

Cisco Cable Diagnostic Manager Release 1.0

- 1 Product Overview
- 2 System Requirements
- 3 Installing and Starting CCDM
- 4 Reviewing the CCDM Task Menu
- 5 Setting Up CCDM
- 6 Retrieving Subscriber or Provisioning Data by Using an External Interface
- 7 Using Southbound Application Program Interfaces in CCDM
- 8 Setting Parameters in .INI Files
- 9 Using and Customizing Troubleshooting Tips
- 10 Viewing Real-Time Modem Status Reports
- 11 Troubleshooting the Poller
- 12 Uninstalling CCDM
- 13 Sample Code for Northbound and Southbound APIs



1 Product Overview

Cisco Cable Diagnostic Manager Release 1.0 (CCDM 1.0) is a web-based, network management application that is designed for the customer service representative (CSR) at a multiple system operator (MSO). An MSO provides a variety of cable services such as TV, data or voice telephony. The CSR uses CCDM to support the MSO's customers who have cable modems for high-speed data, or telephony. With CCDM, the CSR can provide first-line troubleshooting and support to a customer who is experiencing problems with a cable modem.

CCDM 1.0 runs on PCs, Linux workstations, and Solaris workstations. It provides the following functionality:

- Locates a subscriber's cable modem with one of the following identifiers:
 - MAC address
 - Phone number
 - IP address
- Gathers information from the following sources:
 - Cable Modem Termination System (CMTS)
 - Cable modem
 - Provisioning data
 - Subscriber data
- Provides the following external interface options to integrate a customer's provisioning and subscriber information:
 - Script for various databases
 - Hypertext Transfer Protocol (HTTP) application
 - Lightweight Directory Access Protocol (LDAP)
 - Cisco Broadband Provisioning Registrar (BPR)
 - Cisco Network Registrar (CNR)

To simplify integration, guidelines and samples are provided for the script and HTTP options.

- Automates the work of a highly skilled troubleshooting technician by:
 - Providing troubleshooting tips for common cable modem failure scenarios
 - Allowing customized tips for other cable modem failure scenarios
- Presents information in an easy-to-read format, which is ideal for nontechnical CSRs who need to provide subscribers with information quickly during the course of a phone call.

2 System Requirements

This section describes the system requirements for the following platforms:

- Server running the Linux OS
- Server running the Solaris OS
- Server running the Windows OS
- Client computer
- Cisco uBR7100 series universal broadband router, Cisco uBR7200 series universal broadband router, and Cisco uBR10000 series universal broadband router

Linux System Requirements

You can install CCDM on a server that is running the Linux OS.

Minimum CCDM Server Requirements

- 400 megahertz (MHz) Pentium III workstation

- RedHat Linux 7.2 OS installed
- 400 megabytes (MB) of available disk space
- 128 MB of memory
- Compact Disc Read-Only Memory (CD-ROM) drive
- Simple Network Management Protocol (SNMP) connectivity between the server and managed Cable Modem Termination Systems (CMTSs)
- Connectivity between the server and the location of subscriber and provisioning information

Recommended CCDM Server Requirements

- 1 gigahertz (GHz) Pentium III workstation
- RedHat Linux 7.2 OS installed
- 18 gigabytes (GB) of available disk space
- 2 GB of memory
- CD-ROM drive
- SNMP connectivity between the server and the managed CMTSs
- Connectivity between the server and the location of subscriber and provisioning information

Solaris System Requirements

You can install CCDM on a server that is running the Solaris OS.

Minimum CCDM Server Requirements

- Ultra 5 Solaris workstation
- Solaris 2.8 OS installed
- 400 megabytes (MB) of available disk space
- 128 MB of memory
- Compact Disc Read-Only Memory (CD-ROM) drive
- SNMP connectivity between the server and managed Cable Modem Termination Systems (CMTSs)
- Connectivity between the server and the location of subscriber and provisioning information

Recommended CCDM Server Requirements

- Ultra 5 Solaris workstation
- Solaris 2.8 OS installed
- 18 gigabytes (GB) of available disk space
- 2 GB of memory
- CD-ROM drive
- SNMP connectivity between the server and the managed CMTSs
- Connectivity between the server and the location of subscriber and provisioning information

Windows System Requirements

You can install CCDM on a server that is running the Windows OS.

Minimum CCDM Server Requirements

- 400 megahertz (MHz) Pentium III workstation
- Windows NT or Windows 2000 OS installed
- 250 megabytes (MB) of available disk space
- 128 MB of memory
- Compact Disc Read-Only Memory (CD-ROM) drive
- SNMP connectivity between the server and managed Cable Modem Termination Systems (CMTSs)
- Connectivity between the server and the location of subscriber and provisioning information

Recommended CCDM Server Requirements

- 1 gigahertz (GHz) Pentium III workstation
- Windows NT or Windows 2000 OS installed
- 18 gigabytes (GB) of available disk space
- 2 GB of memory
- CD-ROM drive
- SNMP connectivity between the server and the managed CMTSs
- Connectivity between the server and the location of subscriber and provisioning information

Minimum Client Requirements

After CCDM is installed on a server, you can access and use CCDM from a client computer with a web browser.

- Pentium II workstation with Windows 98, Windows NT, or Windows 2000
or
Ultra 5 workstation with Solaris 2.8
- 128 MB of memory
- IP connection to the CCDM server
- Browser for Linux OS
 - Netscape 4.78
- Browser for Solaris OS
 - Netscape 4.5 or a later release
- Browser for Windows OS
 - Netscape 4.5 or a later release
 - Internet Explorer 5.0 or a later release

Cisco uBR7100, Cisco uBR7200, and Cisco uBR10000 Series Router Requirements

The routers must have the following attributes set to interact with CCDM.

- Cisco IOS Release 12.1(11)EC or a later release for the Cisco uBR7100 series and the Cisco uBR7200 series
or
Cisco IOS Release 12.2(x)BC or later release for the Cisco uBR10000 series
- SNMP enabled by entering the following commands:

```
Router# config t
Router(config)# snmp-server community public RO
```



Note CCDM retrieves the Cisco IOS release number, router name, and router type by using SNMP.

3 Installing and Starting CCDM

The following procedures describe how to install Cisco Cable Diagnostic Manager on each supported platform.

Installing and Starting CCDM 1.0 on the Linux Platform

Step 1 Log in as root.

Step 2 Insert the CCDM CD-ROM into the CD-ROM drive.

Step 3 To mount the CD, enter:

```
/bin/mount /mnt/cdrom
```

Step 4 To change to the CCDM Linux directory, enter:

```
cd /mnt/cdrom/linux
```

Step 5 To start the installation program, enter:

```
./install
```

The CCDM server is now running.

Step 6 To start CCDM by typing a command, enter:

```
cd /opt/CSCOcdm/bin
```

```
./start_app
```

or

To launch CCDM by using a browser:

Open a browser and enter the URL for your CCDM server:

```
http://CCDM server IP address:8090/CCDMHomePage.html
```

Step 7 In the Log In window (see Figure 1), enter your username and password.

Step 8 (Optional) To stop CCDM, enter:

```
cd /opt/CSCOcdm/bin
```

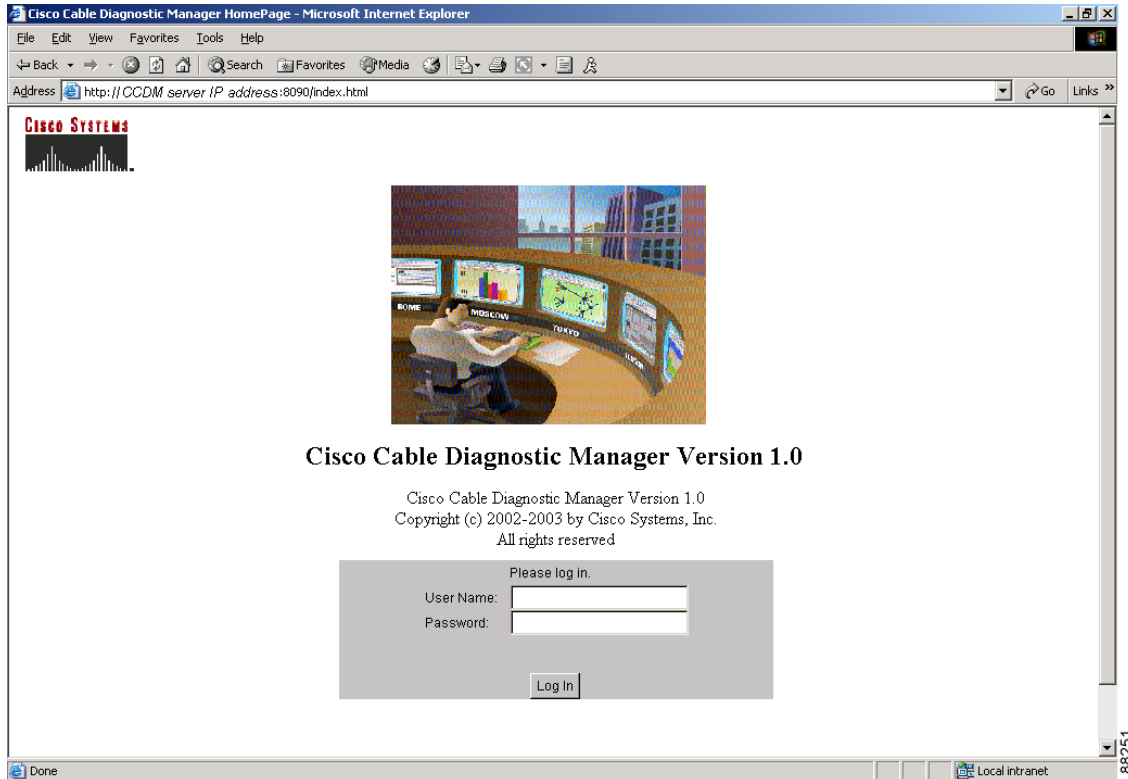
```
./stop_app
```

or

To exit CCDM by using a browser:

From the Exiting CCDM Menu, choose Log Out.

Figure 1 CCDM Log In Window



Installing and Starting CCDM 1.0 on the Solaris Platform

Step 1 Log in as root.

Step 2 Insert the CCDM CD-ROM into the CD-ROM drive.

Step 3 To change to the CCDM Solaris directory and install CCDM, enter:

```
cd /cdrom/cdrom0/solaris
./install
```

The CCDM server is now running.

Step 4 To start CCDM by typing a command, enter:

```
cd /opt/CSCOcdm/bin
./start_app
or
```

To launch CCDM by using a browser:

Open a browser and enter the URL for your CCDM server:

<http://CCDM server IP address:8090/CCDMHomePage.html>

Step 5 In the Log In window (see Figure 1), enter your username and password.

Step 6 (Optional) To stop CCDM, enter:

```
cd /opt/CSCOcdm/bin
./stop_app
```

or

To exit CCDM by using a browser:

From the Exiting CCDM Menu, choose Log Out.

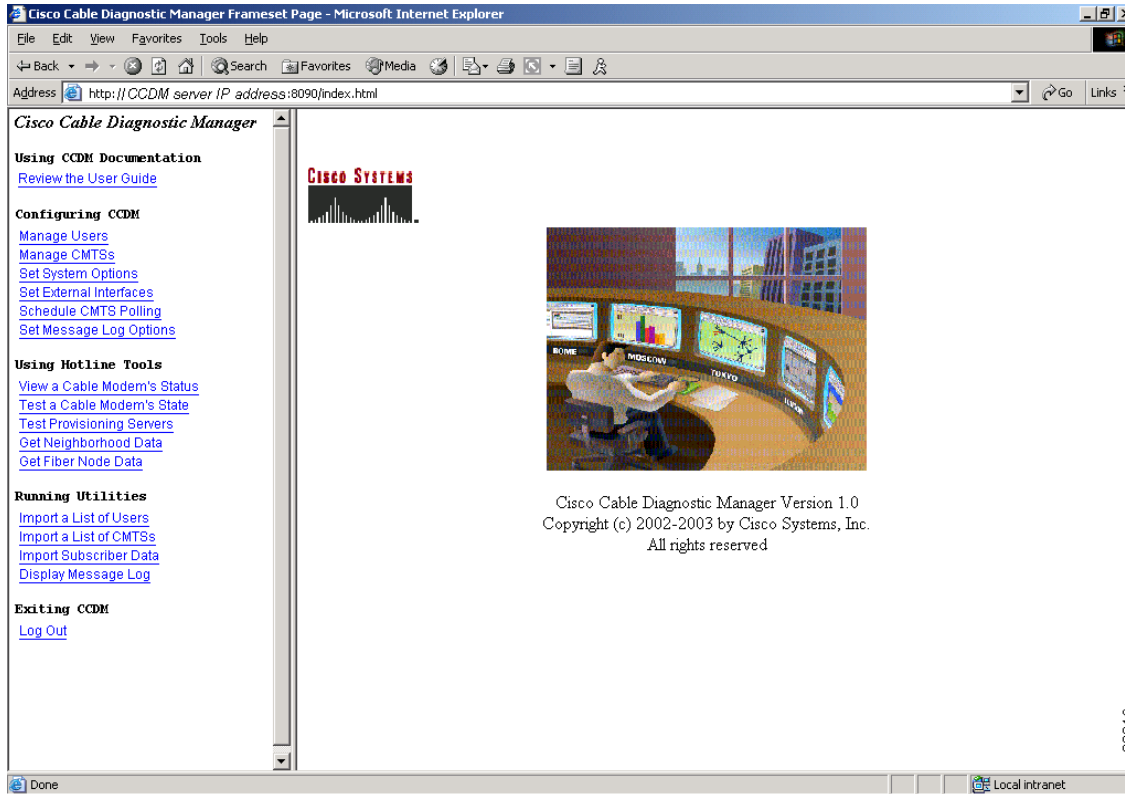
Installing and Starting CCDM 1.0 on the Windows Platform

- Step 1** Insert the CCDM CD-ROM into the CD-ROM drive.
- Step 2** From Windows Explorer, double-click SETUP in CDROM_DRIVE\WIN\DISK1.
- Step 3** Select the default answers to the installation questions.
The CCDM server is now running.
- Step 4** To start CCDM by using Windows menus, choose:
Start > Programs > CCDM > Start CCDM
or
To launch CCDM by using a browser, open a browser and enter:
http://CCDM server IP address:8090/
- Step 5** In the Log In window (see Figure 1), enter your username and password.
- Step 6** (Optional) To stop CCDM, choose:
Start > Programs > CCDM > Stop CCDM
or
To exit CCDM by using a browser:
From the Exiting CCDM Menu, choose Log Out.
-

4 Reviewing the CCDM Task Menu

After you start CCDM, you see the CCDM task menu. A version of the menu that lists all the available tasks is shown in Figure 2:

Figure 2 CCDM Task Menu



The tasks in the menu are divided into the following groups:

- Using CCDM Documentation
- Configuring CCDM
- Using Hotline Tools
- Running Utilities
- Exiting CCDM

The next sections describe the tasks within each group.

Using CCDM Documentation

This section contains the following task:

- Review the User Guide—To link to the CCDM User Guide directly from CCDM

Configuring CCDM

This section contains the following tasks:

- Manage Users—To specify who can access the application and with what level of access permissions
- Manage CMTSs—To specify the routers you want to manage, by entering the information manually
- Set System Options—To modify system-level parameters that are saved to the CONFIGS.INI file

- Set External Interfaces—To specify how and where to access subscriber and provisioning information
- Schedule CMTS Polling—To schedule how often CCDM's Poller gets provisioning information from CMTSs
- Set Message Log Options—To filter which error and informational messages are logged by the server

Using Hotline Tools

This section contains the following tasks:

- View a Cable Modem's Status—To specify a cable modem to query, ping, and view its status report
- Test a Cable Modem's State—To determine the state of a specific cable modem
- Test Provisioning Servers—To test connectivity from a CMTS or the CCDM server to the provisioning servers
- Get Neighborhood Data—To see how many cable modems in a neighborhood are online, offline, and in the same state as a modem you query
- Get Fiber Node Data—To get modem counts and see how many are online and offline, in the fiber node where a cable modem is located

Running Utilities

This section contains the following tasks:

- Import a List of Users—To import an ASCII text file of a list of users who can access the application, and with what level of access permissions
- Import a List of CMTSs—To import an ASCII text file of all the managed CMTSs, so that you don't have to enter each CMTS manually
- Import Subscriber Data—To import an ASCII text file of subscriber data, such as name and phone number
- Display Message Log—To view error and information messages logged by the server, by using filters to show the level of detail you want

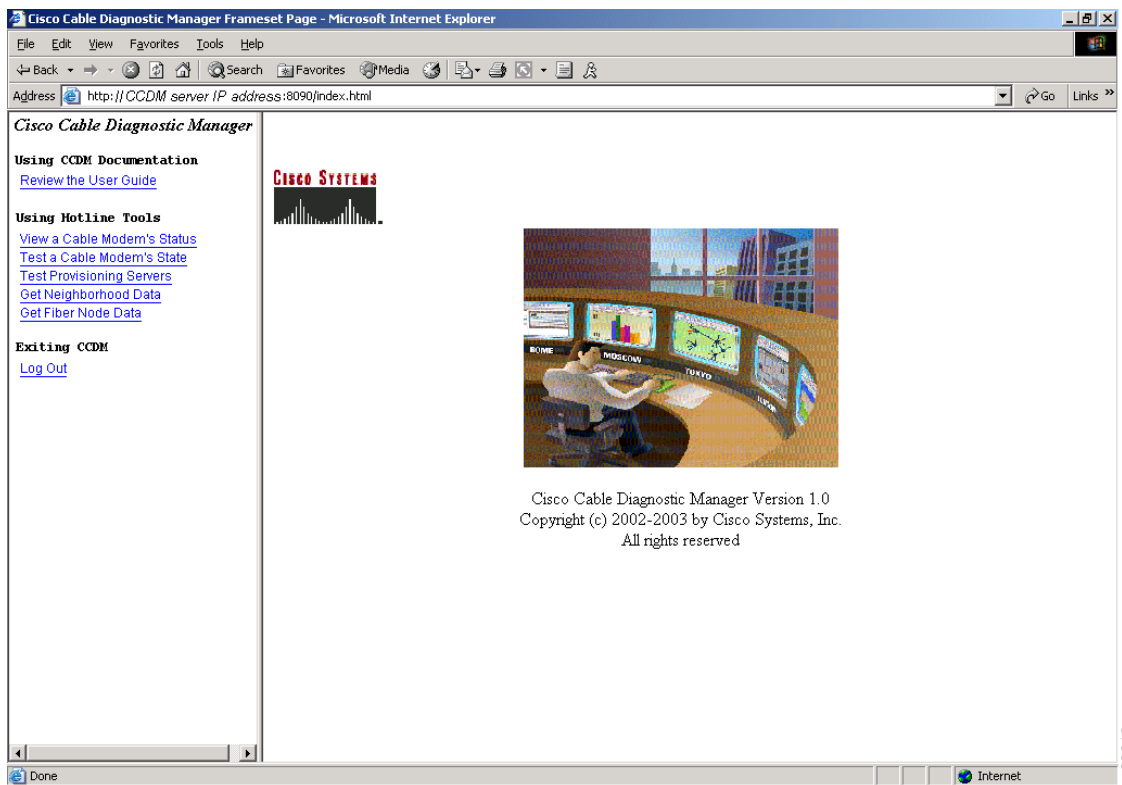
User Types in CCDM

The number of tasks you see in the CCDM menu varies according to the type of user you are. There are three user types in CCDM, each with access to different tasks:

- ADMIN—The administrator has access to all tasks in the menu. See Figure 2.
- CSR-T2—The second-tier customer service representative has access to all the tasks in the Using Hotline Tools group. This group of tasks allows a CSR-T2 representative to:
 - Query a specific cable modem
 - Ping the modem
 - Review the modem's status
 - Troubleshoot problems with the modem by using the diagnostic tools: Test a Cable Modem's State, Test Provisioning Servers, Get Neighborhood Data, and Get Fiber Node Data.
- CSR-T1—The first-tier customer service representative has access to one task in the Using Hotline Tools group. This task, View a Cable Modem's Status, allows a CSR-T1 representative to:
 - Query a specific cable modem
 - Ping the modem
 - Review the modem's status

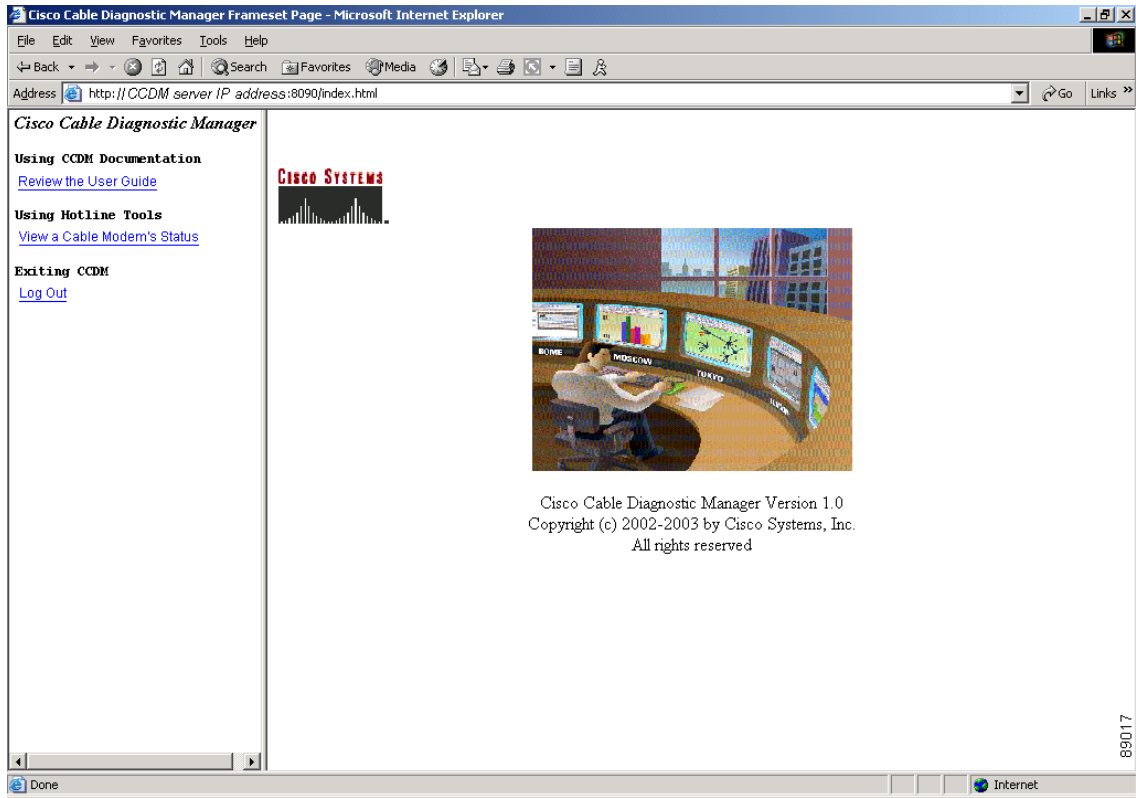
The menu for the CSR-T2 user is shown in Figure 3.

Figure 3 CCDM Task Menu for CSR-T2Users



The menu for the CSR-T1 user is shown in Figure 4.

Figure 4 CCDM Task Menu for CSR-T1 Users



Workflow of Administrator and CSR Tasks

Figure 5 Administrator and CSR Tasks in CCDM

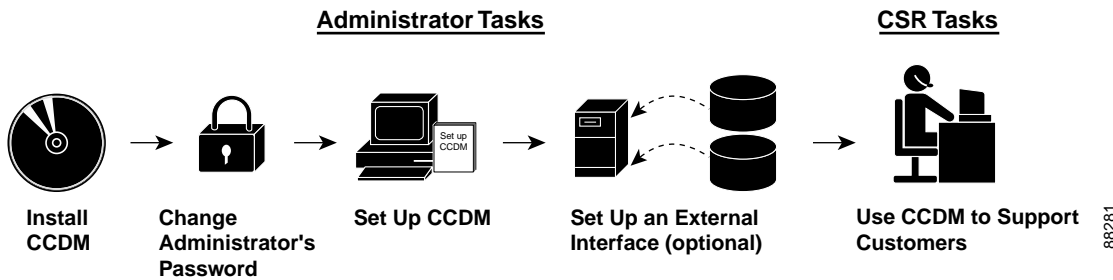


Figure 5 illustrates a high-level workflow of tasks that each type of user, the administrator and the CSR, performs in CCDM.

Administrator Tasks

The first four tasks shown in Figure 5 are ones that only an administrator can perform.

- **Install CCDM**—The administrator installs CCDM on one of the supported platforms: Linux, Solaris, or Windows.
- **Change Administrator's Password**—The administrator changes the default ADMIN password to ensure proper security.
- **Set Up CCDM**—The administrator performs the following setup tasks to configure CCDM according to the site's needs:
 - Add users, routers, and subscriber information
 - Configure parameters in .INI files: CONFIGS.INI, POLLER.INI, and TROUBLESHOOT_TIPS.INI
 - Set and display the message log to see errors logged by the server
- **Set Up an External Interface (optional)**—If CCDM will not retrieve subscriber or provisioning data from its local database, which is the system default, the administrator sets up an external interface to retrieve the data by using one of the following methods:
 - **Subscriber Information**—Can be accessed by script, HTTP or Lightweight Directory Access Protocol (LDAP)
 - **Provisioning Information**—Can be accessed by script, HTTP, LDAP, Cisco Broadband Provisioning Registrar (BPR), or Cisco Network Registrar (CNR)

The remainder of this guide explains these tasks in greater detail.

CSR Tasks

The last task shown in Figure 5 is one that a customer service representative performs.

- **Use CCDM to Support Customers**—The first-tier customer service representative can:
 - Query a specific cable modem
 - Ping the modem
 - Review the modem's status

In addition to the above tasks, the second-tier customer service representative can troubleshoot the modem by performing the following diagnostic tests:

- Test a cable modem's state
- Test provisioning servers
- Get neighborhood data
- Get fiber node data

5 Setting Up CCDM

This section describes the following important setup tasks that only a CCDM administrator can perform:

- Changing the administrative password
- Adding users
- Adding router information
- Adding subscriber information

Changing the Admin Password

For security reasons, we recommend that you change the default password for the administrator.

To change the administrator password:

Step 1 Open the browser and enter the URL for your CCDM server:

`http://yourmachine:8090/CCDMHomePage.html`



Note In the URL, *yourmachine* is the server on which you installed CCDM.

The Login dialog box appears. See Figure 1.

Step 2 Log in as admin with the default password changeme.

Step 3 To change the password, from the Configuring CCDM menu choose Manage Users.

About Adding Users

You can add users in CCDM by using either of the following methods:

- Enter information for a user by manually filling in fields for each user
- Import an ASCII file that contains a list of users

For each user, the ASCII text file must contain the following three fields: User Name, Password, and User Type. Each field must be separated with a comma. A sample file is in the following locations:

- Linux and Solaris—`opt/CSCOcdm/samples/text_files/users.txt`
- Windows—`%CSCOCCDM_ROOT%\samples\text_files\users.txt`

Example 1 ASCII File Format for One User

The following example shows one user's data in this format.

```
jsmith,feb1962,CSR-T2
```



Note For a complete description of each field in the ASCII file, see the online help for Import a List of Users.

Adding Users

To add users in CCDM:

Step 1 Log in as admin.

Step 2 To import a list of users, from the Running Utilities menu, choose Import a List of Users and specify the ASCII file that contains a list of users. (See Figure 6.)

or

To add a user manually, from the Configuring CCDM menu, choose Manage Users. (See Figure 7.)

Figure 6 *Import a List of Users Dialog Box*

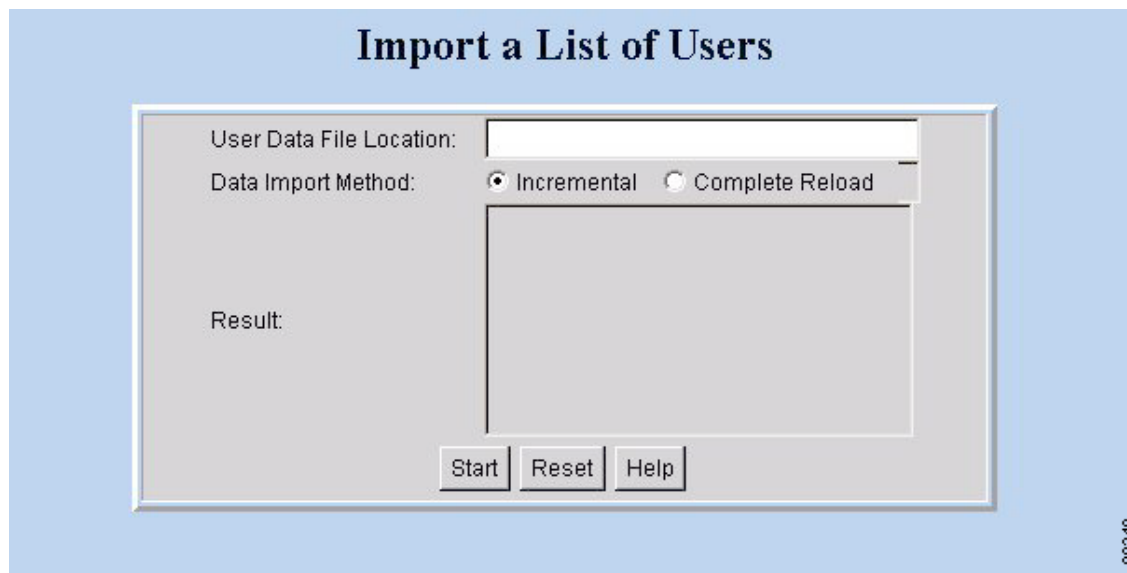
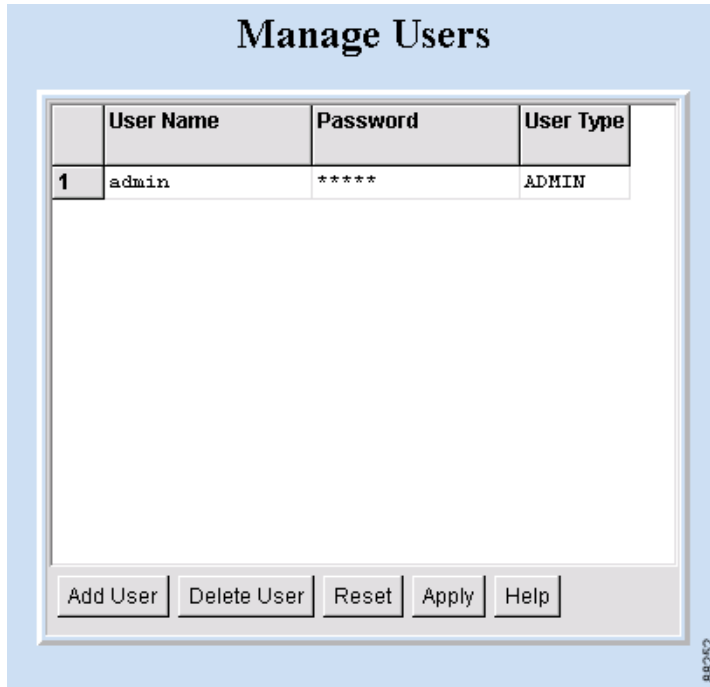


Figure 7 Manage Users Dialog Box



About Adding Router Information

You can add router information in CCDM by using either of the following methods:

- Enter information for a router by manually filling in fields for each router
- Import an ASCII file that contains a list of managed CMTSs

If you choose to import an ASCII file, for each CMTS the file must contain information or a placeholder for the 13 fields described below.

- Fields 1 to 4—Enter the IP address, Community String (Read), Community String (Write), and CM Community String (Read).
- Fields 5 to 9—Enter a comma as a placeholder for each field; these fields are used only in Cisco Cable Broadband Troubleshooter (CBT).
- Field 10—Enter the Telnet port this router uses (the default is 23).
- Fields 11 and 12—Enter a comma as a placeholder for each field; these fields are used only in CBT.
- Field 13—Enter the setting for Ping CMTS. The options are *yes* or *no*, and the default is *yes*.

A sample file is in the following locations:

- Linux and Solaris—`opt/CSCOccdm/samples/text_files/cmts.txt`
- Windows—`%CSCOCCDM_ROOT%\samples\text_files\cmts.txt`

Example 2 ASCII File Format for One CMTS

The following example shows the data for a CMTS in this format:

```
172.21.73.10,public,private,public,,,,,23,,,yes
```



Note For a complete description of each field in the ASCII file, see the online help for Import a List of CMTSs.

Adding Router Information

To add the Cisco uBR7100, Cisco uBR7200, and Cisco uBR10000 series routers in CCDM:

Step 1 Log in as admin.

Step 2 To import a list of routers, from the Running Utilities menu, choose Import a List of CMTSs and specify the ASCII file that contains a list of managed CMTSs. (See Figure 8.)

or

To add a router manually, from the Configuring CCDM menu, choose Manage CMTSs.

To successfully manage routers, make sure that you enable SNMP on these routers.



Tip Verify that access lists, if configured in the router, allow for SNMP read access from your workstation.

Figure 8 Import a List of CMTSs Dialog Box

Import a List of CMTSs

CMTS Data File Location:

Data Import Method: Incremental Complete Reload

Import Status:

Start Time:

End Time:

Result:

88248

About Adding Subscriber Information

To add subscriber information in CCDM, you import an ASCII text file that contains the subscriber information.

For each subscriber, the ASCII text file can contain 12 fields. Each field, including blank ones, must be separated with | (the pipe symbol). The following fields are typical, although only the first one is required:

1. Cable Modem MAC Address
2. Account ID
3. Customer Name
4. Phone Number
5. Address
6. City
7. State
8. Zip

9. Country
10. Class of Service
11. Fiber Node
12. User-Defined Field

A sample file is in the following locations:

- Linux and Solaris—opt/CSCOcdm/samples/text_files/subscriber.txt
- Windows—%CSCOCCDM_ROOT%\samples\text_files\subscriber.txt

Example 3 *ASCII File Format for One Subscriber*

The following example shows the data for a subscriber in this format:

```
000216d5a0cf|ID987654|JohnDoe|4085551212|175 West Tasman|San Jose|CA|95134|USA|Gold|FiberNode_SanJose|Paid
```



Note For a complete description of each field in the ASCII file, see the online help for Import Subscriber Data.

Adding Subscriber Information

To add subscriber information to CCDM:

-
- Step 1** Log in as admin.
 - Step 2** From the Running Utilities menu, choose Import Subscriber Data and specify the ASCII file that contains subscriber data. (See Figure 9.)
-

Figure 9 Import Subscriber Data Dialog Box

Import Subscriber Data

Subscriber Data File Location:	<input type="text"/>
Data Import Method:	<input checked="" type="radio"/> Incremental <input type="radio"/> Complete Reload
Import Status:	Idle
Start Time:	NA
End Time:	NA
Result:	<div style="border: 1px solid black; height: 100px; width: 100%;"></div>

86250

6 Retrieving Subscriber or Provisioning Data by Using an External Interface

CCDM supports two types of interfaces:

- Northbound interfaces—These interfaces provide two capabilities. They allow:
 - An external application to communicate with CCDM
 - CCDM to communicate with the subscriber and provisioning databases, which can be in CCDM or external to CCDM

This section describes using northbound interfaces in CCDM.

- Southbound interfaces—These interfaces allow CCDM to retrieve proprietary information from a CMTS that is not from Cisco. Southbound interfaces are described in “Using Southbound Application Program Interfaces in CCDM” page 29.

With CCDM you can set up an external interface to retrieve subscriber or provisioning data.

- Subscriber information—Data such as a customer’s name and account number
- Provisioning information—Data such as the IP address for a customer’s cable modem and the IP address for the CMTS on which the modem is located

You can use the following methods to set up an external interface:

- Script—Can be used for subscriber or provisioning information or both
- Application on an HTTP server—Can be used for subscriber or provisioning information or both
- LDAP—Can be used for subscriber or provisioning information or both
- BPR—Can be used for provisioning information only
- CNR—Can be used for provisioning information only
- Local Database—(Default) Can be used for subscriber or provisioning information or both



Note If you set an external interface for provisioning information and CCDM does not find provisioning data there, CCDM reverts to the local database. This fallback behavior does not apply to subscriber information.

Figure 10 shows the dialog box where you specify the method. The remainder of this section describes how to set up an external interface with each method.

Figure 10 Set External Interfaces Dialog Box

Set External Interfaces

Subscriber Information:

Script Script Location:

HTTP URL Location:

LDAP

Local database

LDAP Access Parameters

Import Subscriber

Provisioning Information:

Script Script Location:

HTTP URL Location:

BPR

CNR

RDU server IP:

RDU server port:

Admin password:

Cluster Name / IP:

Admin User Name :

Admin Password:

Client Path:

LDAP Access Parameters

Poller

Reset Apply Help

88308

About Retrieving Data with a Script

You can implement an external data retrieval application via a scripting language. To see a sample script that you can modify according to your needs, see “Northbound Shell Script for Retrieving Subscriber or Provisioning Information” page 44.



Note You can also implement an external data retrieval application in programming languages such as PL/SQL, C/C++, and Java, as long as it is embedded in a shell script.

Script Parameters for MAC Address Information

The script must support the ability to search for a cable modem’s MAC address by entering the customer’s phone number, IP address, or fiber node. The input parameters for the script are:

- *subscriber-info-script-name* GET_MAC PHONE *phone-number*
- *subscriber-info-script-name* GET_MAC FIBER_NODE *fibernode*
- *provision-info-script-name* GET_MAC IP *ip-address*

For example, the following script invokes a search for the modem’s MAC address by using the phone number 408-123-4567:

```
/opt/tools/subscriber-query.sh GET_MAC PHONE 4081234567
```

The next script invokes a search for the modem's MAC address by using the fiber node SanJose_1:

```
/opt/tools/subscriber-query.sh GET_MAC FIBER_NODE SanJose_1
```

The last script invokes a search for the modem's MAC address by using the IP address 172.2.3.1:

```
/opt/tools/provision-query.sh GET_MAC IP 172.2.3.1
```



Note A MAC address query by phone number or fiber node invokes the subscriber information script. A MAC address query by IP address invokes the provisioning information script.

The output of the script must be in the following format:

OUT_DATA=MAC followed by one or more MAC addresses (non-dotted format) separated by ^

The following examples show the output for one and two MAC addresses:

- OUT_DATA=MAC^001c64ff23ef^
- OUT_DATA=MAC^001c64ff23ef^013e45ed1245^

Script Parameters for Subscriber Information

If you use a script to query subscriber information, it needs to support the ability to search for subscriber data, such as name and account number, by entering the MAC address of the subscriber's cable modem. The input parameters for the script are:

script-name GET_SUBSCRIBER MAC *non-dotted-modem-mac-address*

For example, the following script invokes a search for subscriber data by using the MAC address 001c.ab23.45fe in a non-dotted format:

```
/opt/tools/subscriber-query.sh GET_ SUBSCRIBER MAC 001cab2345fe
```

The output of the script must be in the following format:

OUT_DATA=SUBSCRIBER^Account Number=*value*^Name=*value*^Address=*value*^Phone=*value*^Class of Service=*value*^Fiber Node=*FIBER_*^misc1=*value1*^misc2=*value2*

- The fields specified in the above output are required and must be in the order shown.
- If information for any of the required fields is not available from the database, its value should be set to N/A.
- The format for the fields is up to the customer. For example, Name=*First_Last* and Name=*Last_First* are both acceptable.
- (Optional) User-defined fields can follow the Fiber Node value.

The following example shows the output for a subscriber's data, which includes two user-defined fields after Fiber Node:

```
OUT_DATA=SUBSCRIBER^Account Number=123456^Name=Doe_John^Address=123 Tasman, San Jose, CA  
93443^Phone=4081234567^Class of Service=N/A^FiberNode=FIBER_1^Customer Since=1999^Account Status=Paid
```



Note In the Real-Time Modem Status Report, CCDM displays subscriber information in the order in which the fields are returned from the script.

Script Parameters for Provisioning Information

If you use a script to query provisioning information, it needs to support the ability to search for provisioning data, such as CMTS and cable modem IP addresses, by entering one MAC address of a cable modem or multiple MAC addresses of multiple cable modems. The input parameters for the scripts are:

- *script-name* GET_PROVISION MAC *non-dotted-modem-mac-address*
- *script-name* GET_PROVISION MAC_LIST *list non-dotted-modem-mac-addresses separated by ^*

For example, the following script invokes a search for provisioning data by using the MAC address 001c.ab23.45fe:

```
/opt/tools/subscriber-query.sh GET_PROVISION MAC 001cab2345fe
```

The output of the script must be in the following format:

```
OUT_DATA=PROVISION^cmts-ip-address^cm-ip-address^
```

The following example shows the output for provisioning data:

```
OUT_DATA=PROVISION^127.23.45.1^127.23.127.5^
```

The next script example invokes a search for provisioning data by using multiple MAC addresses, 001c.ab23.45fe and 001c.ab23.45ff:

```
/opt/tools/subscriber-query.sh GET_PROVISION MAC_LIST 001cab2345fe^001c.ab23.45ff
```

The output of the script must be in the following format:

```
OUT_DATA=PROVISION^mac1=cmts-ip-address,cm-ip-address^mac2=cmts-ip-address,cm-ip-address^
```

The following example shows the output for provisioning data:

```
OUT_DATA=PROVISION^001cab2345fe=127.23.45.1,127.23.127.5^001cab2345ff=127.23.45.1,127.23.127.5^^
```

Error Handling for Script

If an error occurs within the script or the embedded application called by the script, the script should return the output in the following format:

```
OUT_DATA=ERROR^error-message^
```

The following example shows output for an error:

```
OUT_DATA=ERROR^Unable to query the subscriber database.^
```

In the graphical user interface (GUI), CCDM displays the error message to the user in a message box.

Retrieving Data with a Script

To retrieve subscriber or provisioning information by using a script:

-
- Step 1** Review the sample script and modify it according to your needs. See “Sample Script Code” page 46.
 - Step 2** From the Configuring CCDM menu, choose Set External Interfaces to specify one or both of the following:
 - In the Subscriber Information section, click Script and enter the Script Location of the file.
 - In the Provisioning Information section, click Script and enter the Script Location of the file.
-

Troubleshooting Script Problems

See “Directions for Troubleshooting the Script” page 45.

About Retrieving Data with an Application on an HTTP Server

You can implement an external data retrieval application via an application running on an HTTP server. CCDM sends the request to the HTTP server via the POST method. To see a sample application file that you can modify according to your needs, see “Northbound Java Code for Retrieving Subscriber or Provisioning Information” page 47.

HTTP Parameters for MAC Address Information

The application running on an HTTP server must support the ability to search for a cable modem’s MAC address by entering the customer’s phone number, IP address, or fiber node. CCDM sends the following parameters to the appropriate server:

- REQUEST=GET_MAC
- SEARCH_TYPE=PHONE, IP or FIBERNODE

- IN_PARAM=*phone* or *ip*

For example, the following parameters search for the modem's MAC address by using the phone number 408-123-4567:

```
REQUEST=GET_MAC
SEARCH_TYPE=PHONE
IN_PARAM=4081234567
```

The response from the server must be in the following format:

OUT_DATA=MAC followed by one or more MAC addresses separated by ^

The following examples show the output for one and two MAC addresses:

```
OUT_DATA=MAC^001c64ff23ef^
OUT_DATA=MAC^001c64ff23ef^013e45ed1245^
```



Note When the SEARCH_TYPE is PHONE, the subscriber information URL is used. When the SEARCH_TYPE is IP, the provisioning information URL is used.

HTTP Parameters for Subscriber Information

If you use an application running on an HTTP server to query subscriber information, it needs to support the ability to search for subscriber data, such as name and account number, by entering the MAC address of the subscriber's cable modem. CCDM sends the following parameters to the server:

- REQUEST=GET_SUBSCRIBER
- SEARCH_TYPE=MAC
- IN_PARAM=*non-dotted-modem-mac-address*

For example, the following parameters search for subscriber data by using the MAC address 001cab2345fe:

```
REQUEST=GET_SUBSCRIBER
SEARCH_TYPE=MAC
IN_PARAM=001cab2345fe
```

The response from the server must be in the following format:

OUT_DATA=SUBSCRIBER^Account Number=value^Name=value^Address=value^Phone=value^Class of Service=value^Fiber Node=FIBER_^misc1=value1^misc2=value2

- The fields specified in the above response are required and must be in the order shown.
- If information for any of the required fields is not available from the database, its value should be set to N/A.
- The format for the fields is up to the customer. For example, Name=First_Last and Name=First_Last are both acceptable.
- (Optional) User-defined fields can follow the Fiber Node value.

The following example shows the output for a subscriber's data, which includes two user-defined fields after Fiber Node:

```
OUT_DATA=SUBSCRIBER^Account Number=123456^Name=Doe_John^Address=123 Tasman, San Jose, CA
93443^Phone=4081234567^Class of Service=N/A^FiberNode=FIBER_1^Customer Since=1999^Account
Status=Paid
```



Note In the Real-Time Modem Status Report, CCDM displays subscriber information in the order in which the fields are returned from the script.

HTTP Parameters for Provisioning Information

If you use an application running on an HTTP server to query provisioning information, it needs to support the ability to search for provisioning data, such as CMTS and cable modem IP addresses, by entering the MAC address of the cable modem or multiple MAC addresses of multiple cable modems.

To search for provisioning data by entering a single MAC address, CCDM sends the following parameters to the server:

- REQUEST=GET_PROVISION
- SEARCH_TYPE=MAC
- IN_PARAM=*non-dotted-modem-mac-address*

For example, the following parameters search for provisioning data by using the MAC address 001cab2345fe:

```
REQUEST=GET_PROVISION
SEARCH_TYPE=MAC
IN_PARAM=001cab2345fe
```

The response from the server must be in the following format:

```
OUT_DATA=PROVISION^cmts-ip-address^cm-ip-address^
```

The following example shows the output for provisioning data:

```
OUT_DATA=PROVISION^127.23.45.1^127.23.127.5^
```

To search for provisioning data by entering multiple MAC addresses, CCDM sends the following parameters to the server:

- REQUEST=GET_PROVISION
- SEARCH_TYPE=MAC_LIST
- IN_PARAM=*list non-dotted-modem-mac-addresses separated by ^*

For example, the following parameters search for provisioning data by using the MAC addresses 001cab2345fe and 0001abcdefff:

```
REQUEST=GET_PROVISION
SEARCH_TYPE=MAC_LIST
IN_PARAM=001cab2345fe^0001abcdefff^
```

The response from the server must be in the following format:

```
OUT_DATA=PROVISION^mac1=cmts-ip-address,cm-ip-address^mac2=cmts-ip-address,cm-ip-address^
```

The following example shows the output for provisioning data:

```
OUT_DATA=PROVISION^001cab2345fe=127.23.45.1,127.23.127.5^0001abcdefff=127.23.45.1,127.23.127.5^
```

Error Handling for HTTP

If an error occurs within the application running on an HTTP server, the script should return the output in the following format:

```
OUT_DATA=ERROR^error-message^
```

The following example shows output for an error:

```
OUT_DATA=ERROR^Unable to query the subscriber database.^
```

In the GUI, CCDM displays the error message to the user in a message box.

Retrieving Data with an HTTP Application

To retrieve subscriber or provisioning information with an application on an HTTP server:

-
- Step 1** Review the sample application file and modify it according to your needs. See “Sample Java Code” page 51.
 - Step 2** From the Configuring CCDM menu, choose Set External Interfaces to specify one or both of the following:
 - In the Subscriber Information section, click HTTP and enter the URL Location of the file.
 - In the Provisioning Information section, click HTTP and enter the URL Location of the file.
-

Troubleshooting HTTP Problems

See “Directions for Troubleshooting the Sample HTTP Application” page 50.

About Retrieving Data with LDAP

You can use Lightweight Directory Access Protocol (LDAP) to retrieve external subscriber or provisioning information. LDAP is a nonproprietary, standards-based protocol. If you use the LDAP method, see your LDAP server administrator to obtain the information about the LDAP server that CCDM requires.

Retrieving Data with LDAP

To retrieve external subscriber or provisioning information by using LDAP:

-
- Step 1** From the Configuring CCDM menu, choose Set External Interfaces.
- Step 2** Depending on the type of information you want to retrieve, do one or both of the following:
- To retrieve subscriber information, in the Subscriber Information section select LDAP and click LDAP Access Parameters to open the LDAP Access Parameters dialog box.
 - To retrieve provisioning information, in the Provisioning Information section select LDAP and click LDAP Access Parameters to open the LDAP Access Parameters dialog box.
- Step 3** To fill in the LDAP Access Parameters dialog box, see your LDAP server administrator to obtain the required information and refer to the online help for directions on how to fill in each of the LDAP fields that CCDM requires.
-

About Retrieving Data with BPR

You can retrieve provisioning information only by using BPR, which stands for Cisco Broadband Provisioning Registrar. BPR automates provisioning and configuration tasks. CCDM supports BPR 2.0. For product information on BPR, refer to:

<http://www.cisco.com/en/US/products/sw/netmgtsw/ps529/index.html>

Retrieving Data with BPR

To retrieve provisioning information by using BPR:

-
- Step 1** From the Configuring CCDM menu, choose Set External Interfaces.
- Step 2** In the Provisioning Information section, click BPR. Follow the online help for directions on how to fill in each of the following BPR fields that CCDM requires:
- RDU Server IP Address
 - RDU Server Port Number
 - Admin Password
-

About Retrieving Data with CNR

You can retrieve provisioning information only by using CNR, which stands for Cisco Network Registrar. CNR is an application that provides scalable Domain Name System (DNS), Trivial File Transfer Protocol (TFTP), and Dynamic Host Configuration Protocol (DHCP) services.

For product information on CNR, refer to:

<http://www.cisco.com/en/US/products/sw/netmgtsw/ps1982/index.html>

If you use CNR, the following caveats apply to the setup:

- Either the CNR GUI or server component must be installed on the CCDM server.
- The CNR server component can be installed on the same server as CCDM or on a different server. If it is installed on a separate server, the CCDM server must know the location of that server.

- CCDM currently works with one CNR at a time.
- CCDM supports CNR 5.0.3 or a later release.

Retrieving Data with CNR

To retrieve provisioning information by using CNR:

- Step 1** From the Configuring CCDM menu, choose Set External Interfaces.
- Step 2** In the Provisioning Information section, click CNR. Follow the online help for directions on how to fill in each of the following CNR fields that CCDM requires:
- Cluster Name/IP Address
 - Admin User Name
 - Admin Password
 - Client Path
-

About Retrieving Subscriber Data from the Local Database

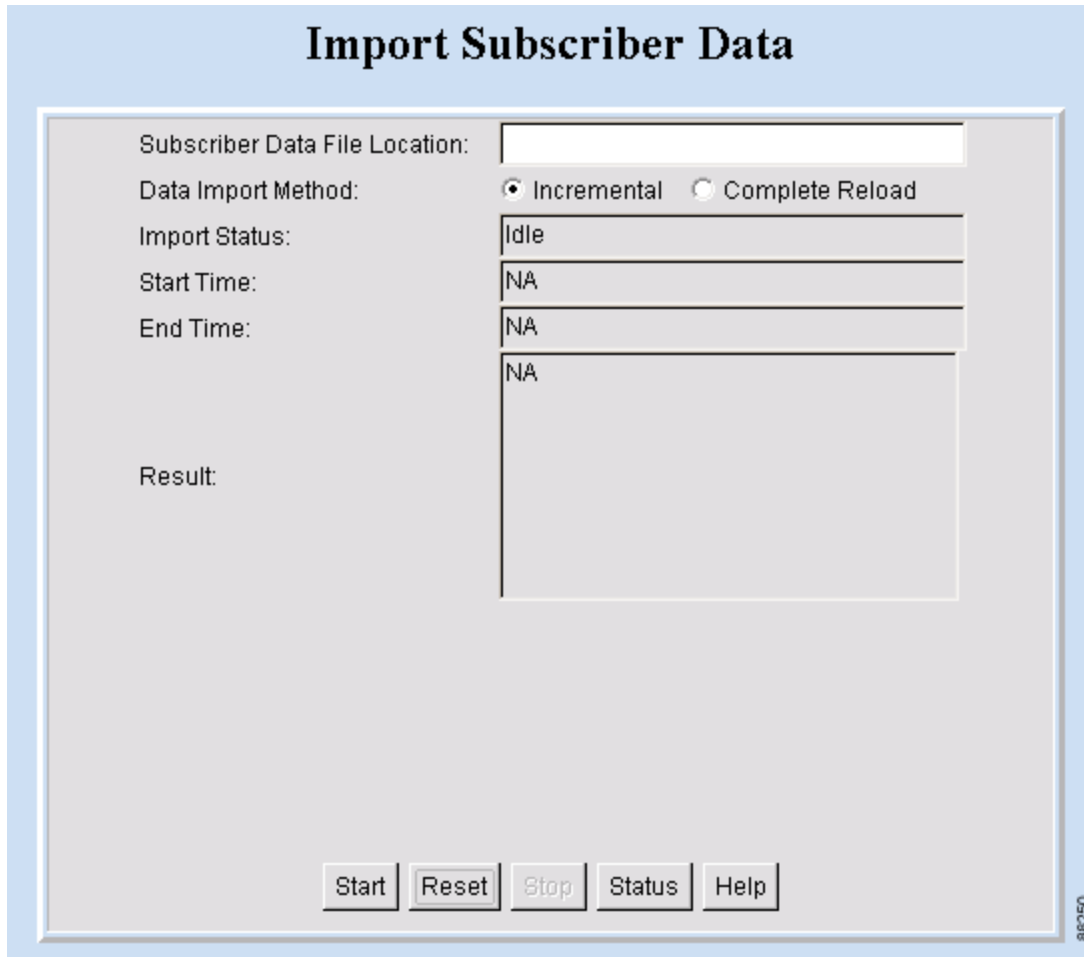
CCDM can retrieve subscriber data from its own local Sybase database after the data has been added in CCDM. After installation, Local Database is the default method for retrieving subscriber data.

Retrieving Subscriber Data from the Local Database

To retrieve subscriber information from the local database:

- Step 1** Add subscriber information by following the directions in “Adding Subscriber Information” page 17.
- Step 2** From the Configuring CCDM menu, choose Set External Interfaces.
- Step 3** In the Subscriber Information section, click Local Database.
- Step 4** (Optional) To update or replace the existing subscriber information, click Import Subscriber to open the Import Subscriber Data dialog box, as shown in Figure 11. In it, follow the online help for directions on how to import a subscriber.
-

Figure 11 Import Subscriber Data Dialog Box



About Retrieving Provisioning Data from the Local Database

CCDM can retrieve provisioning data from its own local Sybase database after the data has been added to it. After installation, Local Database is the default method for retrieving provisioning data. To store provisioning information in its local database, CCDM polls CMTSSs using SNMP. CCDM provides two ways to do this:

- The Poller Scheduler dialog box—Use this method if you prefer to work in the CCDM GUI.
- A separate Java poller application—Use this method, referred to as the Poller, if you prefer to work with a command-line interface.



Tip If you are using CCDM on the Windows OS, Cisco recommends that you use the CCDM GUI method.

About the Poller

When you install CCDM, the default method for retrieving provisioning information is the local database. Because the Poller is part of the process for this method, it is on by default after installation. The default schedule for the Poller is to run at or about midnight and to repeat every 24 hours. You can change these settings according to your own needs.

To query CMTSSs using the Poller, you specify two parameters:

- *delta-start-time*—When the Poller should start to poll the CMTSSs. Expressed in hours, the delta is the difference between when you schedule the polling and when the polling begins.
- *poll-interval*—The interval, expressed in hours, between polling sessions.

The syntax to start the Poller on Linux or Solaris is:

```
/opt/CSCOccdm/bin/start_poller delta-start-time poll-interval
```

Example 4 Starting the Poller

The following example shows a polling session that starts 8 hours after you schedule it and repeats every 24 hours:

```
/opt/CSCOccdm/bin/start_poller 8 24
```

Example 5 Stopping the Poller

The following example shows how to stop the Java poller application on Linux or Solaris:

```
/opt/CSCOccdm/bin/stop_poller
```



Note There is no impact to CCDM if you turn the Poller off.

At each polling interval, the poller accesses the CCDM routers list file and performs SNMP queries to each CMTS using its SNMP read-only community string. The CCDM routers list file is in the following location:

```
/opt/CSCOccdm/jakarta-tomcat-4.0.3/webapps/ROOT/WEB-INF/classes/config/myrouters
```

In addition to the on-demand polling that you schedule, CCDM submits a polling request when it detects that a change has been made to the list of routers that CCDM manages. For example, CCDM submits a polling request when it detects that a new chassis has been added to a router.



Note The more frequent the polling interval, the more SNMP traffic that is generated to the CMTSs. This means that CPU utilization on the CMTS increases when polling is in progress. The impact on the CCDM server side is minimal.

Retrieving Provisioning Data from the Local Database

To retrieve provisioning information from the local database:

-
- Step 1** Add provisioning data by following the directions in “Adding Router Information” page 16.
 - Step 2** From the Configuring CCDM menu, choose Set External Interfaces.
 - Step 3** In the Provisioning Information section, click Local Database.
-

Scheduling the Poller

To schedule when and how often the Poller populates the local database with provisioning information that it gathers from CMTSs, you can use the GUI or type a command.

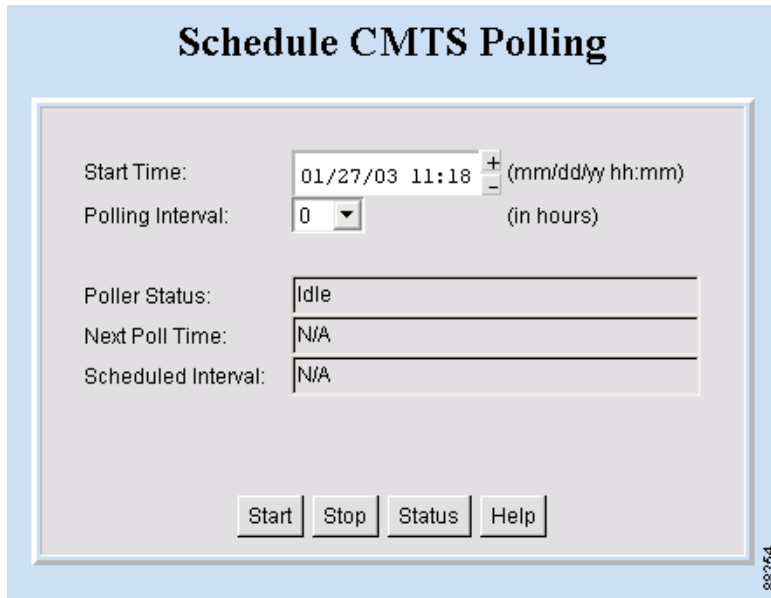
To use the GUI, from the Configuring CCDM menu, choose Schedule CMTS Polling. The Schedule CMTS Polling dialog box appears, as shown in Figure 12. In it, follow the online help for directions on how to schedule the Poller.

or

To use a command, type the following command and specify the parameters in hours, as explained in the previous section:

```
/opt/CSCOccdm/bin/start_poller delta-start-time poll-interval
```

Figure 12 Schedule CMTS Polling Dialog Box



7 Using Southbound Application Program Interfaces in CCDM

CCDM supports two types of interfaces:

- Northbound Interfaces—These interfaces provide two capabilities. They allow:
 - An external application to communicate with CCDM
 - CCDM to communicate with the subscriber and provisioning databases, which can be in CCDM or external to CCDMNorthbound interfaces are discussed in “Retrieving Subscriber or Provisioning Data by Using an External Interface” page 19.
- Southbound Interfaces—These interfaces allow CCDM to retrieve proprietary information from a CMTS that is not from Cisco. This section describes southbound interfaces in CCDM.

CCDM supports three southbound application program interfaces (APIs):

- GetCMMaxCpeNumber—Retrieves the maximum number of customer premises equipment (CPE) devices that can be attached to the cable modem
- GetCMCurrCpeNumber—Retrieves the current number of CPE devices that are attached to the cable modem
- GetFiberNodeTestResult—Retrieves the state of modems on a fiber node

The following parameters are common to these APIs:

- CMTS-IP—IP address of the CMTS you are querying
- UserName—User name to access the CMTS if Telnet or another similar protocol is necessary
- Password—Encrypted user password to access the CMTS if Telnet or another similar protocol is necessary
- Read-Only Community String—Encrypted SNMP read-only community String for SNMP access to the CMTS

CCDM retrieves the values for these parameters from the Manage CMTSs dialog box.

Retrieving Maximum Number of CPE Devices

To retrieve the maximum number of CPE devices that can be attached to a cable modem, use the API `GetCMMaxCpeNumber`. The input parameters, separated by a space, for this API are:

```
script-name GET_CM_MAX_CPE CMTS-IP UserName Password ReadOnlyCommunityString  
non-dotted-modem-mac-address
```

For example, the following API queries the maximum number of CPE devices that can be attached to the modem with the MAC address 000164ffc3c7 on CMTS 145.2.3.4:

```
/opt/tools/CCDM-SB-Script GET_CM_MAX_CPE 145.2.3.4 UserJoe SomeEncryptedPW AnotherEncryptedRO 000164ffc3c7
```

The output of the script must be in the following format:

```
OUT_DATA=CM_MAX_CPE^integer^
```

The following example shows the output for the maximum number of CPE devices for this modem:

```
OUT_DATA=CM_MAX_CPE^20^
```

Retrieving Current Number of CPE Devices

To retrieve the current number of CPE devices that are be attached to a cable modem, use the API `GetCMCurrCpeNumber`. The input parameters, separated by a space, for this API are:

```
script-name GET_CURR_CPE_NUM CMTS-IP UserName Password ReadOnlyCommunityString  
on-dotted-modem-mac-address
```

For example, the following API queries the current number of CPE devices that are be attached to the modem with the MAC address 000164ffc3c7 on CMTS 145.2.3.4:

```
/opt/tools/CCDM-SB-Script GET_CURR_CPE_NUM 145.2.3.4 UserJoe SomeEncryptedPW AnotherEncryptedRO 000164ffc3c7
```

The output of the script must be in the following format:

```
OUT_DATA=CURR_CPE_NUM^integer^
```

The following example shows the output for the current number of CPE devices attached to this modem:

```
OUT_DATA=CURR_CPE_NUM^2^
```

Retrieving Fiber Node Data

To get the state of modems on a fiber node, use the API `GetFiberNodeTestResult`. This API provides the following data for each upstream on each cable modem termination system:

- CMTS IP Address—Address of each CMTS that is associated with the fiber node you are testing.
- Upstream—Port number of each upstream frequency attached to the CMTS.
- Total Modems—Total number of cable modems assigned to the upstream
- Online Count and Percentage—Count and percentage of cable modems on the upstream that are online
- Offline Count and Percentage—Count and percentage of cable modems on the upstream that are offline

The input parameters, separated by a space, are:

```
ScriptName GET_FIBER_NODE_TEST_RESULT CMTS-IP UserName Password ReadOnly-CommunityString  
non-dotted-modem-mac-address nodeName
```

For example, the following API queries the fiber node assigned to the MAC address 000164ffc3c7:

```
/opt/tools/CCDM-SB-Script /GET_FIBER_NODE_TEST_RESULT 145.2.3.4 Joe encrypPasswd encrypROCommString  
000164ffc3c7 FiberNode_SanJose
```

The output format of the script must be in the following format:

```
OUT_DATA=FIBER_NODE_TEST^^NUM_OF_ROWS^FiberNodeName^CMTS_IP^US_DESCRIPTOR^TOTAL_MOD
EM^ONLINE%^OFFLINE%^
```

The following example shows the output for the current number of each upstream on each cable modem termination system in the fiber node:

```
OUT_DATA=FIBER_NODE_TEST^^1^FiberNode_SanJose^172.22.85.7^Cable3/0-upstream5^
1^1(100%)^0(0%)^^
```

Error Handling for Southbound Interface Scripts

If an error occurs within the script, the script should return the output in the following format:

```
OUT_DATA=ERROR^error-message^
```

The following example shows output for an error:

```
OUT_DATA=ERROR^Unable to login device 172.3.4.5^
```

In the GUI, CCDM displays the error message to the user in a message box.

8 Setting Parameters in .INI Files

This section describes the parameters you set in two .INI files in CCDM:

- CONFIGS.INI—Where you set certain system-level parameters
- POLLER.INI—Where you set parameters for the Poller, the separate Java application that polls CMTSs for provisioning data

Parameters in the CONFIGS.INI File

This section describes parameters that you configure in the CONFIGS.INI file, which is located in the following platform-dependent directories:

- Linux and Solaris—/opt/CSCOcdm/jakarta-tomcat-4.0.3/webapps/ROOT/WEB-INF/classes
- Windows—%CSCOCCDM_ROOT%\jakarta-tomcat-4.0.3\webapps\ROOT\WEB-INF\classes



Note When you configure parameters in CONFIGS.INI, you must restart the CCDM server in order for the new settings to take effect.

Phone Number Format

One way to specify a cable modem in CCDM is to search on a subscriber's phone number. You can configure the phoneLength parameter in the CONFIGS.INI file. This parameter determines the number of integers for the phone number format in CCDM. The default setting is 10, as shown below:

```
phoneLength=10
```

You can increase this setting to handle international phone numbers.



Note You can also configure the phone number format in the CCDM GUI. From the Configuring CCDM menu, choose Set System Options.

Modem Status Thresholds

When CCDM displays detailed information on a cable modem, color-coding indicates the modem's status:

- Green—Indicates a normal condition. The modem is online and is within the thresholds that have been specified to indicate proper performance.
- Red—Indicates that an error condition exists. The modem is under or over thresholds that have been specified to indicate proper performance.

The following parameters set the thresholds that determine the boundaries for color-coding and must be configured in the CONFIGS.INI file:

- `upstreamXmitPowerFloor`
- `upstreamXmitPowerCeiling`
- `downstreamSnrFloor`
- `downstreamReceivePowerFloor`
- `downstreamReceivePowerCeiling`

`upstreamXmitPowerFloor`—This parameter determines the minimum upstream transmit power for a cable modem to show a functioning status. This measurement is expressed in decibels millivolt (dBmV). The default setting is 34, as shown below:

```
upstreamXmitPowerFloor=34
```

`upstreamXmitPowerCeiling`—This parameter determines the maximum upstream transmit power for a cable modem to show a functioning status. This measurement is expressed in decibels millivolt (dBmV). The default setting is 52, as shown below:

```
upstreamXmitPowerCeiling=52
```

`downstreamSnrFloor`—This parameter determines the minimum signal-to-noise (SNR) for a cable modem to show a functioning status. SNR is a measure of transmission quality, which is expressed by the ratio of good data (signal) to bad data (noise) that is heard on a line. This ratio is measured in decibels (dB). A higher ratio indicates a better transmission quality. For example, 30 dB is better than 24 dB. The default setting is 30, as shown below:

```
downstreamSnrFloor=30
```

`downstreamReceivePowerFloor`—This parameter determines the minimum downstream receive power for a cable modem to show a functioning status. This measurement is expressed in decibels millivolt (dBmV). The default setting is 15, as shown below:

```
downstreamReceivePowerFloor=15
```

`downstreamReceivePowerCeiling`—This parameter determines the maximum downstream receive power for a cable modem to show a functioning status. This measurement is expressed in decibels millivolt (dBmV). The default setting is 5, as shown below:

```
downstreamReceivePowerCeiling=5
```

Southbound Interface Settings

The following parameters are for using a southbound interface:

`SouthBoundExtIntf`—This parameter specifies the script location of the southbound external interface that CCDM uses to access cable modem termination systems that are not from Cisco to gather information from the device by using a non-standard (i.e. non-DOCSIS SNMP) method. For more information, see “Southbound Script for Accessing CMTSs Not from Cisco” page 63.

`SouthBoundExtIntfTimeout`—This parameter determines when CCDM will time out when executing `SouthBoundExtIntf`. This measurement is expressed in milliseconds. The default setting is 5000, as shown below:

```
SouthBoundExtIntf=5000
```


Modem State Settings

The following parameters are used when you test a modem's state:

ModemStateMaxLife—This parameter specifies the maximum length of time that CCDM keeps the modem state test, once the **ModemStateMaxHash** (see below) has reached its limit. This measurement is expressed in hours. The default setting is 4, which is also the minimum allowed value, as shown below:

```
ModemStateMaxLife=4
```

ModemStateMaxHash—This parameter specifies the maximum number of modem state test entries CCDM keeps before purging old tests that have passed the length of time set in **ModemStateMaxLife** (see above). The default setting is 2000, which is also the minimum allowed value, as shown below:

```
ModemStateMaxHash=2000
```

Query Timeout

The administrator specifies external interface parameters to tell CCDM where to access subscriber and provisioning information. Most of these parameters are in the Set External Interfaces dialog box. However, you must configure the following external interface parameters in the CONFIGS.INI file.

SubscriberExtIntfTimeout—This parameter determines the maximum wait time before CCDM times out during a query for subscriber information. This measurement is expressed in milliseconds (msecs). The default setting is 5000, which is equal to 5 seconds, as shown below:

```
SubscriberExtIntfTimeout=5000
```

ProvisionExtIntfTimeout—This parameter determines the maximum wait time before CCDM times out during a query for provisioning information. This measurement is expressed in milliseconds (msecs). The default setting is 5000, which is equal to 5 seconds, as shown below:

```
ProvisionExtIntfTimeout=5000
```

Parameters in the POLLER.INI File

When it starts, the Poller accesses the POLLER.INI file, which is in the following location:

```
/opt/CSCOccdm/bin/POLLER.INI
```



Note For a detailed description of the Poller, see “About the Poller” page 27.

The poller accesses POLLER.INI to gather the following configurable parameters:

- **maximumThread**
- **maximumDBConnection**
- **SnmpTimeout**
- **SnmpRetry**
- **Debug**
- **maximumAge**
- **PollerServerPort**

maximumThread—This parameter determines the maximum number of different execution threads that can run simultaneously in the Poller. The valid range is 1 to 50. The default setting is 20, as shown below:

```
maximumThread=20
```

maximumDBConnection—This parameter determines the maximum number of CCDM Sybase database connections the Poller uses to perform the polling operation. The valid range is 1 to 7. The default setting is 7, as shown below:

```
maximumDBConnection=7
```

SnmpTimeout—This parameter determines the number of seconds before the attempt to create the SNMP connection will reach a timeout or fail. The valid range is 1 to 3. The default setting is 2, as shown below:

```
SnmpTimeout=2
```

SnmpRetry—This parameter determines the number of times you want to try to create the SNMP connection. The valid range is 1 to 5. The default setting is 3, as shown below:

```
SnmpRetry=3
```

Debug—This parameter determines if debug messages are sent to the log file, which is in the following location:

```
/opt/CSCOccdm/logs/Poller.log
```

Valid values are true and false. The default setting is false, as shown below:

```
Debug=false
```

maximumAge—This parameter tells the poller which outdated records to purge, based on the number of polling intervals that have occurred during which those records have not been updated. This measurement is expressed in polling intervals. A valid range is 1 to 7. The default setting is 2, as shown below:

```
maximumAge=2
```

PollerServerPort—This parameter sets the HTTP server port number where Poll Manager listens to polling requests. The default setting is 8040:

```
PollerServerPort=8040
```



Note If a parameter in POLLER.INI is set out of the valid range, CCDM resets the parameter to its default value.

9 Using and Customizing Troubleshooting Tips

CCDM provides troubleshooting tips to help you diagnose and solve cable modem failure scenarios. These troubleshooting tips are for common cable modem failure scenarios. You can customize troubleshooting tips by adding causes and tips for other failure scenarios.

About Troubleshooting Tips

CCDM provides troubleshooting tips to help you diagnose and solve the following common cable modem failure scenarios:

- **Downstream Frequency Problem**—Related to the frequency that distributes signals from the CMTS to a cable modem
- **Upstream Frequency Problem**—Related to the set of frequencies that send data from a subscriber's cable modem to the CMTS in the headend facility
- **DHCP Server Problem**—Related to the server that dynamically allocates IP addresses to cable modems
- **TFTP Server Problem**—Related to the server that allows you to transfer files to and from remote systems
- **General Causes**—Related to problems that are not isolated to a specific area, such as checking to see that all connections are secure

If a cable modem is not online, you see troubleshooting tips in two places:

- **Real-Time Modem Status Report**—From the Using Hotline Tools menu, choose View a Cable Modem's Status.
- **Test a Cable Modem's State**—From the Using Hotline Tools menu, choose Test a Cable Modem's State.

Table 1 shows the troubleshooting tips for the TFTP server problem.

Table 1 *Troubleshooting Tips for TFTP Server Problem*

Potential Causes	Solution
Incorrect or invalid DOCSIS file is specified. or There are TFTP server issues such as an incorrect IP address or the TFTP server is unreachable. or There is an incorrect router setting in the TFTP configuration.	1. Check against the provisioning server for the accurate TFTP server IP address. 2. Execute an extended ping from the CMTS cable interface (MC line card) to the TFTP server. 3. To understand whether the TFTP server is working or overloaded, set up an emulation to see whether the TFTP server is responding properly with the specific file in question.
The cable modem is still stuck in the TFTP process.	Escalate the problem with TFTP server to the next support tier.

About the TROUBLESHOOT_TIPS.INI File

The TROUBLESHOOT_TIPS.INI file allows you to add troubleshooting tips for other cable modem failure scenarios. The blank TROUBLESHOOT_TIPS.INI file is shown below:

```
[Trouble-shooting Tips]
TFTP_CAUSES=
TFTP_TIPS=
DHCP_CAUSES=
DHCP_TIPS=
US_FREQUENCY_CAUSES=
US_FREQUENCY_TIPS=
DS_FREQUENCY_CAUSES=
DS_FREQUENCY_TIPS=
GENERAL_CAUSES=
GENERAL_TIPS=
```

The file contains a line of causes and tips for each cable modem failure scenario.

When you add a failure scenario and customize the causes and tips in the TROUBLESHOOT_TIPS.INI file, the following format restrictions apply:

- Each cause must have a nonblank corresponding tip specified.
- Each tip must have a nonblank corresponding cause specified.
- CCDM ignores any cause or tip with a corresponding blank.
- CCDM allows three user-defined rows of causes and tips. If you enter additional causes after the third one, CCDM ignores them.



Note One cause or one solution can have multiple parts, as shown in the first row of Table 1. In the first row, the cause has three conditions and the solution has three steps, but CCDM still considers it one cause and one solution.

Modifying the TROUBLESHOOT_TIPS.INI File

To modify the TROUBLESHOOT_TIPS.INI file:


-
- Step 1** Open the TROUBLESHOOT_TIPS.INI file from the appropriate directory for your operating system:
- Linux and Solaris—/opt/CSCOCcdm/jakarta-tomcat-4.0.3/webapps/ROOT/WEB-INF/classes
 - **Windows**—%CSCOCDDM_ROOT%\jakarta-tomcat-4.0.3\webapps\ROOT\WEB-INF\classes
- Step 2** Add a line for *SCENARIO_CAUSES*=, where *SCENARIO* represents the name of the failure scenario you are adding.
- Step 3** For the cable modem failure scenario that you are adding, type the information for its causes according to one of the following formats:
- ^Cause 1a,Cause 1b,Cause 1c^—Use this format when a cause has more than one condition. (See Row 1 of Table 2)
- or
- ^Cause 2^—Use this format when a cause has only one condition. (See Row 2 of Table 2)
-  **Note** You cannot add more than three causes for a failure scenario.
-
- Step 4** Add a line for *SCENARIO_TIPS*=, where *SCENARIO* represents the name of the failure scenario you are adding.
- Step 5** For the cable modem failure scenario that you are customizing, type the information for its respective tips according to one of the following formats:
- ^Tip 1a,Tip 1b,Tip 1c^—Use this format when a tip has more than one step. (See Row 1 of Table 2)
- or
- ^Tip 2^—Use this format when a tip has only one step. (See Row 2 of Table 2)
- Step 6** After you have made all of your modifications, save the TROUBLESHOOT_TIPS.INI file.
- Step 7** Restart the CCDM server for the new settings to take effect.
-

Table 2 Format of Troubleshooting Tips in CCDM GUI

Potential Causes	Solution
Cause 1a. or Cause 1b. or Cause 1c.	1. Tip 1a. 2. Tip 1b. 3. Tip 1c.
Cause 2.	Tip 2.


The following example shows the format for a new cable modem failure scenario in the TROUBLESHOOT_TIPS.INI file.

Example 6 Format of Troubleshooting Tips in TROUBLESHOOT_TIPS.INI File

```
NEW_SCENARIO_CAUSES=^1a) Add cause text.,1b) Add cause text.,1c) Add cause text.^2) Add cause text.^  
NEW_SCENARIO_CAUSES=^1a) 1. Add tip text.,1b) 2. Add tip text.,1c) 3. Add tip text.^2) Add tip text.^
```

10 Viewing Real-Time Modem Status Reports

Figure 13 Real-Time Modem Status Report



Real-Time Modem Status Report

Jan 14, 2003
 5:45 PM America/New_York

[\[Save Report\]](#)
 [\[Modem State\]](#)
 [\[Fiber Node\]](#)
 [\[Provisioning Server\]](#)
 [\[Refresh\]](#)
 [\[Help\]](#)

Modem: [000164ffea91](#) **Modem State:** online

Subscriber Information

Acct ID	Name	Phone	Class Of Service
ID000006	John Doe	1235551212	Policy000006
Address			
300 Main Street San Jose, CA 95134 USA			

Neighborhood Summary

Source	Total	Online (%)	Offline (%)	Selected Modem's State (%)
US	N/A	N/A (N/A%)	N/A (N/A%)	N/A (N/A%)
Cable4/0-downstream	12	11 (92%)	1 (8%)	11 (92%)
Fiber Node: User A000006	3	3 (100%)	0 (0%)	3 (100%)

CMTS Information

IP	Vendor	OS	Type	UpTime
172.22.85.7	Cisco	12.2(11)BC2	7246VXR	3 days 15 hours

Modem Info (from CMTS)

172.22.85.7	
Ping -> successful	
CM IP Address	172.22.127.22
Downstream Channel	Cable4/0-downstream
Upstream Channel	Cable4/0-upstream5
Upstream Receive Power	0.2 dBmV
Maximum Number Of Connected CPEs	1
Current Number of Connected CPEs	0
Service Id	1
Maximum Upstream	0.0 kbps
Guaranteed Upstream	0.0 kbps
Maximum Downstream	0.0 kbps

Modem Info (from CM)

172.22.127.22 (Modem 000164ffc3c7)	
Ping -> successful	
Description	Cisco Internetwork Operating System Software IOS (tm) 120 Software (CVA120-K8V4Y5-M), Version 12.2(2)XA, EARLY DEPLOYMENT RELEASE SOFTWARE (fc1) TAC:Home:SW:IOS:Specials for info Copyright (c) 1986-2001 by cisco Systems, Inc. Compiled Wed 27-Jun-01 0
Device Type	ciscoCVA122E
System Up Time	3 days 21 hours
Downstream Channel Power	-20.0 dBmV
Upstream Transmit Power	23.0 dBmV
Resets	137
Lost Syncs	2
Downstream Signal to Noise Ratio	33.4 dB
CM Config File	gold-e.cm
DHCP	172.22.127.21
TOD	172.22.127.21
TFTP	172.22.127.21

Miscellaneous Modem Info (from Database)

Fiber Node	SanJose_1
User Field	Paid

[\[Save Report\]](#)
 [\[Modem State\]](#)
 [\[Fiber Node\]](#)
 [\[Provisioning Server\]](#)
 [\[Refresh\]](#)
 [\[Help\]](#)

88308

After CCDM is configured according to your needs and the subscriber and provisioning data is accessible, you can review a modem's status. This information is in the Real-Time Modem Status Report, as shown in Figure 13.

CCDM provides two ways to get a cable modem status report:

- View a Cable Modem's Status—If you prefer to work in the CCDM GUI, choose this task from the Hotline Tools menu.
- An external script—If you prefer to work with a command-line interface, use this method.

Viewing a Real-Time Modem Status Report by Using a Script

To get a report using the external script, you specify five parameters:

- **REPORT_TYPE**—This parameter determines the level of detail in the report. Valid settings are **REAL_TIME** or **DETAILED_REAL_TIME**.
- **USER_NAME**—This parameter indicates the CCDM user who runs the report. This user must have the proper privileges to run a report.
- **USER_PASSWORD**—This parameter provides the password of the CCDM user who runs the report.
- **MODEM_MAC**—This parameter determines the MAC address for which the report will provide status.
- **DESTINATION**—This parameter specifies the name of the data file on the CCDM server where you saved the report.

Separate each input parameter with a space. The format for this script is:

```
/opt/CSCOcdm/bin/getStatusReport REPORT_TYPE USER_NAME USER_PASSWORD MODEM_MAC  
DESTINATION
```

For example, the following script retrieves a Cable Modem Real-Time Status Report for the cable modem with the MAC address 001cab2345fe:

```
/opt/CSCOcdm/bin/getStatusReport REAL_TIME admin changeme 001cab2345fe /opt/CSCOcdm/xml/report.xml
```

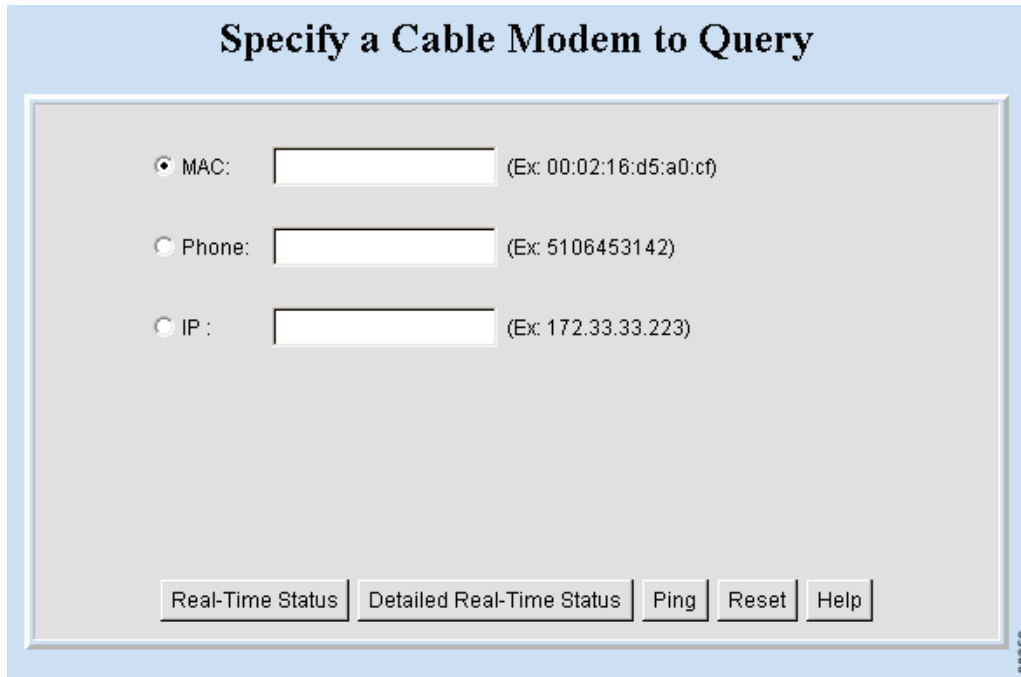
The output of the report is in Extensible Markup Language (XML) format. For details on and a sample of the format, see “Northbound XML Schema and Sample Files” page 55.

Viewing a Real-Time Modem Status Report by Using the GUI

To view a status report for a cable modem by using the GUI:

-
- Step 1** From the Using Hotline Tools menu, choose View a Cable Modem's Status. The Specify a Cable Modem to Query dialog box opens.
 - Step 2** In the Specify a Cable Modem to Query dialog box, shown in Figure 14, follow the online help for directions on how to fill in each field so that you can query a cable modem for its status.
-

Figure 14 Specify a Cable Modem to Query Dialog Box



11 Troubleshooting the Poller

This section provides directions to troubleshoot error messages for the following CCDM components:

- Script—Retrieves subscriber or provisioning information from an external source
- Application on an HTTP server—Retrieves subscriber or provisioning information from an external source
- Poller—Queries CMTSs by using SNMP and stores provisioning information in CCDM's local Sybase database

Troubleshooting Poller Problems

You can use the Poller, a separate Java application, to store provisioning information in CCDM's local Sybase database. This database poller queries the CMTSs by using SNMP. If the Debug parameter in POLLER.INI is set to true, the Poller sends information to the Poller.log file in the following location:

- Linux and Solaris—/opt/CSCOcdm/logs/Poller.log
- Windows—%CSCOCCDM_ROOT%\logs\Poller.log



Note See "Parameters in the POLLER.INI File" page 33 for an explanation of each parameter in POLLER.INI.

This section:

- Shows output from Poller.log that indicates there is a problem
- Describes the cause of the problem
- Provides a resolution for the problem

Error Message: No Router

Problem

The following output from Poller.log indicates that no router was found:

```
Thu Jun 27 15:47:35 EDT 2002: INFO: Start polling...
Thu Jun 27 15:47:37 EDT 2002: INFO: No router found in routers list file:
/opt/CSCOccdm/jakarta-tomcat-4.0.3/webapps/ROOT/WEB-INF/classes/config/myrouters
Thu Jun 27 15:47:37 EDT 2002: INFO: Finished polling... elapsed time: 2371 milliseconds
Thu Jun 27 15:47:37 EDT 2002: INFO: Poller exited.
```

Cause

No router was found in the routers list file because of one of the following conditions:

- The routers list file does not exist. This is the case when you first install CCDM.
- The CMTS routers list is empty; it does not contain any routers.

Resolution

To add routers in CCDM, see “Adding Router Information” page 16.

Error Message: Database Login Failed

Problem

The following output from Poller.log indicates that the login to the database failed:

```
Wed Jun 26 11:14:12 PDT 2002: INFO: Start polling...
Wed Jun 26 11:14:13 PDT 2002: FATAL: SQL Exception in setupDB makeDBConnection: 1
JZ00L: Login failed. Examine the SQLWarnings chained to this exception for the reason(s).
Software aborted.
```

Cause

The database login failure occurs because the database ran out of connections. That is, it tried to use more connections than it can support.

Resolution

If the database ran out of connections, restart CCDM to clear all existing database connections and start the database.



Note For directions on restarting CCDM, see “Installing and Starting CCDM” page 5.

Error Message: SNMP Timeout

Problem

The following output from Poller.log indicates that an SNMP timeout has occurred:

```
Thu Jun 20 13:45:08 EDT 2002: INFO: Start polling...
Thu Jun 20 13:45:18 EDT 2002: ERROR: Thread16 got SNMP exception on Router: 24.216.122.254. Exception:
Snmp Timeout... Router 24.216.122.254 is unreachable
```

Cause

The SNMP timeout occurs when the network connection to the router, or the router itself, is so busy that it cannot satisfy the request within the specified time.

Resolution

If the SNMP timeout occurs too frequently, change the settings in the POLLER.INI file:

To adjust the SNMP timeouts:

Step 1 Open the POLLER.INI file for your operating system:

- Linux and Solaris—/opt/CSCOcdm/bin/POLLER.INI
- Windows—%CSCOCCDM_ROOT%\bin\POLLER.INI

Step 2 Increase one or both of the following settings:

- SnmpTimeout—You can increase the number of seconds before the attempt to create the SNMP connection reaches a timeout.
- SnmpRetry—You can increase the number of times to try to create the SNMP connection.

Step 3 To activate the new settings, restart the CCDM server.



Tip Check for incorrect SNMP community strings because an invalid community string for read/write will also result in an SNMP timeout.

12 Uninstalling CCDM

Uninstalling CCDM on the Linux Platform

- Step 1** Log in as root.
- Step 2** Insert the Cisco Cable Diagnostic Manager CD-ROM into the CD-ROM drive.
- Step 3** To mount the CD, enter:
`/bin/mount /mnt/cdrom`
- Step 4** To change to the CCDM Linux directory, enter:
`cd /mnt/cdrom/linux`
- Step 5** To uninstall CCDM, enter:
`./uninstall`
-

Uninstalling CCDM on the Solaris Platform

- Step 1** Log in as `root`.
- Step 2** Insert the Cisco Cable Diagnostic Manager CD-ROM into the CD-ROM drive.
- Step 3** To change to the CCDM Solaris directory and uninstall CCDM, enter:
`cd /cdrom/cdrom0/solaris`
`./uninstall`
-

Uninstalling CCDM on the Windows Platform

To uninstall CCDM by using the CCDM CD-ROM:

- Step 1** Insert the Cisco Cable Diagnostic Manager CD-ROM into the CD-ROM drive.
- Step 2** From Windows Explorer, double-click `SETUP` in `CDROM_DRIVE\WINDISK1`.
- Step 3** To uninstall CCDM, click Remove.
-

To uninstall CCDM by using Windows menus:

- Step 1** Choose Start > Settings > Control Panel > Add/Remove Programs
- Step 2** Select Cisco Cable Diagnostic Manager.
- Step 3** Click Remove.
-

1.3 Sample Code for Northbound and Southbound APIs

This section contains the README files for the following northbound and southbound application program interfaces:

- Northbound shell script for retrieving subscriber or provisioning information from an external source by using a script
- Northbound Java code for retrieving subscriber or provisioning information from an external source by using an HTTP application
- Northbound XML schema and sample files for saving a real-time modem status report
- Southbound script and code for accessing CMTSs that are not from Cisco to gather network information for CCDM

The sections that describe scripts and code provide the following information:

- Location
- Purpose
- Requirements
- Description of components
- Directions for modifying the code
- Directions for troubleshooting the code
- Sample code that you can modify to meet your needs

The XML section provides the XML schema and a sample file.

Northbound Shell Script for Retrieving Subscriber or Provisioning Information

You can use a sample script written for various databases to retrieve subscriber or provisioning information from an external source. These scripts, that you can modify according to your needs, are in the README file in the following locations:

- Linux and Solaris—`/opt/CSCOcdm/samples/db_script/sybase/`
- Windows—`%CSCOCCDM_ROOT%\samples\db_script\sybase\`

This section provides a printed copy of that README file, formatted for this user guide.

Purpose of Northbound Shell Script

This file describes how to use scripts written against a Sybase database. The sybase directory contains sample scripts that you can modify to meet your needs. These scripts demonstrate how to:

- Receive requests from CCDM
- Query data against a Sybase database
- Return data to CCDM in the format specified in “Retrieving Data with a Script” page 22.

Sample Script Requirements

The sample scripts:

- Require Sybase running with CCDM's database schema, which is created automatically during the CCDM install.
- Are written for the Solaris OS. To use the scripts on a Linux machine, see “Directions for Troubleshooting the Script” page 45.

After trying the sample scripts, you can tailor them for your database or any other external data source.

Script and File Descriptions

The following list describes the purpose of each script or file:

- `get_subscriber`—Queries subscriber information from the database
- `get_provision`—Queries provisioning information from the database
- `insert_cmts`—Invokes `insert_cmts.sql`
- `insert_cmts.sql`—Inserts CMTSs and cable modems into the database and is invoked by `insert_cmts`
- `subscriberinfo.txt`—Contains subscriber information in ASCII text file format
- `add_col_name.pl`—Adds column names to the subscriber fields (for example, Customer Last Name) queried from `get_subscriber`

Directions for Modifying the Script

1. Modify `insert_cmts.sql` to insert a list of CMTSs and cable modems into the database.
2. Execute `insert_cmts` from the command line:
`./insert_cmts`
3. Modify `subscriberinfo.txt` to import subscriber information into CCDM. For field descriptions, refer to online help for the Import Subscriber Data dialog box.
4. Start CCDM. Log in as an Admin user.
5. Click Import Subscriber Data, enter the Subscriber Data File Location, and click Start.
6. Click Set External Interfaces.
7. Under Subscriber Information, select the Script radio button and enter the location of the `get_subscriber` filename (for example, `/my_dir/get_subscriber`) in the Script Location text box.
8. Under Provisioning Information, select the Script radio button and enter the location and the `get_provision` filename (for example, `/my_dir/get_provision`) in the Script Location text box.

Directions for Troubleshooting the Script

- The `get_subscriber` and `get_provision` scripts can be run from the UNIX command line. The following examples show how to execute a script and the output the script produces:

```
./get_provision GET_PROVISION MAC 000216d5a0cf
OUT_DATA=PROVISION^172.22.85.10^172.22.127.26^
```

```
./get_provision GET_PROVISION MAC_LIST 000216d5a0cf^000164ffeb95
OUT_DATA=PROVISION^000216d5a0cf=172.22.85.10,172.22.127.26^000216d5eb95=172.22.85.10,172.22.127.26
```

```
./get_provision GET_MAC IP 172.22.127.26
OUT_DATA=MAC^000216d5a0cf^
```

```
./get_subscriber GET_SUBSCRIBER MAC 000216d5a0cf
OUT_DATA=SUBSCRIBER^AccId=ID000006^Name=Name000006^Phone=6172300006^Address=175 West
Tasman^ClassOfService=Policy000006^FiberNode=User A000006^
```

```
./get_subscriber GET_MAC PHONE 5106663152
OUT_DATA=MAC^000164ffeb95^000164ffc3c7^
```

```
./get_subscriber GET_ADDRESS MAC 000164ffc3c7^000164ffeb95^
OUT_DATA=ADDRESS^000164ffc3c7=170 West Tasman Drive,95134^000164ffeb95=170 West Tasman Drive,95134^
```

- The `get_subscriber` script invokes `add_col_name.pl`. Modify the `scriptDir` to accurately reflect where `add_col_name.pl` is located.
- All scripts use `/tmp` as a temporary directory for creating temp files. Create the `/tmp` directory with read and write privileges for all users.
- If you are running the scripts on a Linux machine, make the following changes to the `get_provision`, `get_subscriber`, and `insert_cmts` scripts:
 - Modify `commLib` to `/bin`
 - Replace `dbisql` to `dbisqlc`
- If you get Invalid Data when using `echo` in Window's batch file:
 - Turn off `echo`:
`@echo off`
 - If you are not using double quotes around the message, then the delimiter has to be `"^^"` instead of `"^"`.
For example:
`echo OUT_DATA=PROVISION^^172.22.85.10^^172.22.127.26^^`

Sample Script Code

The following sample code is extracted from get_subscriber script:

```
#!/bin/ksh
argc=$#
set -A argv $*
integer i=0
while let "i < $argc"; do
    case ${argv[$i]} in
        "GET_MAC") function=${argv[$i]};;
    esac
    let "i = i + 1"
done

#retrieving MAC based on Customer phone number
if [ "$function" = "GET_MAC" ] ; then

    str="SELECT MACAddress FROM SUBSCRIBERINFO where CusPhone='$in_param'";
    echo $str >> "$tmpDir/script.$filenameExt"
    str="OUTPUT TO $tmpDir/test.$filenameExt FORMAT ASCII;"
    echo $str >> "$tmpDir/script.$filenameExt"

    tmpi=`dbisqlc -q -c $dbAccess read $tmpDir/script.$filenameExt`
    rm $tmpDir/script.$filenameExt

    MAC=`$commLib/cat $tmpDir/test.$filenameExt | $commLib/sed s/\'//g`
    echo $MAC > $tmpDir/test.$filenameExt
    MAC=`$commLib/cat $tmpDir/test.$filenameExt | $commLib/sed 's/ /^/g'`

    tmp="OUT_DATA=MAC^$MAC^";
    if [ $tmp = "OUT_DATA=MAC^^" ] ; then
        echo "OUT_DATA=ERROR^No data found for $in_param^";
        exit;
    fi
    echo $tmp

    rm $tmpDir/test.$filenameExt
    exit;

fi
```

Northbound Java Code for Retrieving Subscriber or Provisioning Information

You can use an application that is running on an HTTP server to retrieve subscriber or provisioning information from an external source.

A sample application that you can modify according to your needs is in the README file in the following locations:

- Linux and Solaris—`/opt/CSCOcdm/samples/db_http`
- Windows—`%CSCOCCDM_ROOT%\samples\db_http`

This section provides a printed copy of that README file, formatted for this user guide.

Purpose of Northbound Java Code

This file describes how to use Data Manager, a sample HTTP application. The `http` directory contains this sample application that you can modify to meet your needs. Data Manager demonstrates how to:

- Receive requests from CCDM
- Query data against an Oracle database
- Return the data to CCDM in the format specified in “Retrieving Data with an HTTP Application” page 24.

Sample HTTP Application Requirements

The sample HTTP application:

- Requires the Oracle database running with the schema described in `create_tbls.sql`.
- Is written to use Oracle OCI, instead of the pure Java JDBC, to communicate with the database. To use the sample HTTP application on a Solaris machine, see “Directions for Troubleshooting the Sample HTTP Application” page 50.

Module and File Descriptions

This section lists the modules and files used in Data Manager, the sample HTTP application. For a description, refer to a specific module or file. To modify Data Manager to suit your needs, refer to the next section, “Modifiable Components in the Data Manager Application.”

- Java Modules:
 - `DataManager.java`
 - `Db.java`
 - `DbProps.java`
 - `DbQuery.java`
 - `GetResponse.java`
 - `HttpManager.java`
 - `HttpObj.java`
 - `HttpRequest.java`
 - `HttpResponse.java`
 - `HttpServer.java`
 - `LoadOracleJDBCdriver.java`
 - `Logger.java`
 - `PostResponse.java`
 - `ShutDown.java`

- System and Configuration Files:

- Makefile
- Db-Vital.properties
- Db-Sigma.properties
- install
- start_dm
- stop_dm

- SQL Files:

- create_tbls.sql
- insert_cmts.sql
- insert_subscriber.sql

Modifiable Components in the Sample HTTP Application

- create_tbls.sql, insert_cmts.sql, insert_subscriber.sql—Scripts that create database tables and insert data into the database.
- DbQuery.java—Module that queries the data from the database. The following functions were written based on the schema described in the create_tbls.sql:
 - getMacByIP
 - getMacByPhone
 - getProvInfo
 - getProvInfoExt
 - getFiberNode
 - getSubscriberInfoIf create_tbls.sql is changed, the code for these functions must be changed accordingly.

- Db-Sigma.properties—Contains attributes of the subscriber database.
- Db-Vital.properties—Contains attributes of the provisioning database.

If the provisioning and subscriber data tables reside in the same database, these two files must have identical information. In Data Manager, the provisioning and subscriber data is in two separate databases.

For both files, the following settings must be changed accordingly:

- db_sid
- db_port
- db_userid
- db_password

For more information, contact your local DBA.

- Makefile—Compiles the java files.

start_dm—Starts the Data Manager HTTP application.

The current variable settings in start_dm are:

debug_flag=1

(Send debug messages to a log file.)

port=8012

(Change to the port of your choice.)

logDir= /opt/CSCOccdm/jakarta-tomcat-4.0.3/logs

(Specify the location where the log file should be created and changes should be written.)

stop_dm—Stops the Data Manager HTTP application.

The following variable settings are in stop_dm:

ip=171.71.50.52

(IP address where the HTTP application is running.)

port=8012

(Port in which the HTTP application is running. This should be the same port specified in the start_dm script.)

For these three scripts (Makefile, start_dm, and stop_dm), the following properties need to be updated according to the system environment:

- ORACLE_HOME
- JAVA_HOME
- LIBRARY_PATH

For more information, check with your DBA.

Directions for Building and Running the Sample HTTP Application

1. Build class files and copy files to destination.

From the command line, type:

```
./make
```

```
./install
```

2. Start the application.

From the command line, type:

```
/opt/CSCOccdm/bin/start_dm
```

3. Stop the application.

From the command line, type:

```
/opt/CSCOccdm/bin/stop_dm
```

4. Link Data Manager to CCDM.

- a. Start CCDM. Log in as an Admin user.

- b. Click Set External Interfaces.

- c. Under Subscriber Information, select the HTTP radio button and enter the URL for the application in the URL Location text box:

```
http://your_machine_name:port_num/DataManager
```

5. Under Provision Information, select the HTTP radio button and enter the URL for the application in the URL Location text box:

```
http://your_machine_name:port_num/DataManager
```



Note

NOTE: *port_num* is the port number that is configured in start_dm.

Directions for Troubleshooting the Sample HTTP Application

- Failed to create database connection.

Make sure there is connectivity between the machine and the Oracle database. A quick way to test this is to invoke Oracle's sqlplus application. For example, assume that the following settings are in Db-Vital.properties:

- `db_sid=CCDM`
- `db_userid=user1`
- `db_password=myspasswd`

From the command line, type:

```
sqlplus user1/mypasswd@CCDM
```

If there is no response or an error occurs, check the tnsnames.ora setting with your local DBA.

- Makefile failed - javac not found

Make sure that JAVA_HOME in the Makefile points to the correct JAVA compiler.

- Makefile failed while linking or missing Oracle library error generated during run time.

The sample HTTP application was written to use the Oracle OCI protocol for communicating with the database. Oracle OCI requires machine-dependent libraries, so make sure that the path to the libraries is set correctly in Makefile.

- Socket creation error.

This error occurs when `port_num` is changed in `start_dm` and `stop_dm`. To correct it, restart the program `/opt/CSCOccdm/bin/start_dm`

- DbQuery returns error status.

It is possible that some of the Oracle environment did not get set up correctly. To check this, review `oracle.csh`, update it as required, and source it. Then, run `start_dm` again.

Sample Java Code

1. Setup Http Connection to receive request from CCDM (HttpManager.java)

```
HttpRequest = new HttpRequest()
```

2. Fetch message's header (HttpRequest.java):

```
StringBuffer sb = new StringBuffer();
InputStream is = sock.getInputStream();
DataInputStream fromBrowser = new DataInputStream(is);
while (true) {
    msg = fromBrowser.readLine();
    if (msg.equals("")) break;
    sb.append(msg + nl); // put back the '\n\r' for newline
}
```

3. Fetch message's body (HttpRequest.java):

```
DataInputStream fromBrowser = sock.getInputStream();
StringBuffer sb = new StringBuffer();

for (int x=0; x < len; x++) {
    // return int range 0-255, -1: end of stream
    i = fromBrowser.read();
    if (i == -1) break;
    // however, char is 16 bits for unicode 2 bytes
    // now use 2 bytes to hold just one byte data.
    sb.append((char)i);
    //System.out.println (sb);
}
```

4. Parse message (HttpObj.java):

```
//parse and save message into Hashtable

String req_command = decodeString(command);
//System.out.println ("req_command=" + req_command);

// Parse the data from the servlet
StringTokenizer rdata = new StringTokenizer(req_command, "&");

if (rdata == null) return;
```

```

//System.out.println ("data=" + data + ", rdata=" + rdata);
while (rdata.hasMoreTokens()) {
    String cmd = rdata.nextToken();
    int index = cmd.indexOf("=");
    if (index == -1) return;
    String key = cmd.substring(0, index);
    String val = cmd.substring(index+1);
    hParams.put(key, val);
}

```

5. Process message (HttpManager.java & DataManager.java):

```

//if POST message then process the message:
if (req.getReqType().equals ("POST")) {
    req.fetchBody(sock_);
    hParams = req.getParams();
    resp = new PostResponse(req);
    String status = null;
    status = dm_.processCommand(req);
...
}
//examine the request type and process the message accordingly:
Hashtable hParams = req.getParams();
String task = (String)hParams.get("REQUEST");
int status = 1;

if (task.equals(GET_PROV_INFO)) {
    String sStr = "";
    String sType = (String)hParams.get("SEARCH_TYPE");
    if (sType.equals("MAC"))
    {
        String sTmp = (String)hParams.get("IN_PARAM");
        String sMac = sTmp.toLowerCase();
        Logger.debug("In Mac is " + sMac);
        sStr = DbQuery.getProvInfo(sMac);
        Logger.debug("Prov Info is " + sStr);
    }
.....
}

// retrieve information from the database (DbQuery.java):
public static String getProvInfo(String Mac) {

```

```

String query = "";
String subscriberId = "";
String modemIp = "";
String cmtsIp = "";
Statement stmt = null;

try {
    long start = (new java.util.Date()).getTime();

    start = (new java.util.Date()).getTime();
    stmt = vital_db_conn.createStatement();
    query = "select subscriberid, ipaddress,giaddr from device where deviceid = '"
+ Mac + "'";
    Logger.debug(query);
    boolean bFound = false;
    ResultSet rs = stmt.executeQuery(query);
    while (rs.next()) {
        bFound = true;
        subscriberId = rs.getString(1);
        modemIp = rs.getString(2);
        cmtsIp = rs.getString(3);
        rs.clearWarnings();
    }
    rs.close();
    rs = null;
    stmt.close();
    stmt = null;

    long end = (new java.util.Date()).getTime();
    Logger.debug("total msec to execute getProvInfo: " + (end - start));
    if (!bFound)
        return ("OUT_DATA=ERROR^" + Mac + " does not exist in the Provisioning
Database.^");
    String sStr = "OUT_DATA=PROVISION^" + cmtsIp + "^" + modemIp + "^";
    return (sStr);
}
catch (SQLException se) {
    String sError = "OUT_DATA=ERROR^" + se.getMessage() + "^";
    if (stmt != null)
    {
        try {
            stmt.close();
        }
    }
}

```

```

        catch (Exception e)
        {
        }
        stmt = null;
    }
    return sError;
}
}

```

6. Build a Post Http Response to return data to CCDM (HttpResponse.java):

```

//send header to CCDM
OutputStream os = null;
try {
    os = sock.getOutputStream();
}
catch (IOException e) {
    Logger.error("DataManager.HttpResponse:sendHeader: error: " + e);
    e.printStackTrace(DataManager.getLogPW());
    e.printStackTrace();
}

StringBuffer sb = new StringBuffer();
sb.append ("HTTP/1.0 200 OK" + nl);
sb.append ("Content-type: text/html" + nl);
byte[] b = (sb.toString()).getBytes();
try {
    os.write(b);
    os.flush();
}
catch (IOException e) {
    Logger.error ("DataManager.HttpResponse:sendHeader: error write: " + e);
    e.printStackTrace(DataManager.getLogPW());
    e.printStackTrace();
}
}

```

7. Send result data off to CCDM

```

os.write(result_data);
os.flush();

```

Northbound XML Schema and Sample Files

CCDM is a diagnostic tool that is deployed in an environment where a trouble-ticket system exists. Therefore, it is helpful to transfer the diagnostic information collected by CCDM to a trouble-ticket system. This functionality is provided by using XML to encode the diagnostic information.

When you view a Real-Time Status Report for a cable modem, you can view a detailed version of the report. From the detailed version, you can click Save Report. CCDM saves the report in XML format.

This section provides:

- XML schema file—Defines the content used in the real-time status report
- XML sample file—Shows an example of a real-time status report that conforms to the schema

XML Schema File

The XML schema file in this section defines the content used in the real-time status report:

```
<?xml version="1.0" encoding="UTF-8"?>
<ccdm>
  <input type="macAddress">00:02:16:d5:a4:27</input>

  <cmts>
    <fqdn>Horsham_ubr_1.cisco.com</fqdn>
    <ipaddress>10.87.117.2</ipaddress>
  </cmts>

  <modem>
    <macaddress>00:02:16:d5:a4:27</macaddress>
    <ipaddress>10.1.4.27</ipaddress>
  </modem>

  <test name="cmts-snmp-connectivity" status="successful">
    <status-text>test completed successfully</status-text>
    <start-time>09/04/02 12:00:36 EDT</start-time>
    <finish-time>09/04/02 12:00:39 EDT</finish-time>

    <result-set>
      <result>
        <name>CM IP Address</name>
        <value>10.1.4.27</value>
      </result>

      <result>
        <name>DownStreamChannel</name>
        <value>Cable3/0downstream</value>
      </result>

      <result>
```

```
<name>UpStreamChannel</name>
  <value>Cable3/0upstream1</value>
</result>

<result>
  <name>Upstream Receive Power</name>
  <value>1.0</value>

  <unit>dBmV</unit>
</result>

<result>
  <name>Modem Status (DOCSIS)</name>
  <value>online</value>
</result>

<result>
  <name>Maximum Number of Connected CPE's</name>
  <value>4</value>
</result>

<result>
  <name>Current Number of Connected CPE's</name>
  <value>1</value>
</result>

<result>
  <name>Service Id</name>
  <value>2</value>
</result>

<result>
  <name>Maximum Upstream</name>
  <value>0.0</value>

  <unit>kbps</unit>
</result>

<result>
  <name>Maximum Downstream</name>

  <value>0.0</value>
  <unit>kbps</unit>
```



```

        </result>

        <result>
            <name>CPE IPs</name>
            <value>192.168.2.118, 192.168.2.119</value>
        </result>
    </result-set>
</test>
</test-set>
</ccdm>

```

Sample XML File

The sample XML file in this section shows a real-time status report that conforms to the schema in the previous section:

```

<?xml version="1.0" encoding="UTF-8" ?>
_ <ccdm>
_ <cmts>
<fqdn>172.22.85.7</fqdn>
<ipaddress>172.22.85.7</ipaddress>
</cmts>
<input type="MAC">000164ffeb95</input>
_ <modem>
<macaddress>000164ffeb95</macaddress>
<status>online</status>
</modem>
_ <test-set>
_ <test name="neighborhood-lookup" status="successful">
<status-text>test completed successfully</status-text>
<start-time>Jan 15, 2003 12:14 PM America/Los_Angeles</start-time>
<end-time>Jan 15, 2003 12:14 PM America/Los_Angeles</end-time>
_ <result-set>
_ <result>
<name>Cable3/0-upstream5 total modems count(percentage)</name>
<value>1</value>
<name>Cable3/0-upstream5 online modems count(percentage)</name>
<value>1 (100%)</value>
<name>Cable3/0-upstream5 offline modems count(percentage)</name>
<value>0 (0%)</value>
<name>Cable3/0-upstream5 online count(percentage)</name>
<value>1 (100%)</value>
</result>
_ <result>

```

```

<name>Cable3/0-downstream total modems count(percentage)</name>
<value>1</value>
<name>Cable3/0-downstream online modems count(percentage)</name>
<value>1 (100%)</value>
<name>Cable3/0-downstream offline modems count(percentage)</name>
<value>0 (0%)</value>
<name>Cable3/0-downstream online count(percentage)</name>
<value>1 (100%)</value>
</result>
_ <result>
<name>Fiber Node total modems count(percentage)</name>
<value>N/A</value>
<name>Fiber Node online modems count(percentage)</name>
<value>N/A (N/A%)</value>
<name>Fiber Node offline modems count(percentage)</name>
<value>N/A (N/A%)</value>
<name>Fiber Node online count(percentage)</name>
<value>N/A (N/A%)</value>
</result>
</result-set>
</test>
_ <test name="cmts-info" status="successful">
<status-text>test completed successfully</status-text>
<start-time>Jan 15, 2003 12:14 PM America/Los_Angeles</start-time>
<end-time>Jan 15, 2003 12:14 PM America/Los_Angeles</end-time>
_ <result-set>
_ <result>
<name>IP</name>
<value>172.22.85.7</value>
</result>
_ <result>
<name>Vendor</name>
<value>Cisco</value>
</result>
_ <result>
<name>OS</name>
<value>12.2(11)BC1b</value>
</result>
_ <result>
<name>Type</name>
<value>7246VXR</value>
</result>
_ <result>

```

```
<name>UpTime</name>
<value>8 days 0 hours</value>
</result>
</result-set>
</test>
_ <test name="cmts-icmp-ping" status="successful">
<status-text>test completed successfully</status-text>
<start-time>Jan 15, 2003 12:14 PM America/Los_Angeles</start-time>
<end-time>Jan 15, 2003 12:14 PM America/Los_Angeles</end-time>
</test>
_ <test name="cmts-snmp-connectivity" status="successful">
<status-text>test completed successfully</status-text>
<start-time>Jan 15, 2003 12:14 PM America/Los_Angeles</start-time>
<end-time>Jan 15, 2003 12:14 PM America/Los_Angeles</end-time>
_ <result-set>
_ <result>
<name>CM IP Address</name>
<value>172.22.127.18</value>
</result>
_ <result>
<name>Downstream Channel</name>
<value>Cable3/0-downstream</value>
</result>
_ <result>
<name>Upstream Channel</name>
<value>Cable3/0-upstream5</value>
</result>
_ <result>
<name>Upstream Receive Power</name>
<value>1.5</value>
<unit>dBmV</unit>
</result>
_ <result>
<name>Modem Status</name>
<value>online</value>
</result>
_ <result>
<name>Maximum Number Of Connected CPEs</name>
<value>1</value>
</result>
_ <result>
<name>Current Number of Connected CPEs</name>
<value>0</value>
```

```

</result>
_ <result>
<name>Service Id</name>
<value>1</value>
</result>
_ <result>
<name>Maximum Upstream</name>
<value>0.0</value>
<unit>kbps</unit>
</result>
_ <result>
<name>Guaranteed Upstream</name>
<value>0.0</value>
<unit>kbps</unit>
</result>
_ <result>
<name>Maximum Downstream</name>
<value>0.0</value>
<unit>kbps</unit>
</result>
</result-set>
</test>
_ <test name="cm-icmp-ping" status="successful.">
<status-text>test completed successfully</status-text>
<start-time>Jan 15, 2003 12:14 PM America/Los_Angeles</start-time>
<end-time>Jan 15, 2003 12:14 PM America/Los_Angeles</end-time>
</test>
_ <test name="cm-snmp-connectivity" status="successful">
<status-text>test completed successfully</status-text>
<start-time>Jan 15, 2003 12:14 PM America/Los_Angeles</start-time>
<end-time>Jan 15, 2003 12:14 PM America/Los_Angeles</end-time>
_ <result-set>
_ <result>
<name>Description</name>
<value>Cisco Internetwork Operating System Software IOS (tm) 925 Software (UBR925-K8SV4Y5-M),
Version 12.2(2)XA, EARLY DEPLOYMENT RELEASE SOFTWARE (fc1) TAC:Home:SW:IOS:Specials for info
Copyright (c) 1986-2001 by cisco Systems, Inc. Compiled Wed 27-Jun-01</value>
</result>
_ <result>
<name>Device Type</name>
<value>ciscoUBR925</value>
</result>
_ <result>
<name>System Up Time</name>

```

```
<value>16 days 1 hours</value>
</result>
- <result>
<name>Downstream Channel Power</name>
<value>-20.0</value>
<threshold-floor>-15</threshold-floor>
<threshold-ceiling>5</threshold-ceiling>
<unit>dBmV</unit>
</result>
- <result>
<name>Upstream Transmit Power</name>
<value>6.0</value>
<threshold-floor>34</threshold-floor>
<threshold-ceiling>52</threshold-ceiling>
<unit>dBmV</unit>
</result>
- <result>
<name>Resets</name>
<value>109</value>
</result>
- <result>
<name>Lost Syncs</name>
<value>22</value>
</result>
- <result>
<name>Downstream Signal to Noise Ratio</name>
<value>33.4</value>
<unit>dB</unit>
</result>
- <result>
<name>CM Config File</name>
<value>gold-e.cm</value>
</result>
- <result>
<name>DHCP</name>
<value>172.22.127.17</value>
</result>
- <result>
<name>TOD</name>
<value>172.22.127.17</value>
</result>
- <result>
<name>TFTP</name>
```

```
<value>172.22.127.17</value>
</result>
</result-set>
</test>
_ <test name="subscriber-db" status="successful">
<status-text>test completed successfully</status-text>
<start-time>Jan 15, 2003 12:14 PM America/Los_Angeles</start-time>
<end-time>Jan 15, 2003 12:14 PM America/Los_Angeles</end-time>
_ <result-set>
_ <result>
<name>Acct ID</name>
<value>ID000007</value>
</result>
_ <result>
<name>Name</name>
<value>Multi-user Test</value>
</result>
_ <result>
<name>Phone</name>
<value>6172300006</value>
</result>
_ <result>
<name>Class Of Service</name>
<value>Policy000006</value>
</result>
_ <result>
<name>Fiber Node</name>
<value>User A000006</value>
</result>
_ <result>
<name>User Field</name>
<value>User B000000</value>
</result>
</result-set>
</test>
</test-set>
</ccdm>
```

Southbound Script for Accessing CMTSs Not from Cisco

You can use a script to access CMTSs that are not from Cisco and gather network information that is needed by CCDM.

A sample script that you can modify according to your needs is in the README file in the following locations:

- Linux and Solaris—`/opt/CSCOcdm/samples/SBExtIntfSample`
- Windows—`%CSCOCCDM_ROOT%\samples\SBExtIntfSample`

This section provides a printed copy of that README file, formatted for this user guide.

Purpose of Southbound Script

This file describes how to use a script to interface with CCDM to access CMTSs that are not from Cisco. The script gathers network information from those devices that is needed by CCDM. The sample directory contains a sample script and code that you can modify to meet your needs. The script and code demonstrate how to:

- Receive requests from CCDM
- Decode the encrypted user password and CMTS read-only community string
- Return data to CCDM in the format specified in “Using Southbound Application Program Interfaces in CCDM” page 29.

Sample Southbound Script Requirements

The following requirements apply to the southbound script for accessing CMTSs that are not from Cisco:

- The sample script/code must reside in a directory named `SBExtIntfSample`.
- You must use Java 2 Platform, Standard Edition (J2SE) v1.3.1; it can be downloaded from <http://java.sun.com/j2se/1.3/download.html>.

Script and File Descriptions

The following list describes the purpose of each script or file:

- `ExtIntfDecoder.class`—Java class file that is used to decode the encrypted user-password and CMTS Read-only community string.
- `SampleCode.class`—Pre compiled Java class file of `SampleCode.java` provided.
- `SampleCode.java`—Java code that illustrates how to receive requests and return data to CCDM.
- `SampleScript.sh`—Unix script that can be used to be invoked by CCDM to call `SampleCode.class`.
- `Makefile`—UNIX Makefile to compile `SampleCode.java`.
- `README.txt`—this file.

Directions for Modifying the Script

1. Install Java 2 Platform, Standard Edition (J2SE) v1.3.1. See Sample Script/Code Requirements section on URL location to download, <http://java.sun.com/j2se/1.3/download.html>.
2. Copy `samples/SBExtIntfSample/*` to a new subdirectory `SBExtIntfSample`.
Example: `/usr/test/SBExtIntfSample`
Throughout these directions, `/usr/test/SBExtIntfSample` will be used as the root directory that holds the sample script and code.
3. Change directory to the newly created sample directory.
Example: `cd /usr/test/SBExtIntfSample`

4. Modify *script-root-dir* and *j2se-installed-dir* in Makefile and SampleScript.sh to reflect the:
 - Sample code root directory
Example: /usr/test
 - J2SE installed root directory
Example: /usr/j2sdk1_3_1_04
5. Modify SampleCode.java to include the necessary logic to query the CMTS that is not from Cisco, based on the request passed in from CCDM. For the south-bound interface APIs that need to be supported, see “Using Southbound Application Program Interfaces in CCDM” page 29.
6. Make.
7. Modify SouthBoundExtIntf in CONFIGS.INI to point to the Sample script:
Example: SouthBoundExtIntf=/usr/test/SBExtIntfSample/SampleScript.sh
8. Start CCDM.

Directions for Troubleshooting the Script

None at this time.

14 Obtaining Documentation

These sections explain how to obtain documentation from Cisco Systems.

World Wide Web

You can access the most current Cisco documentation on the World Wide Web at this URL:

<http://www.cisco.com>

Translated documentation is available at this URL:

http://www.cisco.com/public/countries_languages.shtml

Documentation CD-ROM

Cisco documentation and additional literature are available in a Cisco Documentation CD-ROM package, which is shipped with your product. The Documentation CD-ROM is updated monthly and may be more current than printed documentation. The CD-ROM package is available as a single unit or through an annual subscription.

Ordering Documentation

You can order Cisco documentation in these ways:

- Registered Cisco.com users (Cisco direct customers) can order Cisco product documentation from the Networking Products MarketPlace:
http://www.cisco.com/cgi-bin/order/order_root.pl
- Registered Cisco.com users can order the Documentation CD-ROM through the online Subscription Store:
<http://www.cisco.com/go/subscription>
- Nonregistered Cisco.com users can order documentation through a local account representative by calling Cisco Systems Corporate Headquarters (California, U.S.A.) at 408 526-7208 or, elsewhere in North America, by calling 800 553-NETS (6387).

Documentation Feedback

You can submit comments electronically on Cisco.com. In the Cisco Documentation home page, click the Fax or Email option in the “Leave Feedback” section at the bottom of the page.

You can e-mail your comments to bug-doc@cisco.com.

You can submit your comments by mail by using the response card behind the front cover of your document or by writing to the following address:

Cisco Systems
Attn: Document Resource Connection
170 West Tasman Drive
San Jose, CA 95134-9883

We appreciate your comments.

15 Obtaining Technical Assistance

Cisco provides Cisco.com as a starting point for all technical assistance. Customers and partners can obtain online documentation, troubleshooting tips, and sample configurations from online tools by using the Cisco Technical Assistance Center (TAC) Web Site. Cisco.com registered users have complete access to the technical support resources on the Cisco TAC Web Site.

Cisco.com

Cisco.com is the foundation of a suite of interactive, networked services that provides immediate, open access to Cisco information, networking solutions, services, programs, and resources at any time, from anywhere in the world.

Cisco.com is a highly integrated Internet application and a powerful, easy-to-use tool that provides a broad range of features and services to help you with these tasks:

- Streamline business processes and improve productivity
- Resolve technical issues with online support
- Download and test software packages
- Order Cisco learning materials and merchandise
- Register for online skill assessment, training, and certification programs

If you want to obtain customized information and service, you can self-register on Cisco.com. To access Cisco.com, go to this URL:

<http://www.cisco.com>

Technical Assistance Center

The Cisco Technical Assistance Center (TAC) is available to all customers who need technical assistance with a Cisco product, technology, or solution. Two levels of support are available: the Cisco TAC Web Site and the Cisco TAC Escalation Center.

Cisco TAC inquiries are categorized according to the urgency of the issue:

- Priority level 4 (P4)—You need information or assistance concerning Cisco product capabilities, product installation, or basic product configuration.
- Priority level 3 (P3)—Your network performance is degraded. Network functionality is noticeably impaired, but most business operations continue.
- Priority level 2 (P2)—Your production network is severely degraded, affecting significant aspects of business operations. No workaround is available.
- Priority level 1 (P1)—Your production network is down, and a critical impact to business operations will occur if service is not restored quickly. No workaround is available.

The Cisco TAC resource that you choose is based on the priority of the problem and the conditions of service contracts, when applicable.

Cisco TAC Web Site

You can use the Cisco TAC Web Site to resolve P3 and P4 issues yourself, saving both cost and time. The site provides around-the-clock access to online tools, knowledge bases, and software. To access the Cisco TAC Web Site, go to this URL:

<http://www.cisco.com/tac>

All customers, partners, and resellers who have a valid Cisco service contract have complete access to the technical support resources on the Cisco TAC Web Site. The Cisco TAC Web Site requires a Cisco.com login ID and password. If you have a valid service contract but do not have a login ID or password, go to this URL to register:

<http://www.cisco.com/register/>

If you are a Cisco.com registered user, and you cannot resolve your technical issues by using the Cisco TAC Web Site, you can open a case online by using the TAC Case Open tool at this URL:

<http://www.cisco.com/tac/caseopen>

If you have Internet access, we recommend that you open P3 and P4 cases through the Cisco TAC Web Site.

Cisco TAC Escalation Center

The Cisco TAC Escalation Center addresses priority level 1 or priority level 2 issues. These classifications are assigned when severe network degradation significantly impacts business operations. When you contact the TAC Escalation Center with a P1 or P2 problem, a Cisco TAC engineer automatically opens a case.

To obtain a directory of toll-free Cisco TAC telephone numbers for your country, go to this URL:

<http://www.cisco.com/warp/public/687/Directory/DirTAC.shtml>

Before calling, please check with your network operations center to determine the level of Cisco support services to which your company is entitled: for example, SMARTnet, SMARTnet Onsite, or Network Supported Accounts (NSA). When you call the center, please have available your service agreement number and your product serial number.

