counterparts.

The Writers for this example are created in the configuration file as follows:

```
START WRITER DENCO_WRITER
(
    TYPE = OBJECT_SERVER,
    REVISION = 1,
    SERVER = DENCO,
    MAP = SERVER_MAP,
    COUNTERPART = DENCO_READER
);
START WRITER NCOMS_WRITER
(
    TYPE = OBJECT_SERVER,
    REVISION = 1,
    SERVER = NCOMS,
    MAP = SERVER_MAP,
    COUNTERPART = NCOMS_READER
);
```

The counterpart prevents the Reader from detecting each alert as a new one by disabling IDUC for the alert. IDUC is described in [Client Tool Updates Using IDUC, page 1-14](#).



Note

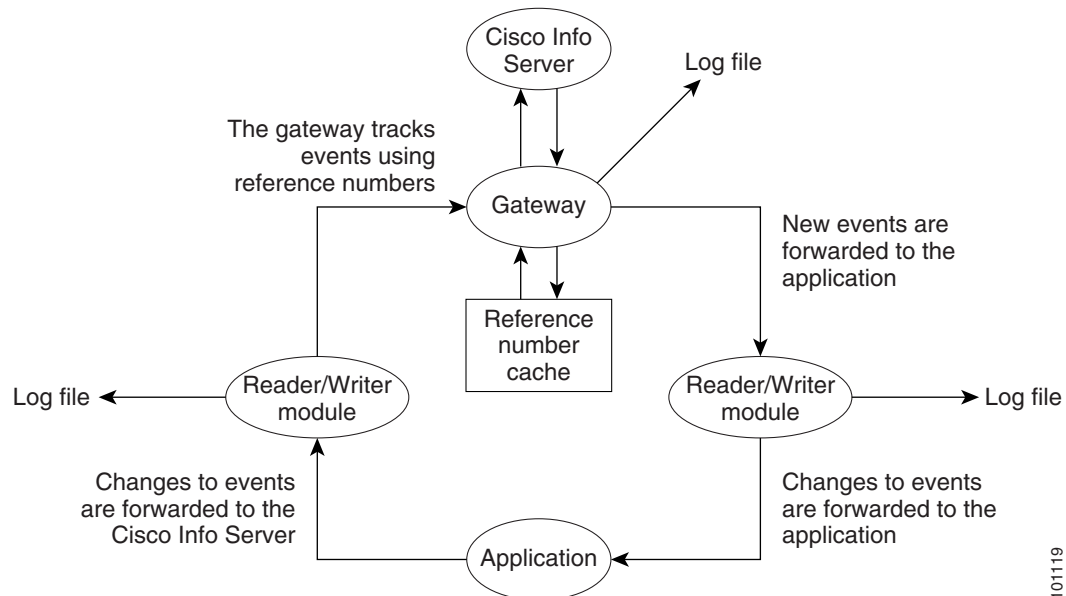
You must use counterparts when linking Cisco Info Servers together. Otherwise, you will create two unidirectional gateways that are not related to each other and alerts received by each Cisco Info Server will be automatically forwarded to the other. The other Cisco Info Server will detect it has received a new alert and forward it back. This will create an endless loop of alerts between the Cisco Info Servers.

The Reader/Writer Module

The Reader/Writer module manages communications between gateways and third-party applications.

Figure 6-5 shows a simplified example of the Reader/Writer module architecture.

Figure 6-5 Reader/Writer Module Architecture



The Reader/Writer module generates log files which can help you debug the gateway. The log files are described in [Gateway Log Files](#), page 6-17.

The gateway uses a reference number cache to track the alerts and their associated reference in the target application (for example, Clarify Cases or ServiceCenter Tickets).

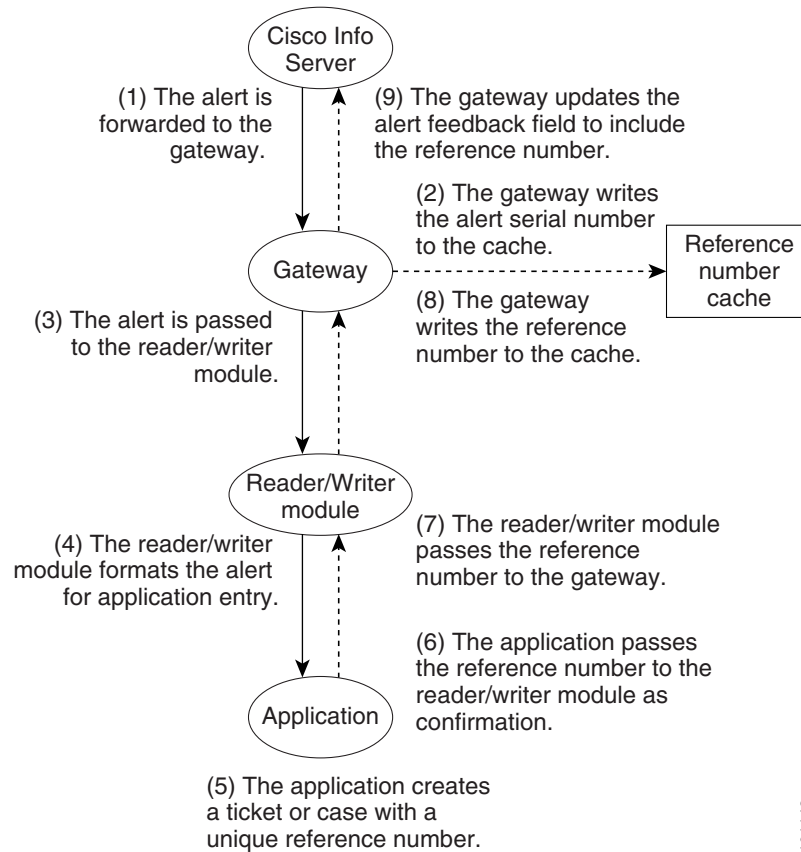
Reference Number Caching in Help Desk Gateways

Help Desk gateways track alerts and trouble tickets by maintaining a cache of reference numbers. For each alert, the cache stores the following numbers:

- the serial number of the alert
- a reference number from the target application (for example, Clarify Cases or ServiceCenter Tickets).

Figure 6-6 shows how these references are created.

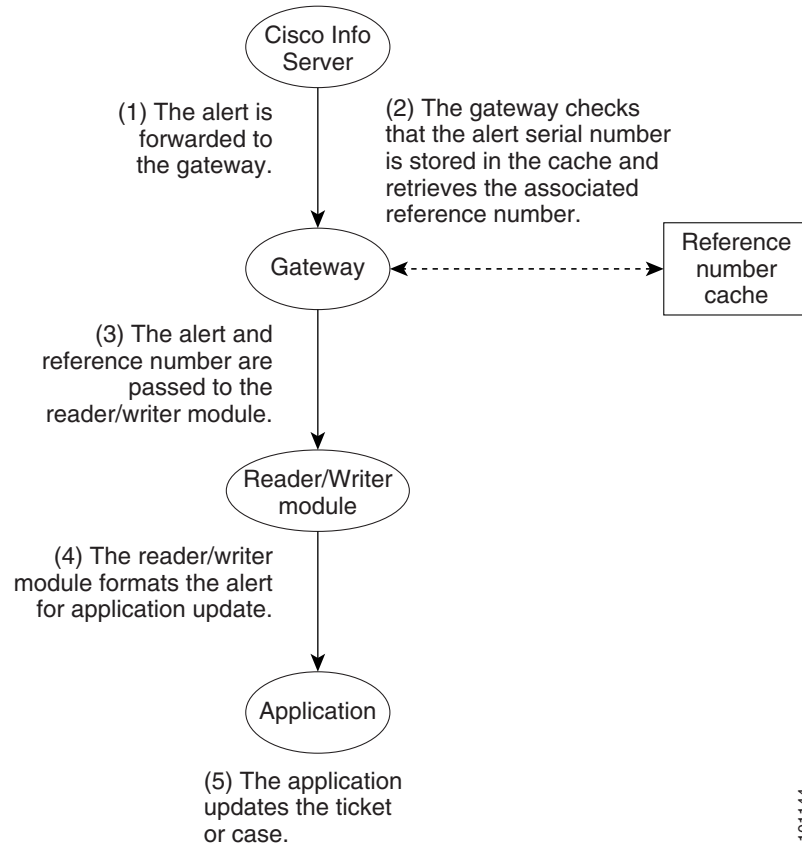
Figure 6-6 Reference Number Cache Creation in a Help Desk Gateway



The gateway and the Reader/Writer module perform this sequence of actions each time they receive a new alert from the Cisco Info Server. If an alert is updated in the Cisco Info Server, the updates are forwarded to the gateway.

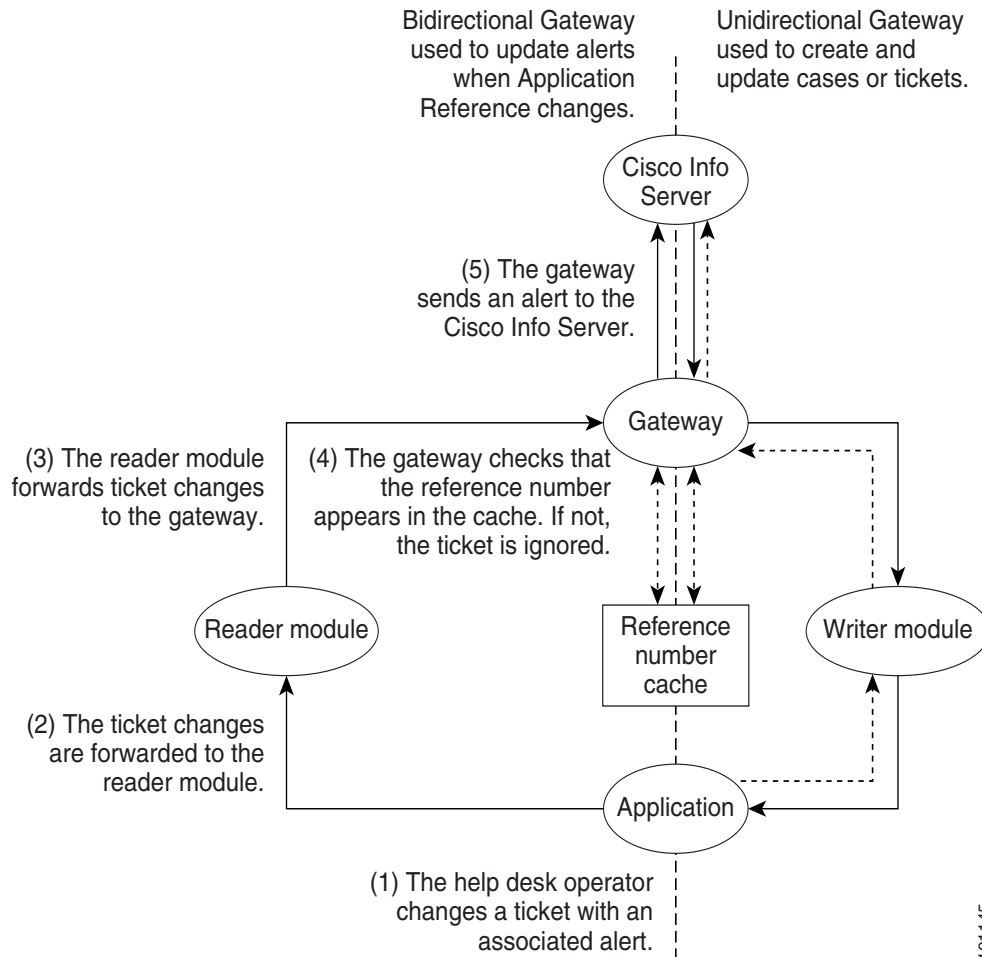
Figure 6-7 shows how changes to an alert are reflected in the Help Desk application. For example, if the alert is deleted through the Event List, the associated ticket or case is closed in the Help Desk application.

Figure 6-7 Processing an Updated Alert in a Help Desk Gateway



Some Help Desk gateways are bidirectional; that is, changes made to a trouble ticket or case are reflected in the alert. For example, if a Help Desk operator updates a Clarify case associated with an alert, the alert is also updated. To do this, bidirectional gateways use a second Reader/Writer module, as shown in [Figure 6-8](#).

Figure 6-8 Bidirectional Alert Updates in a Help Desk Gateway



The Reader/Writer module used to handle bidirectional alert updates is not the same Reader/Writer module used to create and update tickets or cases as shown in [Figure 6-6](#) and [Figure 6-7](#).



Note

The Help Desk application can only update alerts created by the gateway. It cannot generate new alerts within Cisco Info Center.

Gateway Configuration

This section describes the gateway configuration file and the commands used in it to define the operation of the gateway. The configuration file defines the operation of the gateway using:

- Readers
- Writers
- Routes

- Mappings
- Filters.

The Gateway Configuration File

Every gateway has a configuration file, with the extension **.conf**. The default gateway configuration file is:

\$OMNIHOME/etc/NCO_GATE.conf

Use the **-config** command line option to specify the full path and file name of an alternate configuration file. For example, to run a Help Desk gateway with a configuration file named **HDESK.conf**, enter:

\$OMNIHOME/bin/nco_gate -config \$OMNIHOME/etc/HDESK.conf

When a gateway is started, it processes the commands in the configuration file. This defines the connections between the source Cisco Info Server and the alert destinations.

Reader Commands

A Reader extracts alerts from a Cisco Info Server. There is only one type of Reader—the Cisco Info Server Reader. Readers are created using the **START READER** command, which defines the name of the Reader and the name of the Cisco Info Server from which to read.

For example, to create the Reader for the **NCOMS** Cisco Info Server shown in [Figure 6-2 on page 6-3](#), add the following command to the configuration file:

START READER NCOMS_READ CONNECT TO NCOMS;

Once this command has been issued, the Reader is started and the gateway attempts to open a connection to the source Cisco Info Server. If the gateway succeeds in opening the connection, it immediately starts to read alerts from the Cisco Info Server. For the Reader to forward these alerts to their destination, you must define an associated route and Writer.

The **START READER** command is described in more detail in [Reader Commands, page F-3](#).

Writer Commands

Writers send the alerts acquired by a Reader to the destination application or Cisco Info Server. Writers are created using the **START WRITER** command, which defines the name of the Writer and the information allowing it to connect to its destination.

For example, to create the Writer for the Remedy Action Request System (ARS) shown in [Figure 6-2 on page 6-3](#), add the following command to the configuration file:

```
START WRITER ARS_WRITER
(
    TYPE = ARS,
    REVISION = 1,
    MAP = ARS_MAP,
    SERVER = 'orac',
    SCHEMA = 'Alerts 3.6',
    USER = 'Demo',
    PASSWORD = '',
    FEEDBACK = TRUE,
    FEEDBACK_SERVER = NCOMS,
    FEEDBACK_FIELD = 'Location',
    JOURNAL = 536870919
);
```

Once this command has been issued, the gateway attempts to establish the connection to the alert destination (either an application or another Cisco Info Server). The Writer sends alerts received from the source Cisco Info Server until the **STOP WRITER** command is issued.

The **START WRITER** command is described in more detail in [Writer Commands, page F-5](#). The Cisco Info Server Writer is described in more detail in [Cisco Info Server Writer, page 6-30](#).

Route Commands

Routes create the link between Readers and Writers. Routes are created using the **ADD ROUTE** command. This command defines the name of the route, the source Reader, and the destination Writer.

For example, to create the route between the NCOMS Cisco Info Server Reader and the Remedy ARS Writer shown in [Figure 6-2 on page 6-3](#), add the following command to the configuration file:

```
ADD ROUTE FROM NCOMS_READ TO ARS_WRITER;
```

Once this command is issued, the connection between a Reader and Writer is established. Any alerts received by the source Cisco Info Server are read by the Reader, passed through the route to the Writer, and written into the destination Cisco Info Server or application.

The **ADD ROUTE** command is described in more detail in [Route Commands, page F-12](#).

Mapping Commands

Mappings define how alerts received from the source Cisco Info Server should be written to the destination Cisco Info Server or application. Each Writer has a different mapping which is defined using the

CREATE MAPPING command.

For example, to create the mapping between the Cisco Info Server Reader and the Remedy ARS Writer shown in [Figure 6-2 on page 6-3](#), add the following command to the configuration file:

```
CREATE MAPPING ARS_MAP
(
    536870914 = '@Identifier'           ON INSERT ONLY,
    536870913 = '@Serial'             ON INSERT ONLY,
    536870944 = '@Node'              ON INSERT ONLY,
    536870915 = '@NodeAlias'         ON INSERT ONLY,
    536870932 = '@Manager'           ON INSERT ONLY,
    536870939 = '@Agent'             ON INSERT ONLY,
    536870918 = '@AlertGroup'        ON INSERT ONLY,
    536870931 = '@AlertKey'          ON INSERT ONLY,
    7 = '@Severity',
    536870916 = '@Summary'           ON INSERT ONLY,
    536870946 = '@StateChange',
    536870912 = '@FirstOccurrence'   ON INSERT ONLY,
    536870917 = '@LastOccurrence',
    536870935 = '@InternalLast',
    536870938 = '@Poll'              ON INSERT ONLY,
    536870941 = '@Type'              ON INSERT ONLY,
    536870940 = '@Tally',
    536870945 = '@Class'             ON INSERT ONLY,
    536870942 = '@Grade'            ON INSERT ONLY,
    536870934 = '@Location'          ON INSERT ONLY,
    536870933 = '@Acknowledged'
);
```

In this example, the mapping name is **ARS_MAP**.

Each line between the parentheses defines how the fields in ARS receive data from Cisco Info Server fields. The alert fields from the source Cisco Info Server are represented by the @ symbol. These are mapped to ARS fields, which are identified with long integer values rather than field names. For example, the following line specifies the ARS field **536870913** maps to the **Serial** field from the Cisco Info Server:

536870913 = '@Serial' ON INSERT ONLY,

The **ON INSERT ONLY** clause controls when the field is updated. Fields with the **ON INSERT ONLY** clause are only forwarded once, when the alert is created for the first time in the Cisco Info Server. Fields not having the **ON INSERT ONLY** clause are updated each time the alert changes.

The **CREATE MAPPING** command is described in more detail in [Mapping Commands, page F-8](#).

Filter Commands

You may not always want to send all of the alerts read by a Reader to the destination Cisco Info Server or application. For example, you may only want to send alerts having a critical severity level. Filters define which of the alerts read by the Cisco Info Server Reader should be forwarded to the destination.

You create filters using the **CREATE FILTER** command and apply them using the **START READER** command. For example, to create a filter that only forwards critical alerts to the destination application or Cisco Info Server, add the following command to the configuration file:

```
CREATE FILTER CRITONLY AS 'Severity = 5';
```

This command creates a filter named **CRITONLY**, which only forwards alerts with a critical severity level (5).



Note

To perform string comparisons with filters, you must escape the quotes in the **CREATE FILTER** command with back slashes. For example, to create a filter that only forwards alerts from a node called **fred**, the **CREATE FILTER** command is:

```
CREATE FILTER FREDONLY AS 'NODE = \'fred\'';
```

To apply the filter to the Cisco Info Server Reader shown in [Figure 6-2 on page 6-3](#), add the following command to the configuration file:

```
START READER NCOMS_READ CONNECT TO NCOMS USING FILTER CRITONLY;
```

The **CREATE FILTER** command is described in more detail in [Filter Commands, page F-10](#).

Creating Multiple Filters and Multiple Readers

When you need more than one filter for the same Cisco Info Server, you can create multiple Readers for it. For example, to create a Reader that forwards all critical alerts and another that forwards everything else, use the following commands:

```
CREATE FILTER CRITONLY AS 'Severity = 5';
```

```
CREATE FILTER THEREST AS 'Severity <5';
```

```
START READER CRIT_NCOMS CONNECT TO NCOMS USING FILTER CRITONLY;
```

```
START READER REST_NCOMS CONNECT TO NCOMS USING FILTER THEREST;
```

Loading Filters Created Using Filter Builder

You can load and use filters created in the Filter Builder. For example:

```
LOAD FILTER FROM '/usr/filters/myfilt.elf';
```

This command loads the `/usr/filters/myfilt.elf` file as a filter. This filter name is defined by the **Filter Builder Name** field.



Note

The **Name** field must be alphabetical and must not contain spaces.

For more information about the Filter Builder, see the *Cisco Info Center User Guide, 3.6*.

Running a Gateway

A gateway requires an entry in the Server Editor, as described in the *Cisco Info Center Installation and Configuration Guide*, 3.6.

You must also create your configuration file, described in [Gateway Configuration](#), page 6-9.

Once you define the gateway communications and create your configuration file, you can run the gateway.

Starting a Gateway

To run a gateway with a default configuration, enter:

```
$OMNIHOME/bin/nco_gate
```

This runs a gateway with the default name **NCO_GATE** and the default configuration file **\$OMNIHOME/etc/NCO_GATE.conf**

To run a gateway with your own configuration, use command line options, as described in [Gateway Command Line Options](#), page F-2. For example:

```
$OMNIHOME/bin/nco_gate -name ORA1 -config $OMNIHOME/etc/RDBMS.conf
```

This runs a gateway named **ORA1** using a configuration file named **RDBMS.conf**

Gateways should be configured to run under Process Control. Process Control is described in [Chapter 7](#), “Process Control.”

Configuring Gateways Interactively

You can change the configuration of a gateway while it is running using the SQL interactive interface. The SQL interactive interface is described in [Direct Access Using the SQL Interactive Interface \(nco_sql\)](#), page 2-4.



Note

If you are running a gateway, you must be a member of the user group **ncoadmin** to log into a gateway. You may need to ask your systems administrator to create this group.

Use the SQL interactive interface to connect to a gateway as a specific user. Following is an example:

```
$OMNIHOME/bin/nco_sql -server <gateway_name> -user <username>
```

where *<gateway_name>* is the name of the gateway and *<username>* is a valid user name. If you do not specify a user name, the default is the user running the command.

You are prompted to enter a password (the default is to enter your UNIX password). To authenticate users using other methods, use the **-authenticate** command line option, described in [Gateway Command Line Options](#), page F-2.

After connecting with a user name and password, a numbered prompt is displayed.

```
1>
```

You can enter commands to configure the gateway dynamically. The following example shows a session in which a new route is added:

```
$ nco_sql -server NCO_GATE
Password:
User 'admin' logged in.
1> ADD ROUTE FROM DENCO_READ TO ARS_WRITER;
2> ADD ROUTE FROM DENCO_READ TO OS_WRITER;
3> go

1>
```

If you want to disable interactive configuration, add the following line to the end of the gateway configuration file:

```
SET CONNECTIONS FALSE;
```

Saving Configurations Interactively

You can save the interactive gateway configuration with the command:

```
SAVE CONFIG TO '<filename>';
```

where *<filename>* is the name of a file on a local file system.

You can then use the saved configuration file for other gateways.

Dumping and Loading Gateway Configurations Interactively

You can load gateway configurations interactively.

First stop any running Readers and Writers manually with the **STOP** command. Then use the **DUMP CONFIG** command to discard the current configuration.

The **DUMP CONFIG** command will not discard the configuration if any Readers and Writers are running or if the configuration has been changed interactively, unless you use the **FORCE** option. To determine if the configuration has been changed interactively, use the **SHOW SYSTEM** command, described in [SHOW SYSTEM, page F-15](#).

See [Configuration Commands, page F-13](#) for more information.

Once you have dumped the configuration, you can load a new configuration with the command:

```
LOAD CONFIG FROM '<filename>';
```

where *<filename>* is the name of a file on a local file system.

Gateway Features

This section describes some of the key features of gateway operation.

Store and Forward Mode

If there is a problem with the gateway destination, the Cisco Info Server and database Writers can continue to run using Store and Forward mode.

When the Writer detects the Cisco Info Server or database is not present or is not functioning (usually because it is unable to write an alert), it switches into Store mode. In this mode, the Writer stores everything it would normally send to the database in a file named:

\$OMNIHOME/var/<writername>.<destserver>.store

In this file name, <writername> is the name of the Writer and <destserver> is the name of the server to which the gateway is attempting to send alerts.

When the gateway detects the destination server is back online, it switches into Forward mode and sends the alert information held in the **.store** file to the destination server. Once all the alerts in the **.store** file have been forwarded, the Writer returns to normal operation.

Store and forward mode only works when a connection to the Cisco Info Server or database destination, has been established, used, and then lost. If the destination server is not running when the gateway starts, store and forward mode is not triggered and the gateway terminates.

If the gateway connects to the destination Cisco Info Server and a store and forward file already exists, the gateway replays the contents of the store and forward file before it sends new alerts.

Store and forward mode is configured using the **STORE_AND_FORWARD** and **STORE_FILE** attributes.

**Note**

Store and forward does not work with bidirectional gateway configurations, with the exception of the Cisco Info Server bidirectional gateway.

Secure Mode

You can run the Cisco Info Server in Secure mode. When you specify the **-secure** command line option, the Cisco Info Server authenticates Cisco Info Mediator, gateway, and proxy server connection requests by requiring a user name and an encrypted password. When a connection request is sent, the Cisco Info Server issues an authentication message. The Cisco Info Mediator, gateway, or proxy server must respond with the correct user name and password. If the user name and password combination is incorrect, the Cisco Info Server issues an error message and rejects the connection.

If you do not specify the **-secure** option, no security checks are performed on the connection request.

When connecting to a secure Cisco Info Server, the gateway must have the **AUTH_USER** and **AUTH_PASSWORD** commands in the gateway configuration file. You can choose any valid user name for the **AUTH_USER** gateway command. To generate the encrypted **AUTH_PASSWORD**, use the **nco_crypt** command, described in [Running the Cisco Info Server in Secure Mode, page 1-13](#). The command takes the unencrypted password and displays the encrypted password to be entered for the **AUTH_PASSWORD** command.

The **AUTH_USER** and **AUTH_PASSWORD** commands must precede any Reader commands in the gateway configuration file. Enter the user name and corresponding encrypted password in the configuration file and save it before running the gateway.

Encrypting Target System Passwords

The **nco_g_crypt** utility is a two-way encryption tool you can use to encrypt plain text login passwords. The gateways use these encrypted passwords to log into their target systems. Encrypted passwords are decoded by the gateway before they are used to log in to the target system.

The user name and encrypted password are stored in the **USERNAME** and **PASSWORD** attributes in the gateway Writer.



Note

If you are using a Help Desk gateway, substitute **USER** for **USERNAME**.

To encrypt a plain text password for a gateway's target system:

Step 1 Enter the following:

```
$OMNIHOME/bin/nco_g_crypt <password>
```

where *<password>* is the unencrypted form of the password. The **nco_g_crypt** utility displays an encrypted version of the password.

Step 2 Update the gateway Writer in the gateway configuration file by copying the user name into the **USERNAME** attribute value and the encrypted password created in [Step 1](#) into the **PASSWORD** attribute value.

For example:

```
START WRITER SYBASE_WRITER
(
    TYPE = SYBASE,
    REVISION = 1,
    SERVER = DARKSTAR,
    MAP = SYBASE_MAP,
    USER = 'SYSTEM',
    PASSWORD = 'MKFGHIFE',
    FORWARD_DELETES = TRUE
);
```

Step 3 Run the gateway.

Gateway Log Files

The gateway log files are in the **\$OMNIHOME/log** directory. When debugging, you should initially check the following log file:

```
NCO_<GATENAME>.log
```

where *<GATENAME>* is the name of the gateway.

You may receive an error message such as the following:

```
error in srv_select () - file descriptor x is no longer active!
```

This type of error message indicates the gateway has aborted due to one of the Reader/Writer modules failing. In this case check the following log files:

NCO_GATE_<X>RW<Y>_WRITE.log

or

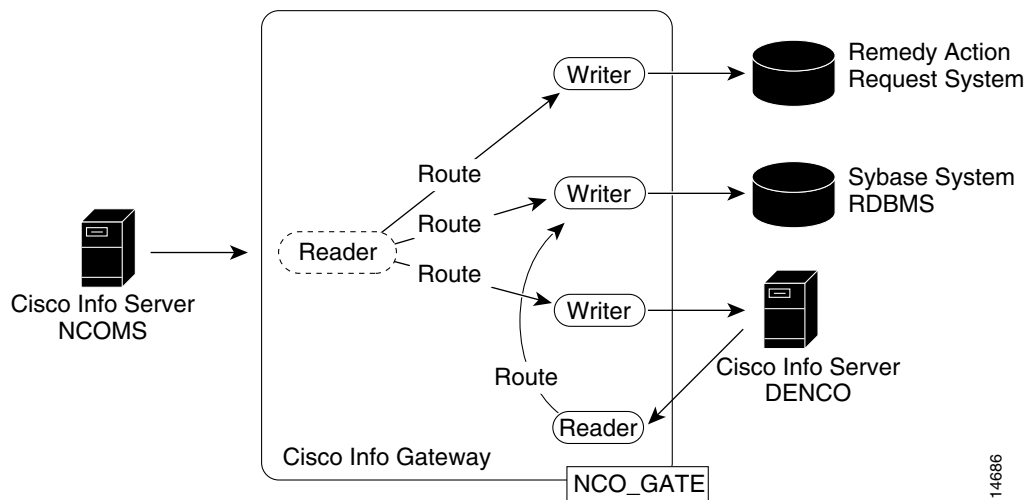
NCO_GATE_<X>RW<Y>_READ.log

where <X> identifies the actual gateway and <Y> identifies the particular version for both files.

Example Gateway Configuration

This section describes how to create the example gateway shown in [Figure 6-9](#).

Figure 6-9 Example Gateway Configuration



For the purpose of this example, the gateway configuration has been split into the following stages:

1. Initializing the Gateway.
2. Creating the Cisco Info Server Reader.
3. Connecting to Remedy ARS by:
 - Creating the Remedy ARS Mapping
 - Creating a Remedy ARS Writer
 - Adding a Remedy ARS Route.
4. Connecting to Sybase by:
 - Creating the Sybase Mapping
 - Creating a Sybase Writer
 - Adding a Sybase Route.

5. Connecting to Another Cisco Info Server by:
 - Creating the Cisco Info Server Mapping
 - Creating a Cisco Info Server Writer
 - Adding a Route to the Cisco Info Server Writer.
6. Sharing a Writer.

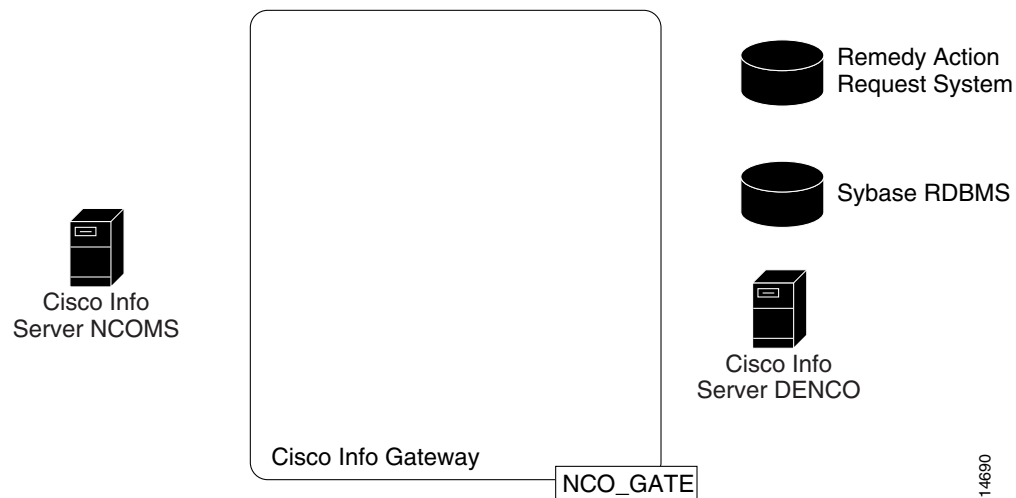
Each stage is described in the following sections.

For the purpose of this example, it is assumed the Cisco Info Servers (**NCOMS** and **DENCO**) and the third-party applications (Remedy ARS and Sybase) are already installed and working correctly. Additional preparation is necessary. For example, you must create gateway entries in the Server Editor. Most gateways also have additional setup requirements, such as database schema changes, which are documented in the individual gateway guides.

Initializing the Gateway

Figure 6-10 shows the system configuration with no gateway connections.

Figure 6-10 Initializing the Gateway



Although the gateway process can be run, it will do nothing until it is issued commands to configure its operation. These commands are typically read from the configuration file at start up. For a gateway named **NCO_GATE**, the configuration file is named:

\$OMNIHOME/etc/NCO_GATE.conf

In this example, it is assumed you already configured the **NCO_GATE** gateway and the **NCOMS** and **DENCO** Cisco Info Servers with the Server Editor (**nco_xigen**). See the *Cisco Info Center Installation and Configuration Guide, 3.6* for more information about configuring Cisco Info Center communications.

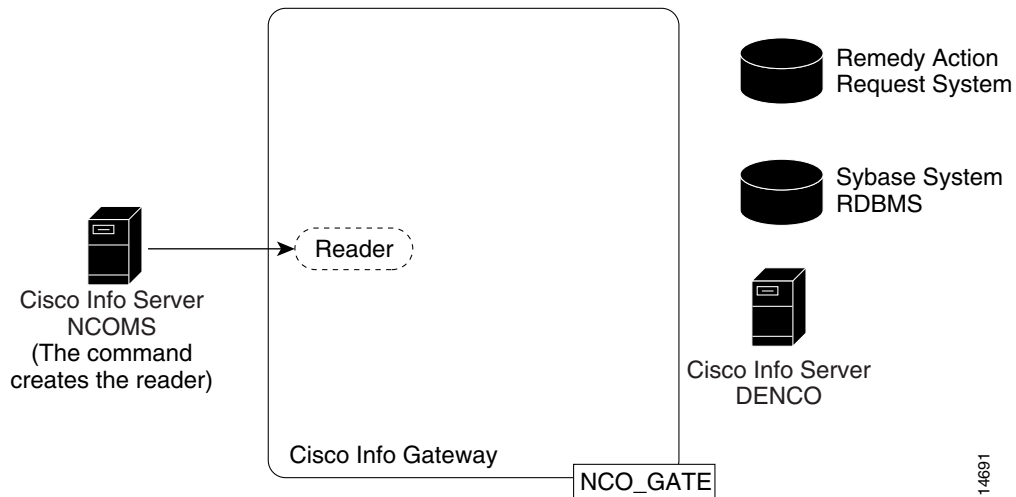
Creating the Cisco Info Server Reader

The first stage for this example is to create a Reader to extract data from the **NCOMS** Cisco Info Server. The following command is the first line of the configuration file:

```
START READER NCOMS_READ CONNECT TO NCOMS;
```

This command creates a Reader called **NCOMS_READ** which connects to **NCOMS**, as shown in Figure 6-11.

Figure 6-11 Adding a Cisco Info Server Reader



The Reader is now reading data from **NCOMS**. However, it does not yet have anywhere to send the data until the corresponding Writers are set up.

The Writers for this example are set up in the following sections.

Connecting to Remedy ARS

This section contains the steps enabling alert data to be sent to the Remedy ARS system.

Creating the Remedy ARS Mapping

Mapping is the mechanism for controlling how a Writer processes information from a Reader. The following command creates a mapping for Remedy ARS:

```
CREATE MAPPING ARS_MAP
(
    536870914 = '@Identifier'          ON INSERT ONLY,
    536870913 = '@Serial'            ON INSERT ONLY,
    536870944 = '@Node'              ON INSERT ONLY,
    536870915 = '@NodeAlias'        ON INSERT ONLY,
    536870932 = '@Manager'          ON INSERT ONLY,
    536870939 = '@Agent'            ON INSERT ONLY,
```

```

536870918 = '@AlertGroup'          ON INSERT ONLY,
536870931 = '@AlertKey'           ON INSERT ONLY,
      7 = '@Severity',
536870916 = '@Summary'           ON INSERT ONLY,
536870946 = '@StateChange',
536870912 = '@FirstOccurrence'   ON INSERT ONLY,
536870917 = '@LastOccurrence',
536870935 = '@InternalLast',
536870938 = '@Poll'             ON INSERT ONLY,
536870941 = '@Type'             ON INSERT ONLY,
536870940 = '@Tally',
536870945 = '@Class'            ON INSERT ONLY,
536870942 = '@Grade'           ON INSERT ONLY,
536870934 = '@Location'        ON INSERT ONLY,
536870933 = '@Acknowledged'
);

```

This mapping requires the Remedy ARS administrator to configure a schema in Remedy ARS to accept this information. If this is not done beforehand, the mapping will not be able to find the specified fields and the gateway will not be able to write to Remedy ARS.

Creating a Remedy ARS Writer

Creating a Writer gives the gateway a way of sending data to another application. There are different types of Writers for each specific application. The type of Writer is determined by the **TYPE** parameter specified when creating the Writer. The following command creates a Remedy ARS Writer:

```

START WRITER ARS_WRITER
(
    TYPE = ARS,
    REVISION = 1,
    MAP = ARS_MAP,
    SERVER = 'orac',
    SCHEMA = 'Omnibus 3.6',
    USER = 'Demo',
    PASSWORD = '',
    FEEDBACK = TRUE,
    FEEDBACK_SERVER = NCOMS,
    FEEDBACK_FIELD = 'Location',
    JOURNAL = 536870919
);

```

After the **TYPE** parameter comes the **REVISION** parameter, which is for setting the version of Writer to use when more than one Writer of a particular type exists. The other parameters are Writer-specific. The following sections describe each line in the order they appear.

This is the beginning of the command to start a Writer called **ARS_WRITER**:

```
START WRITER ARS_WRITER
```

The open parenthesis specifies the start of the parameters for the Writer, followed by the **TYPE** parameter. In this case, it sets the **TYPE** to **ARS** to identify the Writer is for Remedy ARS:

(

TYPE=ARS,

The **REVISION** parameter is for future compatibility to be incremented as new Writers appear. For now, it should be set to 1:

REVISION = 1,

The **MAP** parameter selects which mapping should be used for this Writer:

MAP = ARS_MAP,

The **SERVER** parameter indicates the machine on which the Remedy ARS server can be found. In this case, it is on a machine called **orac**:

SERVER = 'orac',

Remedy ARS supports multiple schemas. The **SCHEMA** parameter specifies which schema it should write to. The schema in Remedy ARS should be created by the ARS administrator with the schema supplied with the gateway:

SCHEMA = 'Omnibus 3.6',

The Writer needs to be able to log on to Remedy ARS to submit and update trouble tickets. The **USER** parameter specifies the user name and the **PASSWORD** parameter specifies the password for the gateway:

USER = 'Demo',

PASSWORD = "",

When a trouble ticket is posted to Remedy ARS, it is given its own trouble ticket number. When looking at an alert in Cisco Info Center, it is useful to know which trouble ticket is being used. The feedback mechanism allows for this. Setting the **FEEDBACK** parameter to **TRUE** instructs the Writer that when a new trouble ticket is written to Remedy ARS, it should take the allocated ticket number and write it back to a Cisco Info Server in a particular field. The **FEEDBACK_SERVER** parameter names the Cisco Info Server to which the ticket number should be written. The **FEEDBACK_FIELD** parameter specifies the field into which the trouble ticket number is written. For example:

FEEDBACK = TRUE,

FEEDBACK_SERVER = NCOMS,

FEEDBACK_FIELD = 'Location',

The **JOURNAL** parameter contains the ID number of the **Journal** field in the ARS schema. If this parameter is present, the Writer will update the journal entry for each alert in ARS:

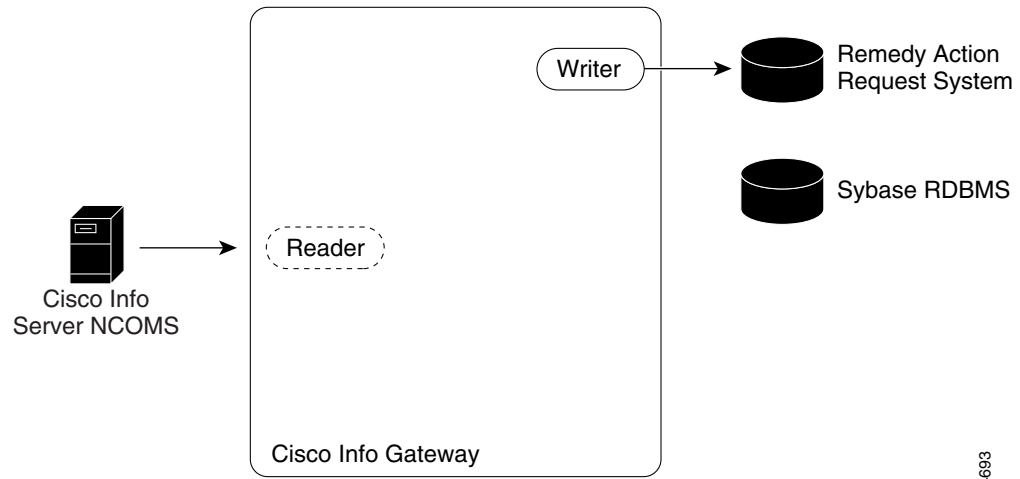
JOURNAL = 563870919

To complete the Writer, the parentheses are closed, followed by a semicolon:

;

When this command is executed, a Writer is started. The configuration now contains one Reader and one Writer as shown in [Figure 6-12](#).

Figure 6-12 Adding an ARS Writer



14693

No data is flowing at this point. The next step is to connect the Reader to the Writer using a route.

Adding a Remedy ARS Route

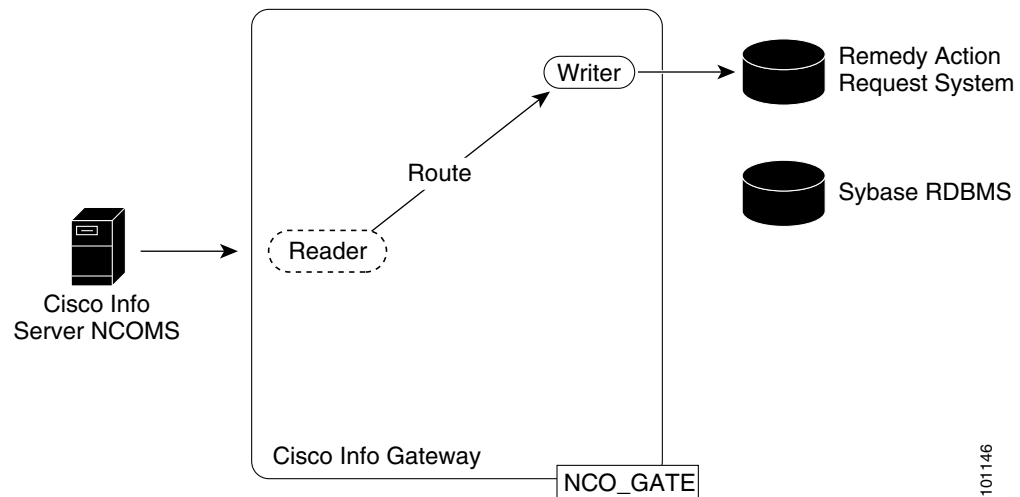
Routes allow information to flow from a Reader to a Writer. The following command adds a route into the ARS gateway:

ADD ROUTE FROM NCOMS_READ TO ARS_WRITER;

When this command is issued, a route is established.

The Remedy ARS Gateway is now complete, as shown in [Figure 6-13](#).

Figure 6-13 Adding the ARS Route



101146

Cisco Info Center alerts will become trouble tickets in Remedy ARS system and the trouble ticket numbers will appear in the alert's **Location** field in the NCOMS Cisco Info Server. When the **Severity**, **InternalLast**, **StateChange**, or **Tally** fields change in alerts on the NCOMS Cisco Info Server, those changes will update the relevant trouble tickets in Remedy ARS; these updates are controlled by the mapping defined previously.

Connecting to Sybase

This section contains the steps enabling alert data to be sent to the Sybase system.

There is no need to configure another Reader for this connection because the Cisco Info Server Reader can forward alerts to multiple Writers.

Creating the Sybase Mapping

The next step is to create a map for Sybase.



Note

The order of the fields to be written to in the mapping must match the order of the fields in the Sybase table. If they do not match, fields will be mapped incorrectly.

The following example creates a Sybase mapping:

```
CREATE MAPPING SYBASE_MAP
(
    ActionTime'= ACTION_TIME,
    ActionCode'= ACTION_CODE,
    Identifier'= '@Identifier',
    Serial' = '@Serial',
    Node'= '@Node',
    NodeAlias'= '@NodeAlias',
    Manager'= '@Manager',
    Agent'= '@Agent',
    AlertGroup'= '@AlertGroup',
    AlertKey'= '@AlertKey',
    Severity'= '@Severity',
    Summary'= '@Summary',
    StateChange'= '@StateChange',
    FirstOccurrence'= '@FirstOccurrence',
    LastOccurrence'= '@LastOccurrence',
    InternalLast'= '@InternalLast',
    Poll'= '@Poll',
    Type'= '@Type',
    Tally'= '@Tally',
    Class'= '@Class',
    Grade'= '@Grade',
    Location'= '@Location',
    OwnerUID'= '@OwnerUID',
```

```

Acknowledged'= '@Acknowledged',
ServerName'= '@ServerName'
;

```

Unlike the Remedy ARS Writer, the Sybase Writer uses names of fields (in single quotes) rather than integer values.

In this mapping, the first two fields do not map to fields in the Cisco Info Server but to **ACTION_TIME** and **ACTION_CODE**. These are special gateway values. **ACTION_TIME** assigns the mapped field the date and time when a change happened. **ACTION_CODE** assigns the letters **I**, **U**, and **D** (for insert, update and delete) to identify what type of change occurred.

The Sybase Writer does not use **ON INSERT ONLY** because the entire record is always written to the Sybase table.

Creating a Sybase Writer

The next step is to create a Sybase Writer.

The Writer for Sybase is different than the Remedy ARS Writer because instead of creating and updating a record for each alert, the Sybase Writer keeps a full transaction log in a database table.

The following command creates the Sybase Writer:

```

START WRITER SYBASE_WRITER
(
    TYPE = SYBASE,
    REVISION = 1,
    SERVER = DARKSTAR,
    MAP = SYBASE_MAP,
    USERNAME = 'sa',
    PASSWORD = '',
    FORWARD_DELETES = TRUE
);

```

The Writer **TYPE** is set to **SYBASE**, with the **REVISION** set to **1**.

The rest of the attributes for the Writer define which Sybase server to communicate with (**DARKSTAR**), which map to use (**SYBASE_MAP**), the user name to log in with (**sa**) and the password (a null string if there is no password, as in this example). The last attribute, **FORWARD_DELETES**, controls the forwarding of alert deletions to the Sybase table. Normally, this is set to **TRUE** to ensure a complete audit trail.

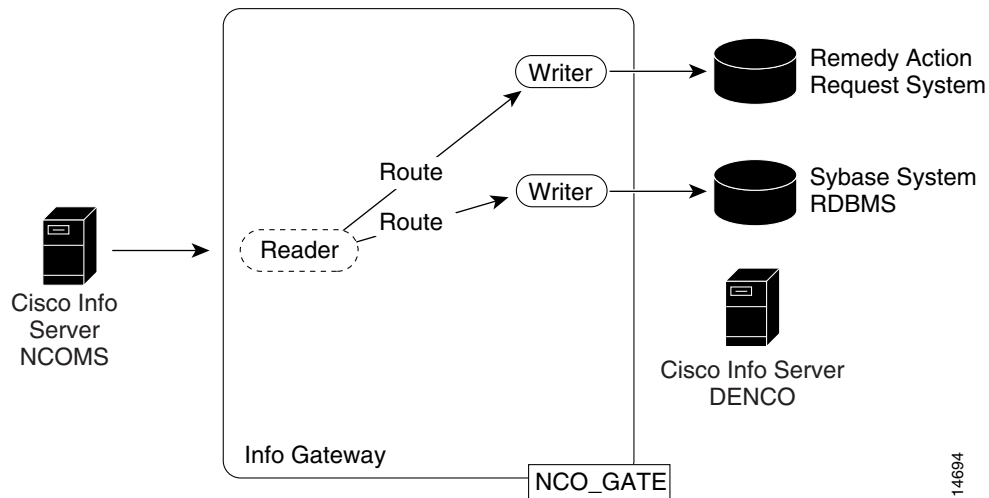
Adding a Sybase Route

With a Writer created, the last stage is to add a route with the command:

ADD ROUTE FROM NCOMS_READ TO SYBASE_WRITER;

The example configuration file now contains a Remedy ARS gateway and a Sybase gateway as shown in [Figure 6-14](#).

Figure 6-14 Adding the Sybase Route



One Reader is now forwarding alerts to two separate Writers.

Connecting to Another Cisco Info Server

The next step is to connect the **DENCO** Cisco Info Server.

By connecting two Cisco Info Servers, a system can be created where different groups of network administrators can be aware of faults under the control of the other groups. In our example, it may be the Cisco Info Server **DENCO** is used by a support group that has its own Cisco Info Mediators connected into **DENCO**. By connecting the Cisco Info Servers through the gateway, Cisco Info Center alerts in **NCOMS** will be forwarded to the **DENCO** Cisco Info Server. **DENCO** operators can manipulate the alerts from **NCOMS** but the changes will not be passed back to the **NCOMS** server.

Creating the Cisco Info Server Mapping

The ordering of the fields in the mapping must exactly match the ordering of the fields in the Cisco Info Server table being written to.

Note the **@Serial** field mapping. Instead of forwarding the value from the source Cisco Info Server, the value is set to **0**. This allows the destination Cisco Info Server to re-serialize the alert—the Cisco Info Servers do not share serial numbers. The following command creates a Cisco Info Server mapping:

```
# Fields marked ON INSERT ONLY are only set when the entry is created
# for the first time (they will not be updated).
#
CREATE MAPPING SERVER_MAP
(
    'Identifier'      = '@Identifier'      ON INSERT ONLY,
    'Serial'         = 0                  ON INSERT ONLY,
    'Node'           = '@Node'            ON INSERT ONLY,
    'NodeAlias'     = '@NodeAlias'       ON INSERT ONLY,
    'Manager'       = '@Manager'         ON INSERT ONLY,
```

```

'Agent' = '@Agent' ON INSERT ONLY,
'AlertGroup' = '@AlertGroup' ON INSERT ONLY,
'AlertKey' = '@AlertKey' ON INSERT ONLY,
'Severity' = '@Severity',
'Summary' = '@Summary' ON INSERT ONLY,
'StateChange' = '@StateChange',
'FirstOccurrence' = '@FirstOccurrence' ON INSERT ONLY,
'LastOccurrence' = '@LastOccurrence',
'InternalLast' = '@InternalLast',
'Poll' = '@Poll' ON INSERT ONLY,
'Type' = '@Type' ON INSERT ONLY,
'Tally' = '@Tally',
'Class' = '@Class' ON INSERT ONLY,
'Grade' = '@Grade' ON INSERT ONLY,
'Location' = '@Location' ON INSERT ONLY,
'OwnerUID' = '@OwnerUID',
'OwnerGID' = '@OwnerGID',
'Acknowledged' = '@Acknowledged',
'Flash' = '@Flash',
'EventId' = '@EventId' ON INSERT ONLY,
'ExpireTime' = '@ExpireTime' ON INSERT ONLY,
'ProcessReq' = '@ProcessReq',
'SuppressEscl' = '@SuppressEscl',
'Customer' = '@Customer' ON INSERT ONLY,
'Service' = '@Service' ON INSERT ONLY,
'PhysicalSlot' = '@PhysicalSlot' ON INSERT ONLY,
'PhysicalPort' = '@PhysicalPort' ON INSERT ONLY,
'PhysicalCard' = '@PhysicalCard' ON INSERT ONLY,
'TaskList' = '@TaskList',
'NmosSerial' = '@NmosSerial' ON INSERT ONLY,
'NmosObjInst' = '@NmosObjInst' ON INSERT ONLY,
'NmosCauseType' = '@NmosCauseType',
'LocalNodeAlias' = '@LocalNodeAlias' ON INSERT ONLY,
'LocalPriObj' = '@LocalPriObj' ON INSERT ONLY,
'LocalSecObj' = '@LocalSecObj' ON INSERT ONLY,
'LocalRootObj' = '@LocalRootObj' ON INSERT ONLY,
'RemoteNodeAlias' = '@RemoteNodeAlias' ON INSERT ONLY,
'RemotePriObj' = '@RemotePriObj' ON INSERT ONLY,
'RemoteSecObj' = '@RemoteSecObj' ON INSERT ONLY,
'RemoteRootObj' = '@RemoteRootObj' ON INSERT ONLY,
'X733EventType' = '@X733EventType' ON INSERT ONLY,
'X733ProbableCause' = '@X733ProbableCause' ON INSERT ONLY,

```

```

'X733SpecificProb' = '@X733SpecificProb'      ON INSERT ONLY,
'X733CorrNotif'   = '@X733CorrNotif'        ON INSERT ONLY,
'ServerName'      = '@ServerName'           ON INSERT ONLY,
'ServerSerial'    = '@ServerSerial'         ON INSERT ONLY
);

```

The final lines of the mapping set the **ServerName** and **ServerSerial** fields. The **ServerName** field is the name of the Cisco Info Server where the alert originated. The **ServerSerial** field is for the serial number of the alert on the original Cisco Info Server. This is how the system maintains serial number information between Cisco Info Servers.

Creating a Cisco Info Server Writer

The following command creates a Cisco Info Server Writer:

```

START WRITER OS_WRITER
(
    TYPE = OBJECT_SERVER,
    REVISION = 1,
    SERVER = DENCO,
    MAP = SERVER_MAP,
);

```

The Cisco Info Server Writer requires four parameters to be set. The **TYPE** parameter is set to **OBJECT_SERVER** to indicate it is a Cisco Info Server Writer. The **REVISION** parameter is currently set to **1**. The **SERVER** parameter specifies the Writer should connect to the **DENCO** Cisco Info Server. The **MAP** parameter selects the mapping to be used.

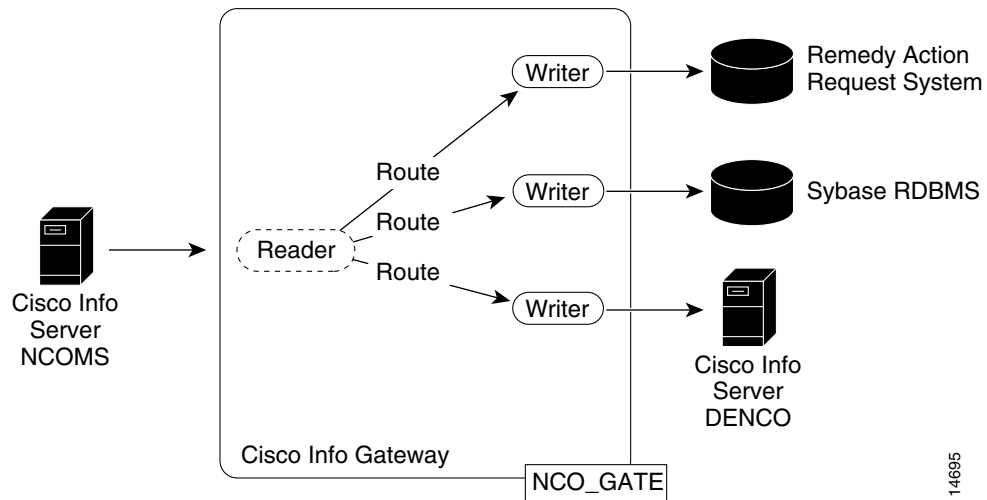
Adding a Route to the Cisco Info Server Writer

The following command adds the route between the Cisco Info Server Reader and Cisco Info Server Writer:

```
ADD ROUTE FROM NCOMS_READ TO OS_WRITER;
```

The example configuration file now contains a Remedy ARS gateway, a Sybase gateway, and a unidirectional Cisco Info Server gateway, as shown in [Figure 6-15](#).

Figure 6-15 Creating a Cisco Info Server Route



In this example, one Reader is forwarding Cisco Info Center alerts to three separate Writers.

Sharing a Writer

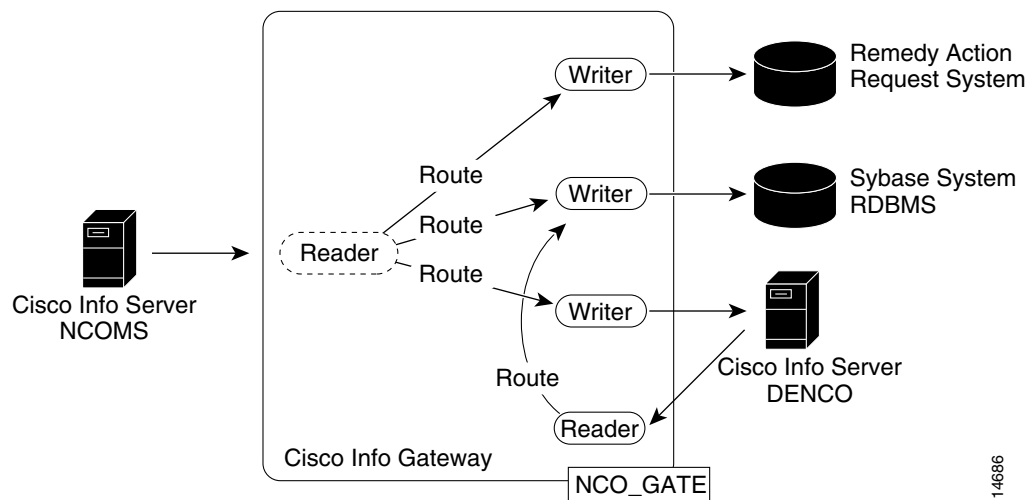
A Writer can be connected to multiple Readers. To connect a Reader to the existing Writer you add a route between them as follows:

START READER DENCO_READ CONNECT TO DENCO;

ADD ROUTE FROM DENCO_READ TO SYBASE_WRITER;

The example configuration file is now complete. The gateway is shown in [Figure 6-16](#).

Figure 6-16 Completed Example Configuration



Cisco Info Server Writer

The Cisco Info Server Writer is used to replicate alerts between Cisco Info Servers. When a route is established between a Cisco Info Server Reader and a Cisco Info Server Writer, alerts are forwarded to the destination Cisco Info Server. In a unidirectional gateway, changes in the Cisco Info Server that is written to are not reflected in the Cisco Info Server being read.

An example mapping is described in [Creating the Cisco Info Server Mapping, page 6-26](#).

Writer Attributes

To create a Cisco Info Server Writer, use the attributes described in the guide for the Cisco Info Server Gateway, available on the Cisco Systems Support Site. The following are sample Writer attributes.

Unidirectional Example

```
START WRITER CUSTOMER_WRITE
(
    TYPE = OBJECT_SERVER,
    REVISION = 1,
    SERVER = DENCO,
    MAP = SERVER_MAP,
    STORE_AND_FORWARD = 1
);
```

Bidirectional Example

```
START WRITER DENCO_WRITER
(
    TYPE = OBJECT_SERVER,
    REVISION = 1,
    SERVER = DENCO,
    MAP = SERVER_MAP,
    COUNTERPART = DENCO_READER
);

START WRITER NCOMS_WRITER
(
    TYPE = OBJECT_SERVER,
    REVISION = 1,
    SERVER = NCOMS,
    MAP = SERVER_MAP,
    COUNTERPART = NCOMS_READER
);
```

Bidirectional Gateways (Failover Pair Setup)

Bidirectional Cisco Info Server gateways allow alerts to flow in both directions between a source and a destination Cisco Info Server. Any changes made in the source Cisco Info Server are replicated in the destination Cisco Info Server, and changes in the destination Cisco Info Server are replicated in the source Cisco Info Server. This ensures both Cisco Info Servers contain the same alerts and allows you to maintain a backup Cisco Info Server.

For more information about Cisco Info Server gateways and how to enable failover, see the guide for the Cisco Info Server Gateway on the Cisco Systems Support Site.

Cisco Info Server Gateway Writers and Failback (Event Replication Between Sites)

Failover occurs when a gateway loses its connection to the primary Cisco Info Server; this allows the gateway to connect to a backup Cisco Info Server. Failback functionality allows the gateway to reconnect to the primary Cisco Info Server when it becomes active again.

Because bidirectional Cisco Info Server gateways are used to resynchronize failover pairs, failback is automatically disabled. This is because one half of the gateway can legitimately be connected to a backup server and so should not be forced to keep failing back to the primary Cisco Info Server.

However, if a bidirectional gateway is being used to share data between two disparate sites, and each site has a failover pair operating, you can manually enable failback on each server. When enabled, the Writer automatically enables failback on its counterpart Reader.

For more information about Cisco Info Server gateways and how to enable failback, see the guide for the Cisco Info Server Gateway on the Cisco Systems Support Site.

Other Gateway Writers and Failback

The Cisco Info Server Reader can fail over and fail back between source Cisco Info Servers without shutting down. This ability is not supported by all gateway Writers. If a Writer is unable to support this mode of failback and failover, the Writer, on detection of the Reader failover/failback, will shut down the gateway and rely on the process agent to restart the gateway.

Writers supporting Reader failover/failback without shutdown are:

- Cisco Info Server
- Sybase Database
- Sybase Reporter
- SNMP
- ServiceView
- Socket
- File
- Informix Database.

Writers supporting failover/failback with shutdown are:

- Remedy
- Siebel

- Oracle Database
- Oracle Reporter
- Peregrine
- Clarify
- HP/ITSM
- Vantive
- Service Desk
- ODBC Database.

Conversion Table Utility

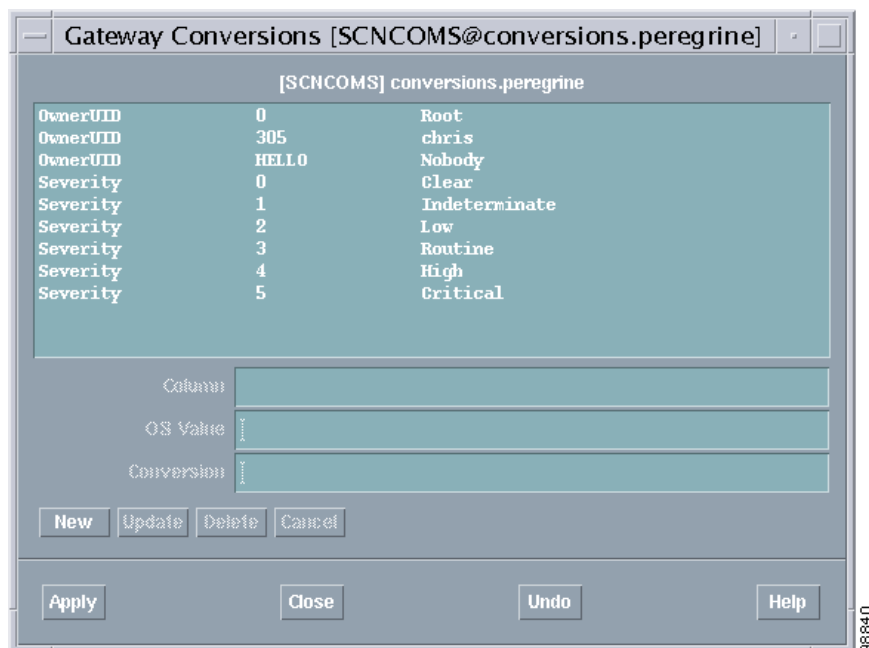
You can create conversion tables to enable certain data conversions to take place between fields. For example, in the Gateway for Peregrine, ServiceCenter users require a status field to be alphabetic and to have a particular value. Cisco Info Center may hold these as numeric values.

A graphical user interface utility is provided to define the conversions. To run the utility enter:

nco_gwconv

The Gateway Conversions window for the Gateway for Peregrine is displayed in [Figure 6-17](#).

Figure 6-17 Gateway For Peregrine Conversion Window



The Gateway Conversion window Title bar contains the server name and table name being used. The work pane displays any existing conversions. Conversion details are displayed in three columns.

The first column contains the Cisco Info Server table column name. The second column contains the values associated with the column. The third column contains the converted value.

Adding a Conversion

To add a new conversion:

-
- Step 1** Click the **New** button.
 - Step 2** Select the **Column** field, then enter the Cisco Info Server column name.
 - Step 3** Enter the Cisco Info Server value in the **OS Value** field.
 - Step 4** Enter the conversion value in the **Conversion** field.
 - Step 5** Click **Apply**.

The new conversion is added.

Updating a Conversion

To update an existing conversion:

-
- Step 1** Select the conversion to update.
The conversion details are populated with the existing values.
 - Step 2** Update the values as required.
 - Step 3** Click **Apply**.
-

Deleting a Conversion

To delete a conversion:

-
- Step 1** Select the conversion to delete. The conversion details are populated with the existing values.
 - Step 2** Click the **Delete** button.
You can undo the delete by clicking the **Undo** button.
 - Step 3** Click **Apply**.
-

