Cisco Network Order Manager Solution Guide

Version 1.3

February 2002

Corporate Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 526-4100

Text Part Number: OL-0833-03

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB’s public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

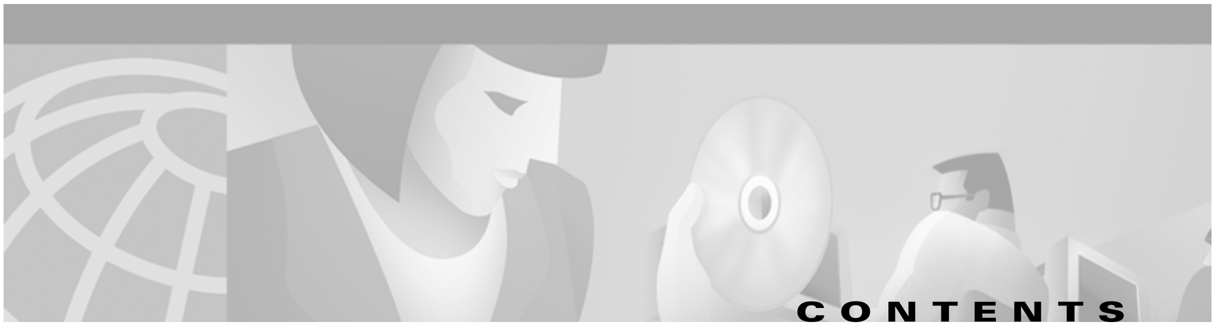
NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED “AS IS” WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

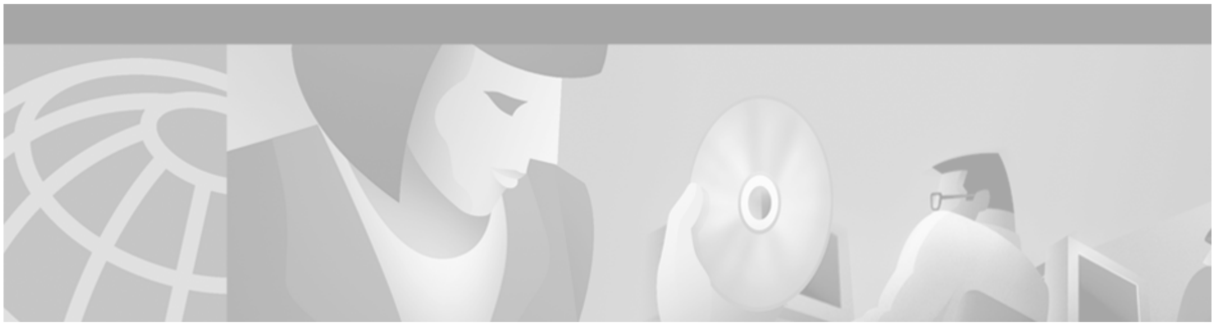
th, AtmDirector, Browse with Me, CCIP, CCSI, CD-PAC, *CiscoLink*, the Cisco *Powered* Network logo, Cisco Systems Networking Academy, the Cisco Systems
ng Academy logo, Cisco Unity, Fast Step, Follow Me Browsing, FormShare, FrameShare, IGX, Internet Quotient, IP/VC, iQ Breakthrough, iQ Expertise, iQ FastT
iQ Net Readiness Scorecard, MGX, the Networkers logo, ScriptBuilder, ScriptShare, SMARTnet, TransPath, Voice LAN, Wavelength Router, and WebViewer are
s of Cisco Systems, Inc.; Changing the Way We Work, Live, Play, and Learn, and Discover All That’s Possible are service marks of Cisco Systems, Inc.; and Airo
PX, Catalyst, CCDA, CCDP, CCIE, CCNA, CCNP, Cisco, the Cisco Certified Internetwork Expert logo, Cisco IOS, the Cisco IOS logo, Cisco Press, Cisco Syster
Capital, the Cisco Systems logo, Empowering the Internet Generation, Enterprise/Solver, EtherChannel, EtherSwitch, FastHub, FastSwitch, GigaStack, IOS, IP/TV
am, MICA, Network Registrar, *Packet*, PIX, Post-Routing, Pre-Routing, RateMUX, Registrar, SlideCast, StrataView Plus, Stratm, SwitchProbe, TeleRouter, and V
trademarks of Cisco Systems, Inc. and/or its affiliates in the U.S. and certain other countries.

trademarks mentioned in this document or Web site are the property of their respective owners. The use of the word partner does not imply a partnership relationsh
Cisco and any other company. (0110R)

Cisco Network Order Manager Solution Guide



CONTENTS



About This Guide

This guide describes how to use the Cisco Network Order Manager to manage DSL and Cisco 6400 Service Connection Manger (SCM) connections. Cisco Network Order Manager is a distributed management system used for setting up connection management from the ingress port of a potentially subtended Digital Subscriber Line Access Multiplexer (DSLAM). The main components of the solution provide the following:

- A simple interface for DSL connection management
- A method for implementing operator-specific connection procedures
- A communications infrastructure that allows element managers to participate in the connection process and which provides an abstracted view of element manager functions

Audience

Use the *Cisco Network Order Manager Solution Guide Version 1.3* to help you manage the DSL connection life cycle and configure network topology and device configuration. The solution leverages a unified object information model and common APIs that are provided by element managers running on the Cisco Element Management Framework (CEMF).

The primary audience of this document is:

- **Component installers**—Assumed to have experience installing telecommunications equipment and cables as well as experience installing data communications equipment and cabling.
- **Network designers/operators/administrators**—Assumed to have experience in telecommunications networks, protocols, and equipment, as well as experience with data communications networks, protocols, and equipment.
- **Project Managers**—Assumed to have some familiarity with telecommunications networks and data networks. Ability to oversee the installation, configuration, and operation of the solution.

Organization

This guide contains the following:

Chapter/Appendix	Description
Chapter 1, "Cisco Network Order Manager Solution"	A general introduction that highlights the functions and properties of the Cisco Network Order Manager Solution.
Chapter 2, "Installing Cisco Network Order Manager"	Describes how to install, remove, and upgrade CNOM software.
Chapter 3, "Working with the Cisco Network Order Manager"	Describes how to perform common CNOM tasks, such as submitting work orders, recovering after a system failure, and hiding connection objects.
Chapter 4, "Using Cisco Network Order Manager in a DSLAM Architecture"	Describes how to use CNOM to configure one or more Cisco DSLAM devices.
Chapter 5, "Cisco 6400 UAC Architecture"	Describes how to create and manage connections between Cisco 6400 Universal Access Concentrators (UACs).
Chapter 6, "Configuring a WAN Environment"	Describes how to create and manage connections in a WAN.
Appendix A, "Cisco Network Order Manager System Configuration"	Describes how to configure the DSL Cisco Network Order Manager.
Appendix B, "Cisco Network Order Manager Policies"	Describes how Cisco Network Order Manager policies work.
Appendix C, "Cisco DSL Manager Error Code Reference"	Defines the causes and troubleshooting for Cisco DSL Manager (CDM) error codes.
Appendix D, "Cisco Service Connection Manager Error Code Reference"	Defines the causes and troubleshooting for Cisco Service Connection Manager (SCM) error codes.
Appendix E, "Cisco Network Order Manager Reference"	Lists the commands and scripts associated with the Cisco Network Order Manager solution.
Index	

Document Conventions

Command descriptions use these conventions:

boldface font	Commands, user entry, and keywords appear in **boldface**.
italic font	Arguments for which you supply values and new terms appear in *italics*.
[ ]	Elements in square brackets are optional.
{x | y | z}	Alternative keywords are grouped in braces and separated by vertical bars.

Examples use these conventions:

`screen` font	Terminal sessions and information the system displays are in `screen` font.
`boldface screen` font	Information you must enter is in **`boldface screen`** font.
< >	Nonprinting characters such as passwords are in angle brackets.
[ ]	Default responses to system prompts are in square brackets.

Notes, cautions, and warnings use these conventions:

Note Means *reader take note*. Notes contain helpful suggestions or references to material not covered in the publication.

Caution Means reader be careful. In this situation, you might do something that could result in equipment damage or loss of data.

Obtaining Documentation

The following sections provide sources for obtaining documentation from Cisco Systems.

World Wide Web

You can access the most current Cisco documentation on the World Wide Web at the following sites:

- http://www.cisco.com
- http://www-china.cisco.com
- http://www-europe.cisco.com

Documentation CD-ROM

Cisco documentation and additional literature are available in a CD-ROM package, which ships 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

Cisco documentation is available in the following ways:

- Registered 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 corporate headquarters (California, USA) at 408 526-7208 or, in North America, by calling 800 553-NETS (6387).

Documentation Feedback

If you are reading Cisco product documentation on the World Wide Web, you can submit technical comments electronically. Click **Feedback** in the toolbar and select **Documentation**. After you complete the form, click **Submit** to send it to Cisco.

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

To submit your comments by mail, use the following address:

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

We appreciate your comments.

Obtaining Technical Assistance

Cisco provides Cisco.com as a starting point for all technical assistance. Customers and partners can obtain documentation, troubleshooting tips, and sample configurations from online tools. For Cisco.com registered users, additional troubleshooting tools are available from the TAC website.

Cisco.com

Cisco.com is the foundation of a suite of interactive, networked services that provides immediate, open access to Cisco information and resources at anytime, from anywhere in the world. This highly integrated Internet application is a powerful, easy-to-use tool for doing business with Cisco.

Cisco.com provides a broad range of features and services to help customers and partners streamline business processes and improve productivity. Through Cisco.com, you can find information about Cisco and our networking solutions, services, and programs. In addition, you can resolve technical issues with online technical support, download and test software packages, and order Cisco learning materials and merchandise. Valuable online skill assessment, training, and certification programs are also available.

Customers and partners can self-register on Cisco.com to obtain additional personalized information and services. Registered users can order products, check on the status of an order, access technical support, and view benefits specific to their relationships with Cisco.

To access Cisco.com, go to the following website:

http://www.cisco.com

Technical Assistance Center

The Cisco TAC website is available to all customers who need technical assistance with a Cisco product or technology that is under warranty or covered by a maintenance contract.

Contacting TAC by Using the Cisco TAC Website

If you have a priority level 3 (P3) or priority level 4 (P4) problem, contact TAC by going to the TAC website:

http://www.cisco.com/tac

P3 and P4 level problems are defined as follows:

- P3—Your network performance is degraded. Network functionality is noticeably impaired, but most business operations continue.
- P4—You need information or assistance on Cisco product capabilities, product installation, or basic product configuration.

In each of the above cases, use the Cisco TAC website to quickly find answers to your questions.

To register for Cisco.com, go to the following website:

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

If you cannot resolve your technical issue by using the TAC online resources, Cisco.com registered users can open a case online by using the TAC Case Open tool at the following website:

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

Contacting TAC by Telephone

If you have a priority level 1 (P1) or priority level 2 (P2) problem, contact TAC by telephone and immediately open a case. To obtain a directory of toll-free numbers for your country, go to the following website:

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

P1 and P2 level problems are defined as follows:

- P1—Your production network is down, causing a critical impact to business operations if service is not restored quickly. No workaround is available.
- P2—Your production network is severely degraded, affecting significant aspects of your business operations. No workaround is available.

CHAPTER 1

Cisco Network Order Manager Solution

This chapter provides an overview of the Cisco Network Order Manager (CNOM), which is a simple-to-use management configuration provisioning tool for the following Cisco element managers:

- SCM (Cisco 6400 Service Connection Manager)
- CDM (Cisco DSL Manager)
- CWM (Cisco WAN Manager)

CNOM insulates the operator from the network and device detail by providing a high-level network order interface for directing the actions of element managers in the network.

This chapter includes the following sections:

- Using the Cisco Network Order Manager Solution, page 1-1
- Supported Element Management Systems, page 1-2
- Cisco Network Order Manager Components and Features, page 1-3
- CNOM Operation, page 1-8

Using the Cisco Network Order Manager Solution

Use the Cisco Network Order Manager (CNOM) to launch the DSL service life cycle: the creation, modification, and deletion of customer services achieved by performing specific network equipment reconfiguration in a specific order. You can use CNOM to manage Layer 2 connections between subtended digital subscriber line access multiplexers (DSLAMs), in a WAN (managed by CWM), and between Cisco 6400 Universal Access Concentrators (UACs).

CNOM can be used to manage the following entities in equipment managed by Cisco SCM:

- ATM connection templates
- NSP PVC cross-connections for subscribers and services
- NRP services and service profiles
- NRP subscriber PVCs

CNOM also supports the management of the following entities in equipment managed by CDM:

- XDSL profiles
- ATM QoS profiles
- PVC and SPVC connections

CNOM supports the following northbound integration methods:

- TCP socket
- CEMF CORBA Gateway

CNOM is made up of a number of processes that work together to execute network management tasks. The processes all use the same API that is used to develop CEMF-based element managers and are able to access all aspects of the CEMF information model. CNOM uses several CEMF capabilities to perform specific activities related to connection management, as shown in Table 1-1.

Table 1-1 CEMF Capabilities and Typical CNOM Activities

CEMF Capability	Typical CNOM Activities
Access/navigate CEMF views	Find the chassis object that contains a specified ATM interface. Find a mapping object with the specified name.
Create/delete CEMF objects	Create profiles which will later be applied to PVC objects.
Read/write attributes	Read topology attributes to determine which interface on another device is connected to a particular port.
Perform EMS user actions	Create a PVC connection within a device.

Supported Element Management Systems

The Cisco Network Order Manager Solution supports the following environments:

- Cisco 6400 UAC—Use this environment to manage a single Cisco 6400 UAC.
- Cisco 6400 UAC to Cisco 6400 UAC—Use this environment to create a network architecture based on the Cisco 6400 UAC. Use this architecture to manage a virtual switch architecture.
- DSLAM—Use this environment to connect valid DSLAM devices from Table 1-2 and valid subtend configurations.
- WAN—Use this environment to connect PVCs through an ATM WAN (BPX) environment.

You can configure CNOM from components, associated line cards, and management software listed in Table 1-2. Cisco Element Management Framework (EMF) allows the individual element management systems associated with CNOM system components to reside within a common framework. CNOM has access to each element manager through CEMF to perform provisioning of a service or a subscriber.

Table 1-2 CNOM System Components

Management System	Device	Description	Module Types and Cisco IOS Releases
Cisco DSL Manager	Cisco DSLAMs	Perform Layer 2 aggregation of DSL traffic into one or more ATM uplinks.	—
	Cisco 6130	Cisco DSLAM	NI-2 (hardware line card)
	Cisco 6160	Cisco DSLAM	NI-2 (hardware line card)
	Cisco 6260	Cisco DSLAM	NI-2 (hardware line card)
Cisco WAN Manager	ATM Cloud	A network of ATM switches managed by CWM.	—
	BPX	An ATM switch.	—

Table 1-2 CNOM System Components (continued)

Management System	Device	Description	Module Types and Cisco IOS Releases
Service Connection Manager	Cisco 6400 UAC	Performs Layer 2 and Layer 3 aggregation of DSL traffic and manages subscribers and services.	For SCM Release 2.2(2): NSP—Cisco IOS Release 12.1(5)DB NRP—Cisco IOS Release 12.1(5)DC NRP2—Cisco IOS Release 12.1(5)DC
—	Gateway	Receives network orders from order management systems through CORBA, or a command line interface.	—
—	CORBA (Common Object Request Broker Architecture)	Used as a standard for object management in which components are objects that can communicate with each other across boundaries such as networks, different operating systems, and programming languages.	—
—	TCP Socket Interface	—	—

Cisco Network Order Manager Components and Features

CNOM provides order management systems with a consistent interface for requesting services when the network equipment that supports the services vary within an operator, and will vary over time. Key CNOM components that support successful operation in complex, evolving environments include:

- **Network Order**—A simple, convenient interface for order management systems to use for DSL connection management which maintains stability when the network architecture changes. A network order is submitted to the *ddmGateway* process to execute a network order policy. When making connections, the network order contains the following elements.
  - A start point and end point
  - Required Quality of Service (QoS)
  - A unique connection identifier
  - Any service-specific attributes for the subscriber
- **Internal Network Order**—The mechanism used to pass information between CNOM processes.
- **Network Order Policy (NOP)**—A method for implementing operator-specific connection procedures that can tolerate varying and evolving network architectures. These are defined in an ASCII file format that represents UML state-machine notation. Each NOP defines one connection management operation for a particular network architecture.
- **Network Order Adapter (NOA)**—A specialized CEMF controller process that connects to specific management modules in CDM, CWM, or SCM. Each NOA interprets the actions of the management module to which it is connected and forwards this information to CNOM. NOAs, which are used in policies:
  - Place device-specific detail inside high-level primitives
  - Communicate with element managers using standard CEMF interfaces
  - Work with a particular underlying EMS, and the code of the EMS is not modified

- **Primitives**—The instructions that are executed in each state in an NOP by a Network Order Adapter (NOA). CNOM provides a set of generic primitives for navigating views, getting and setting attributes, and for deploying and deleting objects. NOAs provide specialized primitives that encapsulate complex operations that are specific to the device (CDM NOA and SCM NOA).
- **Data Recovery**—A feature that:
  - Logs most CNOM network orders
  - Restores the CNOM configuration in case of a system failure
- **Data Synchronization**—The CNOM Synchronization controller is responsible for data integrity checks for all connection objects not created or deleted by CNOM. The CNOMSync controller is informed of all objects and reports those which should be managed by CNOM, but were deployed or deleted by other means. CNOM reports deployment activity in the following format:
  - Date
  - Object Name
  - Parent Name
  - Class Name
  - Action (Created/Deleted)
- **Infrastructure Library**—A component that:
  - Provides support for the key features listed above and is linked into each NOA. The NOAs perform a process that mediates between the detailed device operations involved in connection management and high-level primitives, which are the building blocks of NOPs.
  - Supports the definition of policies in a file. Each NOA reads this library at startup and whenever changes are made.
- ***ddmGateway* process**—A special NOA that receives network orders from order management systems through the Common Object Request Broker Architecture (CORBA) or through a TCP port, controls their execution by the CNOM system, and receives notification of network order completion.

Note CNOM does not have any central managing process. The *ddmGateway* process maintains a persistent store of in-process network orders and logs their completion details. The internal network order is used to pass information between NOAs during policy execution and to maintain state information.

Network Order Policies

CNOM carries out processes defined in network order policies (NOPs). Each NOP defines the work flow required to implement a particular activity, such as adding a profile or connecting a subscriber to service. Each NOP uses an ASCII representation of UML state-machine notation to describe a work flow that implements the required network activity.

NOPs are stored in a set of text files in the directory `<`*`CEMF_ROOT`*`>/config/ddmGateway/policies`. Each CNOM process (including *ddmGateway* and each individual CNOM module) reads all of the policy files at startup, and when you run the CNOM command **policyFileParser**.

An NOP defines the initial state in the policy, and defines a cleanup policy that runs if the policy fails to complete successfully.

The following sections describe the following policy components:

- States
- Transitions
- Primitives
- Aliases
- Cleanup Policy

States

A state defines actions performed at a particular point in the network process. A state contains the following parameters:

- A state name that is unique within the policy
- The target process in which the state is executed
- The name of the primitive (code block) to execute
- Optionally, one or more aliases which allow attributes to be copied within the internal work order

Transitions

Transitions in the policy define the flow of the network process between states. Transitions are either success-based or failure-based. When a primitive completes:

- Successfully—The success transition defines which state to execute next.
- Unsuccessfully—The failure transition causes the policy to try to find alternative routes to complete the policy successfully, or to stop its processing and exit gracefully.

Primitives

Primitives are high-level instructions executed in NOAs during the invocation of a policy. Each state in a policy specifies an NOA and a primitive to run when the state is entered. Primitives can read information from any part of the internal network order, but can only write into the section owned by the NOA that executes the primitive.

A primitive typically oversees functionality that involves many steps, such as:

- Deploying CEMF objects
- Getting and setting CEMF attributes
- Invoking element manager actions

Primitives in NOAs perform simple functions such as setting a CEMF attributes, and they also perform complex functions, such as making PVC connections across subtended DSLAMs. Primitives enable you to develop policies with a minimum knowledge of device-specific detail.

Aliases

Aliases define mappings between attributes in the internal network order. If an attribute is used in a primitive, an alias allows the value of that attribute to be substituted for the value of another attribute. Aliases are most often used to allow primitives to access attribute values supplied in the original network order. They can also be used to copy values from one internal network order attribute to another.

Cleanup Policy

Each policy defines a cleanup policy to be run in case the policy fails (that is, the policy returns to the *dmmGateway* process through a failure transition). The cleanup policy removes any objects that the failed policy may have been created, but not deleted, in CEMF and in the corresponding device configurations, and deletes any cached data from the NOAs relating to that connection.

Network Order Adapters

NOAs are processes that execute primitives to fulfill a network order. NOAs are typically developed to implement primitives specific to a device (for example, a DSLAM or a Cisco 6400 UAC). To develop NOAs, use a library that supplies:

- Awareness of policies
- Work orders
- Primitives

When an NOA starts up, it reads all the policies in the `<CEMF_ROOT>/config/ddmGateway/policies` directory. The NOA then has knowledge of all the policies and the part it plays in each one. States in a policy define the Network Order Adapter that will run the primitive for that state.

Network Order Adapters:

- Are developed using the same API as CEMF element managers and are installed in the same way. This commonality is leveraged to allow NOAs to access the element manager objects, attributes, and user actions to perform device-specific functionality.
- Provide a mediation between the primitives that are used in policy definition and the detailed, device-specific attribute manipulation required to perform the function of the primitives.

ddmGateway Process

The *ddmGateway* process performs the following tasks:

- Receives network orders from external sources
- Creates an internal network order and places the contents of an external network order in it
- Creates a network order object and places it in the *workOrderObjects* view
- Processes network orders in the *workOrderObjects* view in the order received
- Identifies the policy to be invoked from an attribute in the network order being processed
- Identifies, from the policy, the NOA that owns the first state
- Passes the internal network order to the appropriate NOA
- The NOAs complete the execution of the policy
- Receives each internal network order upon completion of a policy
- Signals success or failure to the system that submitted the network order
- Optionally, sends the contents of the internal network order back to the calling system
- Creates a connection object in the *connectionObjects* view
- If a policy fails, the *ddmGateway* process retries according to the retry configuration setting
- If a network order retry fails, the *ddmGateway* process invokes a cleanup policy
- Deletes the original network order object

Profiles

Because CNOM leverages the profile feature of element managers, a network order can include a single `serviceCharacteristics` attribute value that corresponds to the name of profiles in different element managers. A profile is a collection of attribute values that define the operating characteristics of an object in a single step. CEMF element managers use profiles to store sets of attribute values to apply to network objects, such as interface ports and PVCs.

Mapping Objects

In some operational environments, the system sending network orders to CNOM uses its own naming convention for CEMF objects and does not have access to CEMF names. CNOM uses mapping objects to allow network orders to refer to an object using a name other than a CEMF containment name. Mapping objects translate between external names for ports and CEMF object identifiers.

Mapping objects are located in a special view (*ObjectMapView*) which is not visible in the CEMF Map Viewer window. The mapping objects are assigned the names that will be used in network orders, and have an attribute whose value is the CEMF object ID of the CEMF object to which the name refers. To resolve an external name to a CEMF object ID, look up the name in the mapping view, and then get the value of the attribute `LocalDB:DDMMappingObject.targetObject`.

CNOM provides primitives and utilities for mapping objects. The use of the utilities is described in the following chapters in this guide.

From a CEMF shell, enter the following **cemfql** command to list all of the mapping objects in a CNOM system:

```
cd <CEMF_ROOT>/bin
./cemf shell
% cemfql "select objects from containment ObjectMapView: (startLevel=1, endLevel=0)"

Selected 5 objects
            ObjectName

            ObjectMapView:/DDMTopology.lineID/9782441032
            ObjectMapView:/DDMTopology.lineID/4085254903
            ObjectMapView:/DDMTopology.lineID/9493545962
            ObjectMapView:/DDMTopology.lineID/4085436953
            ObjectMapView:/DDMTopology.lineID
```

In the previous example, the mapping objects are used to resolve from a subscriber phone number to the object ID of an ATM interface on a DSLAM.

The mapping object view is categorized by a mapping type (`DDMTopology.lineID` in the example) which allows mapping objects that are used for different purposes to be stored separately. To be able to retrieve a list of mapping objects that refer to a given CEMF object without performing a query on all mapping objects, set an attribute on each target object to contain a comma-separated list of mapping object names. The name of this attribute is specific to the mapping type.

Policy Storage

Policies are stored in a set of text files in the directory `<CEMF_ROOT>/config/ddmGateway/policies`. A single policy file may contain the definitions of multiple policies. Use the **genPolicyDoc** command to generate an HTML representation of policies.

CNOM Operation

The following sections provide background information about the way in which CNOM operates.

- Storing Connection Objects
- Updating CNOM: Data Synchronization
- Data Recovery

Storing Connection Objects

CNOM stores connection objects in two separate views:

- *connectionObjects*—This view is always visible. By default, connections are visible in this view. You can, however, hide connections to improve system performance (see the "Viewing and Hiding Connection Objects" section on page 3-5).
- *connectionObjectsStore*—This view is always hidden, as are the connection objects stored here. This view is used to maintain performance when a system has a large number of connections.

connectionObjectsStore View

In the *connectionObjectsStore* view, CNOM stores connection objects among thousands of containment points (*hash buckets*). CNOM automatically pre-deploys thousands of hash buckets, then uses a hash algorithm to determine which hash bucket to place a newly deployed connection in.

Hash buckets are implemented as a two-layer containment structure. Following is an example of the containment structure for connection objects:

```
connectionObjects:/bucket-1/bucket-1
         .   .   .
                           /bucket-100

connectionObjects:/bucket-2/bucket-1
         .   .   .
                           /bucket-100
         .   .   .
```

Hash Algorithm

CNOM uses a hash algorithm to distribute connection objects evenly among the hash buckets in the connection object store (`connectionObjectsStore:/bucket-`**`x`**`/bucket-`**`y`**). The hash algorithm performs the following steps to determine which hash bucket to place a connection into:

1. CNOM applies the following function to the connection ID string (where `s` is a pointer to a character in the string and `*s` is the value of that character):

    ```
    for (i = 0; s*; s++)
    {
           i = 131 * i + *s;
    }
    i = abs(i);
    ```

2. CNOM divides the result by 100 and the remainder becomes the value of `bucket-`**`x`**.
3. The resulting value is divided by 100 and the remainder becomes the value of `bucket-`**`y`**.

For example, running the hash algorithm on a connection whose ID is `CONN-12345` produces the value `906919850`. Dividing the value by 100 and then dividing the result by 100 produces remainders of `50` and `98`, respectively, which results in the following containment path for the connection object:

```
connectionObjectsStore:/bucket-50/bucket-98/CONN-12345
```

Updating CNOM: Data Synchronization

The CNOMSync controller performs data synchronization to update the CNOM configuration with information about objects you create and delete outside of CNOM (for example, through an element manager like SCM or CDM). Data synchronization ensures that the CNOM configuration matches the actual configuration on the devices being managed. CNOMSync keeps track of all activity that occurs in CEMF and makes changes to the CNOM configuration as necessary (for example, to add or remove connections, profiles, or subscribers).

CNOMSync reads the files in the following directory for information about how to log information about specific objects:

```
/opt/cemf/config/CNOMSync/objectClasses
```

Modifying Logging Information

Each file defines a type of object and provides instructions for logging events related to this object. To modify the logging information for a particular object (for example, to turn off logging or to change the location of the log file), you can use a text editor to make changes to the object class file for that object. For example, to turn off logging for Cisco 6400 subscriber objects created outside of CNOM, edit the Cisco6400Subscriber file and change `enableLog=1` to `enableLog=0`.

After you change object-class files, you must run the following script (from a CEMF shell) to notify CNOMSync of the changes:

```
./updateCNOMSync
```

When you use syslog, it provides a means to log data on a remote machine for use in a recovery process.

Data Recovery

If the CNOM system stops responding and you have to restore the CEMF database from backup, you lose all of the configuration changes between the time of the last backup and the time of the system failure. This means that your CNOM configuration most likely will not match the actual configuration on the physical equipment (DSLAM or UAC) and the corresponding element manager (CDM or SCM). For example, connections and services may have been added on the UAC since the last backup.

CNOM provides data recovery by logging all network orders that modify the CNOM configuration (for example, creating and deleting VCs and services). In the event of a system failure, the system replays those network orders in order to re-create the CNOM configuration as it existed before the failure. See the “Performing Data Recovery to Restore CNOM from Backup” section on page 3-6 for instructions on how to use this feature.

CHAPTER 2

Installing Cisco Network Order Manager

This chapter describes the steps required to install and configure CNOM and the supporting software in a new site. It contains the following sections:

- Prerequisites, page 2-1
- Installing CNOM Software, page 2-2
- Removing CNOM Software, page 2-4
- Upgrading CNOM Software, page 2-4

Note Throughout the procedures in this guide, the term *<CEMF_ROOT>* indicates the directory in which Cisco EMF software is installed. By default, this is the `/opt/cemf` directory.

Prerequisites

The following sections provide important background information for installing CNOM software.

System Requirements

Install the CNOM software on a SUN Ultra 60 server with:

- 9 GB hard disk
- 1 GB memory
- 4 GB swap space

Installation Prerequisites

Before you install CNOM software, make sure the following software is installed on the system. For information about which software releases and patches are required, see the *Release Notes for Cisco Network Order Manager Release 1.3*. Note that the software must be installed in the following order:

1. Cisco EMF
2. Cisco EMF patches

3. The element manager for the type of equipment in the network:
   - SCM for managing Cisco 6400 UACs
   - CDM for managing DSLAMs
   - Cisco WAN Manager (CWM) for managing PVCs in an ATM WAN (BPX)

Note Unlike SCM or CDM, CWM must be installed on a system other than the one on which you are installing CNOM.

For detailed procedures on installing Cisco EMF and Cisco EMF patches, see the *Cisco Element Management Framework Installation and Licensing Guide* (Cisco EMF Release 3.0). For procedures on installing CDM or SCM, see the documentation for the specific element manager.

Installing CNOM Software

This procedure describes how to install CNOM software using the `cnominstall` script. The `cnominstall` installation menu provides choices for installing CNOM in different environments: DSL, Cisco 6400, and CWM. The installation process automatically installs all of the CNOM software components required for each particular environment.

CNOM software is installed in the following directories:

- `<`*`CEMF_ROOT`*`>/bin`—All scripts and utilities
- `<`*`CEMF_ROOT`*`>/lib`—All library files

Note This procedure assumes that you have already installed the Cisco EMF software, any relevant Cisco EMF patches, and the element manager for the equipment in your network (see the "Installation Prerequisites" section on page 2-1).

To run the `cnominstall` script:

Step 1 Log in as superuser on the system on which you plan to install the CNOM software.

Step 2 Create a CNOM installation directory, and go to that directory (for example, `/var/tmp/install/CNOM`).

Step 3 Prepare for installation by copying the CNOM software tar package to the CNOM installation directory. You can download a copy of the CNOM software using Cisco Connection Online (CCO) at either of the following locations:

http://www.cisco.com/cgi-bin/tablebuild.pl/cnom
ftp://ftp.cisco.com/cisco/netmgmt/cnom

Step 4 Uncompress and untar the CNOM software. This process creates several directories and subdirectories (for example, `/packages` and `/packages/ATMDMMc`, `/packages/ATMDMMm`, and `/packages/ATMQOSPc`).

```
uncompress CDDM-1.3.x-pkg.tar.Z
tar -xvf CDDM-1.3.x-pkg.tar
```

Step 5 Make sure that the Cisco EMF is running. If it is not, enter the following command (where <*CEMF_ROOT*> is the directory containing the Cisco EMF software):

```
<CEMF_ROOT>/bin/cemf start
```

Step 6 Start the CNOM software installation process:

```
<cnom_install_dir>/CNOM/cnominstall
```

A menu listing CNOM installation options appears.

Step 7 Select the CNOM software to install:

- DSL Manager
- 6400 Manager
- CWM Manager

The installation process installs all of the CNOM software required for the selected environment.

Note Do not install the DSL Manager and the 6400 Manager on the same system.

Step 8 After you install the CNOM software, enter the following command to make sure that no core dumps occurred:

```
<CEMF_ROOT>/bin/listCores
```

Note To determine the version numbers of installed modules, use the **<*CEMF_ROOT*>/bin/cemf install -s** command.

Although you can use `cemfinstall` to install CNOM software, `cnominstall` is preferred. If you use `cemfinstall`, you must install each of the following CNOM software components separately:

- DDMG Server
- DDM Generic Primitives
- ATMQOSP Manager or Client (CWM installations only)
- ATMDMM Manager or Client (CWM installations only)
- CNOMSync Manager
- The CNOM management software for the type of equipment in your network:
  - CDMDmPkg Manager for Cisco DLSAMs
  - C6400DMM Manager for Cisco 6400 UACs

Installing and Accessing CNOM Man Pages

CNOM man pages install automatically when CNOM is installed. To use the man pages, enter the following commands from the CEMF shell:

```
MANPATH=<CEMF_ROOT>/doc/man:$MANPATH
export MANPATH
```

Now, when you execute the **man** or **xman** command, CNOM man pages appear along with all of the UNIX command man pages.

Removing CNOM Software

Follow the steps in this procedure to remove CNOM software.

Note If you plan to upgrade to a new version of CNOM software, perform the steps in the "Upgrading CNOM Software" section on page 2-4. Do not simply remove the old software and then install the new version. If you do, all CNOM objects in the CEMF database (including connections) will be lost.

Step 1 Log in as superuser on the machine on which the CNOM software is installed.

Step 2 Enter the following command to start the CNOM software removal process:

```
<CEMF_ROOT>/bin/cemfinstall -r
```

The system displays a menu listing currently installed software packages.

Step 3 Select the software package you want to remove and follow the prompts. If you are removing multiple packages, be sure to remove them in the following order:

a. CDM-NOA or SCM-NOA package

b. CNOM infrastructure package

c. Element manager package (SCM or CDM)

d. Cisco EMF patches

e. Cisco EMF

A prompt appears, indicating that the software removal is about to take place. When you proceed with the removal, the system displays informational messages as it deletes the software components.

Upgrading CNOM Software

To upgrade CNOM software, perform the steps in this procedure. You can also upgrade CNOM software manually by running the **<*CEMF_ROOT*>/bin/cemf load -skipportcheck <*package_name*>** command for each CNOM package.

Note When you upgrade from CNOM Release 1.03 or 1.05, or CDM Release 3.1 or later, you must remove the Subtend package before you begin the upgrade.

Step 1 Log in as superuser on the machine on which the CNOM software is installed.

Step 2 Make sure that the Cisco EMF is running. If it is not, enter the following command (where <*CEMF_ROOT*> is the directory that contains the Cisco EMF software):

```
<CEMF_ROOT>/bin/cemf start
```

Step 3 Optional. Use the command **./cemf install -s** to display the current version of all installed software packages.

Step 4 Enter the following command to start the software upgrade process:

```
<cnom_install_dir>/CNOM/cnominstall
```

The system displays a menu listing the software packages available for upgrade.

Step 5 Select the software package you want to upgrade and follow the prompts to upgrade the selected package.

Note If the software package does not already exist on the system, this procedure installs it. However, you must then use the **<*CEMF_ROOT*>/bin/cemf load -remove -skipportcheck** command to remove the package; you cannot use **cemf install -r**.

Step 6 If you are upgrading from CNOM Release 1.0 or 1.2 to CNOM Release 1.3, and your configuration contains connections created in either of those releases, proceed to the next section ("Migrating Existing Connections") for instructions on moving those connections to the new connection object store.

Step 7 If you plan to run existing policies, make sure the policies are compatible with the new CNOM release (see the "Checking Compatibility with Existing Policies" section on page 2-6).

Migrating Existing Connections

After you upgrade from CNOM Release 1.0 or 1.2 to CNOM Release 1.3, you must move existing connection objects to the new connection object store, as described in the following procedure.

Note This procedure is not necessary if you are installing new CNOM software, or upgrading from CNOM Release 1.3 to a later release.

Step 1 Open a CEMF shell:

```
cd <CEMF_ROOT>/bin
./cemf shell
```

Step 2 If you upgraded to SCM 2.2(2) from SCM 2.2(1) or previous releases, make sure the following script was run as part of the data-migration process (see the *SCM 2.2(2) Release Notes* for detailed instructions).

```
./IMTLinkDM
```

Note If the `./IMTLinkDM` script has not already been run, run it now. Otherwise, CNOM policies will fail, causing unexpected results.

Step 3 Issue the following command to migrate existing connection objects:

```
./populateConnectionHash
```

CNOM moves existing connection objects into the new connection object store, which is based on multiple hash buckets, each of which can store thousands of connection objects (see the "Storing Connection Objects" section on page 1-8 for more information).

Step 4 By default, migrated connections are not displayed in the *connectionObjects* view. You can use the policies `MakeConnectionVisible` and `MakeConnectionInvisible` to view the connections and then hide them again. See the "Viewing and Hiding Connection Objects" section on page 3-5 for instructions.

Step 5 Type **`exit`** to close the CEMF shell.

Step 6 CNOM is now ready for operation. See the following chapters for information about using CNOM.

Checking Compatibility with Existing Policies

To ensure that existing policies run properly with the latest CNOM release:

- If the policy contains a globally defined state (for example, `"DDM.failure"`), or a state that is defined in another policy, copy the definition for that state into the policy. State names are no longer required to be globally unique, which means that each state must be defined in the policy in which it is used.
- Make sure the policy calls specific CNOM primitives to deploy and destroy connection objects. Do not use the CEMF `deploy` or `ObjectsDeploy` primitive to create and remove connection objects in CNOM.
- Because connection objects are stored in hash buckets (to improve system performance), you should not use the `GetContainmentPath` primitive (shown below) to find a connection object:

  ```
  GetContainmentPath : path = get(connectionObjects:/<connectionIdentifier>
  ResolveName : connectionOid = resolve(path)
  ```

  Instead, use the following construct to find a connection object:

  ```
  FindConnectionObject : connectionOid = find(connectionIdentifier)
  ```

- It is not necessary to ensure that state names are globally unique. Instead, state names need only be unique within a policy.

CHAPTER 3

Working with the Cisco Network Order Manager

This chapter describes how to perform basic functions using CNOM. It contains the following sections:

- Starting and Stopping CNOM Processes, page 3-1
- Submitting Network Work Orders, page 3-2
- Viewing and Hiding Connection Objects, page 3-5
- Performing Data Recovery to Restore CNOM from Backup, page 3-6
- Suspending CNOM Operation During CEMF Backups, page 3-8

Starting and Stopping CNOM Processes

CNOM processes are started automatically when you launch CNOM. However, it is sometimes necessary to start or stop a CNOM process (for example, to correct problems or to perform troubleshooting), as follows:

Step 1 Log in as super user on the system where CNOM is running.

Step 2 Change directory to the following directory:

```
cd <CEMF_ROOT>/bin
```

Step 3 You can display a list of active CNOM processes by entering:

```
./cemf query
```

Step 4 To start a CNOM process, issue the following command (where *<process_name>* is the name of the process):

```
./sysmgrClient -x <process_name>
```

For example, the following command starts the CNOM gateway process:

```
./sysmgrClient -x ddmGateway
```

Step 5 To stop a CNOM process, issue the following command (where *<process_name>* is the name of the process):

```
./sysmgrClient -k <process_name>
```

For example, the following command stops the CNOM gateway process (ddmGateway):

```
./sysmgrClient -k ddmGateway
```

Submitting Network Work Orders

A network work order is a list of attributes or value pairs set on the *processes:/ddmGateway* object. At a minimum, a network work order contains the following:

- The name *processes:/ddmGateway*
- The name of the policy to execute
- A connection identifier (unique among connections)

Connection, service, and profile management scripts require you to set other attributes. The attributes you set depend on the policy you invoke and on the required features of the policy.

When a network work order appears as arguments to a CNOM command such as **socketDaSet**, you must supply appropriate escape characters to prevent the shell from interpreting the parentheses.

If a network work order command is generated in a different language (for example, a perl script), you must use additional escape characters:

```
$command = "/opt/cemf/bin/socketDaSet processes:/ddmGateway
workOrder:DDMWorkOrder.connectionIdentifier \"(@ARGV[0])\"
workOrder:DDMWorkOrder.workOrderPolicy \"(getATMConnectionData)\"
workOrder:DDM.finalWorkOrder \"()\" ";
system($command);
```

You can submit a network work order to CNOM for processing in any of the following ways:

- Using CNOM Scripts, page 3-2
- Using the CNOM socketDaSet Command, page 3-4
- Using a TCP Socket, page 3-4
- Using the CEMF CORBA Gateway, page 3-5
- Using the C++ API

You can use any or all of these methods on a CNOM system. Network work orders are processed asynchronously, which means that new network work orders may be submitted before previous orders are complete.

Note By default, all CNOM policies call **socketDaSet**. If you create your own policy, make sure the policy includes a call to **socketDaSet**.

Using CNOM Scripts

CNOM provides several scripts that you can use to invoke policies (see Table E-1). CNOM scripts are stored in the directory `<`*`CEMF_ROOT`*`>/bin`.

To run a script, simply enter the name of the script and its command-line arguments at the CEMF command line. The script's command-line arguments are passed to the policy as attribute values. In most cases. each argument corresponds to one of the work-order attributes (`workOrder:DDMWorkOrder.`*`attrb`*) expected by the policy.

To determine a script's command-line arguments, enter the name of the script in the command line. The system displays the command usage, which includes a list of expected work-order attributes and command options. Following is the command usage for the `create_PPPService_fromProfile` script:

```
    ./create_PPPService_fromProfile
```

```
Usage: create_PPPService_fromProfile <NRPContainmentPath> <ProfileName> <Name of
service>  [-noSave|[-syslog] -save <file_path_name>]
```

The arguments displayed in the command usage correspond to the work-order attributes (`workOrder:DDMWorkOrder.`*`attrb`*) expected by the policy. The value you specify for an argument is assigned to that work-order attribute. For example, *`<NRPContainmentPath>`* defines the value of the `workOrder:DDMWorkOrder.NRPContainmentPath` attribute, *`<ProfileName>`* defines the value for `...ProfileContainmentPath`, and *`<Name of Service>`* defines `...serviceName`.

For detailed information about command-line arguments, see the documentation for that element manager (CDM or SCM). The arguments correspond to fields in the GUI windows.

Note When you enter the command-line arguments, be sure to specify the arguments in the correct order, because each value is written to the attribute expected in that position.

Common Arguments for Scripts

Most CNOM scripts contain options to perform common functions, such as hiding connection objects and saving data recovery information to the system log file. To set CNOM to perform a particular function during the execution of a policy, simply include the appropriate option when you run the script to invoke the policy. For example, the following script creates a bridging service from command-line input and saves information about the service to the system log file:

```
./create_B_service ... -syslog
```

Table 3-1 describes the command-line options you can use when you run a CNOM script. Not all CNOM scripts support all options.

Table 3-1 Command-Line Options for Scripts

Option	Description
`-hide`	Hides the connection object created by the `connect` script. The connection is hidden in the *connectionObjects* view to improve system performance. To make a hidden connection visible, run the script `make_visible` *`<connection_ID>`*.
`-nosave`	Tells the system not to save information about the object for data recovery. By default, CNOM saves data recovery information for each created object to use to re-create the CNOM configuration in case of a system failure. If you use this option, the object cannot be recovered after a system failure.

Table 3-1 Command-Line Options for Scripts (continued)

Option	Description
`-save <file_path_name>`	Tells the system to save information about the created object to the specified file, which must already exist. The CNOM data recovery feature uses this information to re-create the object after a CEMF database restore. Use this option if you want to save data recovery information to a file other than the default, which is `<CEMF_ROOT>/logs/connectionsLog.log`. **Note** Make sure the specified file exists and that it has read/write privileges set for root. Also, make sure that there is enough free disk space to store information about the object. Otherwise, the following results occur (in all cases an error message is returned): • Connect scripts do not create the connection. • Disconnect scripts remove the connection. • Other scripts may modify objects.
`-syslog`	Saves information about the created object to the system log file. The CNOM data recovery feature uses this information to re-create the object after a system failure.

Using the CNOM socketDaSet Command

The CNOM command **socketDaSet** submits its arguments in the form of an ASCII message to the *ddmGateway* TCP socket interface on a specified CEMF server.

Use the **socketDaSet** command to:

- Submit network orders from a remote machine
- Test connectivity of remote systems to a CEMF server running CNOM
- Prototype in integration projects

When you use **socketDaSet**, you provide a series of work-order attributes and their values to the *ddmGateway* process. Enclose each attribute value in parentheses and use an escape character (such as quotation marks) to prevent the shell from interpreting the parentheses.

For example, the following input to **socketDaSet** invokes the policy `MakeConnectionVisible` with the necessary attributes. The policy makes visible the connection whose connection ID is CONN-16.

```
/opt/cemf/bin/socketDaSet localhost 10531 processes:/ddmGateway
workorder:DDMWorkOrder:connectionIdentifier “(CONN-16)”
workorder:DDMWorkOrder:workOrderPolicy “(MakeConnectionVisible)”
workorder:DDMWorkOrder:finalWorkOrder “()”
```

Using a TCP Socket

To submit network work orders directly to the *ddmGateway* process:

1. Send the orders to a specified port (10531, by default) on the machine on which CNOM is running. The network order has the same form as the arguments to **socketDaSet**.
2. The *ddmGateway* process starts processing text when it receives a new line character.
3. Responses are sent back to the calling process on the return channel of the socket on which the network order was submitted.

Using the CEMF CORBA Gateway

The CEMF CORBA Gateway provides a way to set attributes on objects in CEMF. You can use it to set network order attributes on *processes:/ddmGateway*. To build a client that can access CEMF services, use the CEMF CORBA Gateway Developer Kit.

Note For projects involving integration of CNOM with other systems, use the TCP socket interface or the CEMF CORBA Gateway.

Viewing and Hiding Connection Objects

By default, connection objects (which represent connections in CNOM) are visible in the *connectionObjects* view, and in other views as defined through CEMF (for example, *componentManaged*). However, you can improve system performance by hiding connection objects in these views, as described in the sections that follow.

Note If you are running CDM, also see the "Adding PVC Objects to the SubtendPVC View" section on page 4-6 for information about how CNOM hides PVC objects to improve system performance, and for instructions on how to make these objects visible (although it is not recommended.)

CNOM provides the following policies, primitives, and scripts for hiding and displaying connections:

- `MakeConnectionVisible`—This policy invokes the `MakeConnectionVisible` primitive to make a hidden connection visible in CNOM views. The `make_visible` script invokes this policy.
- `MakeConnectionInvisible`—This policy invokes the `MakeConnectionInvisible` primitive to hide a connection object in CNOM views. The `make_visible` script with `-hide` option invokes this policy.
- `connect` scripts all contain a `-hide` option that you can include in the command line to hide the connection object created by the script.

Hiding a Connection

You can improve system performance by hiding connection objects in the *connectionObjects* view and in other views, such as *componentManaged*, as defined through CEMF. To do so:

- Include the `-hide` option when you run the `connect` script to create the connection. CNOM creates the connection object but does not display it. (By default, all created connection objects are stored in the *connectionObjectsStore* view, which is a hidden view.)
- Run the following script (where `<connection_ID>` is the ID of the connection you want to hide). To make a connection visible again, run the script without `-hide`.

```
./make_visible <connection_ID> -hide
```

Making a Hidden Connection Visible

Although it is not recommended, it is sometimes necessary to display a hidden connection (for example, to modify the connection). To make a hidden connection visible in *connectionObjects* and other views, run the following script from a CEMF shell (where `<connection_ID>` is the ID of the connection to make visible):

```
./make_visible <connection_ID>
```

Performing Data Recovery to Restore CNOM from Backup

To ensure that you can easily restore your configuration after a system failure, CNOM provides a data recovery feature that logs all network orders that modify the CNOM configuration (for example, creating and deleting VCs and services). In the event of a system failure, the system replays those network orders in order to re-create the CNOM configuration as it existed before the failure.

If the CNOM system fails and you must restore the CEMF database from backup, you lose all of the configuration changes between the time of the last backup and the time of the system failure. For example, if you back up your CEMF data each night at 12 a.m. and experience a system failure at 12 p.m. the following day, you lose any of the configuration changes made between 12 a.m. and 12 p.m. This could result in such inconsistencies as VCs that exist on devices but not in the element manager, or services that exist in CEMF but not on the device.

Note After you restore the CEMF database from backup, run the script `getRestoreWO` to restore the CNOM configuration.

The `getRestoreWO` script re-creates the configuration by re-executing each of the commands in the `<CEMF_ROOT>/logs/connectionsLog.log` file. When you run the script, specify the date and time to restore the data from. For example, if you back up the CEMF database each night at 12 a.m. and a system failure occurs at 12 p.m., you would want the script to re-execute the commands issued between midnight and noon.

The format of the command for invoking the `getRestoreWO` script is as follows:

```
./getRestoreWO [-range {start-range | start} {end-range | end} | -time {start-time |
start} {end-time | end}] <hostname> <socket-portno> [-syslog] <logFileName>
<outputFileName>
```

getRestoreWO option	Description
-range {*start-range* | *start*} {*end-range* | *end*}	The range of configuration commands to re-execute for data recovery. • start and end—Keywords that specify the start and end of the connectionsLog.log file, respectively. • *start-range* and *end-range*—The date and time commands were issued. The system re-executes commands issued during this time. Use the format described in *start-time* and *end-time* below. For example, -range start end re-executes all of the commands in the file, and -range "11/05/2001 00:00:00" end re-executes commands issued between 12 a.m. on November 5, 2001, and the end of the file.
-time {*start-time* | *start*} {*end-time* | *end*}	The time period during which commands were issued. The system re-executes configuration commands issued during this time period. • start and end—Keywords that specify the start and end of the connectionsLog.log file, respectively. • *start-time* and *end-time*—The date and time commands were issued. The system re-executes commands issued during this time. Use the format "*DD/MM/YYYY HH:MM:SS*" (where for HH:MM:SS, 00:00:00 is midnight). For example, -time start end re-executes all of the configuration commands issued between the last CEMF backup and the system failure; and -time "01/01/2001 00:00:00" "02/01/2001 00:00:00" re-executes configuration commands entered between 12 a.m. January 1, 2001 and 12 a.m. January 2, 2001.
<*hostname*>	The name of the CNOM system whose configuration you are restoring.
<*socket-portno*>	The port number of the socket.
[-syslog]	This option tells the command (getRestoreWO) that the following log file (<logFileName>) is a system log file. Use the option when restoring WO from a system log file.
<*logFileName*>	The name of the log file containing the configuration commands to execute to restore the configuration to the state it was in before the system failure. The default is <*CEMF_ROOT*>/logs/connectionsLog.log.
<*outputFileName*>	The name of the file in which the system saves summary information for data recovery.

To re-create specific parts of the configuration, you can open the log file in a text editor (such as vi or emacs) and cut and paste specific commands from the file to the CEMF command line. Note that this might be useful for setting up test configurations or for troubleshooting.

The getRestoreWO script uses the restore and clear policies described in Table 3-2 to re-create the configuration. You can invoke these policies without using the getRestoreWO script if you want (for example, for troubleshooting or for specific configuration needs); however, you should fully understand what you are doing before you attempt it.

Caution Cisco NRP Capacity Statistics might display incorrect values after data recovery. Run the NRP_statistics_fix script to set up valid NRP Capacity Statistics data for all NRPs in the system.

Note See the “Updating CNOM: Data Synchronization” section on page 1-9 for information about data synchronization, a process that updates the CNOM configuration with information about objects created and deleted outside of CNOM (for example, through an element manager like SCM or CDM).

CNOM Restore Policies

Table 3-2 lists the policies CNOM runs to perform data recovery after a system failure. These policies are invoked by the `getRestoreWO` script.

Table 3-2 CNOM Restore Policies

Policy Used to Create Connection...	Policy Used to Restore Connection...
SCM Policies	
`connectSubscriberToATMService`	`restoreSCMATMConnection`
`disconnectSubscriberAndService`	`clearSCMATMConnection`
CDM Policies	
`connectSuscriber_cdm`	`restoreCDMConnection`
`disconnectSuscriber_cdm`	`clearCDMConnection`

Suspending CNOM Operation During CEMF Backups

To back up CEMF data while CNOM is running, CNOM provides two commands for suspending and resuming processing:

- `suspendOperation`—Halts CNOM processing. When you issue this command, CNOM stops accepting new work orders and finishes processing all in-progress work orders. If you attempt to run a CNOM script while processing is suspended, a message appears indicating that the gateway is suspended.
- `resumeOperation`—Restarts CNOM processing after a CEMF backup.

Note If you perform a CEMF backup without first suspending CNOM operation, CNOM scripts fail without displaying a message that the gateway is suspended.

To back up CEMF data while CNOM is running, perform the steps in this procedure:

Step 1 Log in as root on the system on which you want to back up CEMF data, and open a CEMF shell:

```
cd <CEMF_ROOT>/bin
./cemf shell
```

Step 2 Suspend CNOM operation and start the CEMF backup. The system waits until all in-progress work orders are processed before starting the backup.

```
./suspendOperation
./cemf backup
```

Step 3 When the backup is complete, resume CNOM operation:

```
./resumeOperation
```

CHAPTER 4

Using Cisco Network Order Manager in a DSLAM Architecture

This chapter describes how to use the Cisco Network Order Manager to configure one or more Cisco DSLAM devices which may be connected in a subtended configuration.

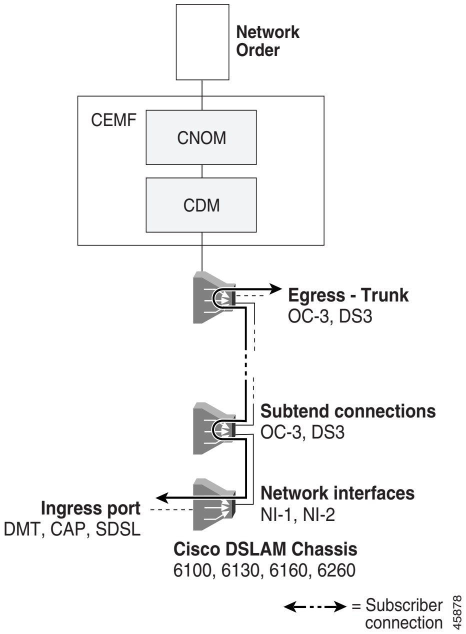
Subtended DSLAM Network Configuration

CNOM supports management of PVC and SPVC connections starting at the ATM interface of a DSL port and terminating at a trunk port. Connections are supported for any combination of chassis and network interface cards, with the following exception:

- NI-2 interface card on a Cisco 6100 chassis

A subtended DSLAM network architecture is illustrated in Figure 4-1. The dotted lines in Figure 4-1 indicate that more DSLAMs can be added to the network.

Figure 4-1 DSLAM-only Network Configuration



Pre-installation Requirements

Before you can use CNOM to configure Cisco DSLAM devices, you must install the following software:

- CEMF Release 3.0.4 and appropriate CEMF patches (see the *Release Notes for Cisco Network Order Manager Release 1.3*)
- Cisco Digital Subscriber Line Manager (CDM)
- Cisco Network Order Manager (CNOM)

Refer to the installation guides for each product for installation procedures. The CNOM installation procedure is described in Chapter 2, "Installing Cisco Network Order Manager."

Post-installation Requirements

After you install CDM and CNOM, use the following procedure to configure the two components to interoperate:

1. Discover and commission DSLAMs and their subrack components.
2. Discover Subtends using the CDM Subtend Manager (refer to the CDM documentation).

Note To make connections on a DSLAM, you must Discover Subtends using the CDM Subtend Manager. This applies to a single chassis as well as subtended chassis.

3. Set up line IDs for the ingress ports using the CNOM `set_line_ids` script (this is an optional step as the containment path can be used instead).
4. Set up profiles of the same name for all interface types and ATM QoS using the CNOM `create_ni2_profile` script.
5. Set up the subscriber DSL connections using the CNOM `connect_cdm` script, which invokes the policy `connectSubscriber_cdm` to set up an end-to-end PVC or SPVC connection across a subtended network of DSLAMs.

   Identify the subscriber DSL port by specifying its containment path (which is defined in CDM), or its line ID (which is an optional identifier created in CNOM). See the "Configuring Line IDs for CNOM Modules" section on page A-3 for more information on line IDs.

Caution When you run policies, be sure that the policy is one designated for the specified application. For example, the policy listed above is specifically designed for use in a CDM environment.

To run the policy successfully, meet the following conditions:

- The DSLAMs are deployed and commissioned.
- The NI-2 profiles are deployed and configured.
- The subtend view is set up correctly. See the "Setting Up a Subtended Hierarchy" section on page A-2 for standards and procedures.
- Optional. The line ID values are set for the line ports. See the "Configuring Line IDs for CNOM Modules" section on page A-3 for more information.

The policy expects the following attributes in the work order:

- `workOrder:DDMWorkOrder.connectionIdentifier`
- `workOrder:DDMWorkOrder.ingressIdentifier`
- `workOrder:DDMWorkOrder.serviceCharacteristics`

- `workOrder:DDMWorkOrder.autoAllocate`
- `workOrder:DDMWorkOrder.setupSoftPVC`
- `workOrder:DDMWorkOrder.ingressVCI`
- `workOrder:DDMWorkOrder.egressVCI`

Managing Profiles

To manage a profile, create a profile which corresponds to each connection metric that is required. The profiles corresponding to the same connection metric must use the same name.

For instance, an operator may provide a service (called Gold) which is characterized by:

- An upstream speed of 1.5 Mbps
- A downstream speed of 384 Kbps

and another service (called Silver) with:

- An upstream speed of 512 Kbps
- A downstream speed of 128 Kbps

To define each type of service, the operator might:

1. Create profiles named Gold and Silver for ATM QoS.
2. Create Gold and Silver profiles for each type of DSL port (for example, DMT and CAP ports on a chassis with NI-2 line cards).
3. Apply the appropriate set of profiles to the CAP or DMT port to configure the port for service:
   - For Gold service on an NI-2 CAP port, apply the NI-2 CAP Gold profile and the ATM QoS Gold profile to the port.
   - For Silver service on an NI-2 DMT port, apply the NI-2 DMT Silver profile and the ATM QoS Silver profile.

In the CNOM work order, the value of `workOrder:DDMWorkOrder.serviceCharacteristics` specifies the appropriate set of profiles to apply; therefore, the value must match the name of the required profile. For example, if `serviceCharacteristics` is set to Gold, the system applies the appropriate DSL-specific Gold profile and the ATM QoS Gold profile.

Note The name of a DSL-specific profile (CAP, DMT, SDSL) must match the name of the corresponding ATM QoS profile; otherwise, the DSL profile is not applied to the connection when the work order is invoked.

Use the following profiles in the connection policy for creating connections in a DSLAM-only architecture:

- NI-2 ATM QoS
- NI-2 CAP
- NI-2 DMT
- NI-2 SDSL

Note An ATM QoS profile is mandatory. The DSL-specific profile is optional; if you do not specify one, the default profile for that port type will be used.

NI-2 profiles consist of DSL parameter profiles and ATM QoS. You can set up the profiles in the CDM GUI, or by using the policies supported by the CDM NOA module.

Note CNOM configures the ADSL and the DMT profiles separately with the GUI, for DMT profile, but configures the ADSL and the DMT profiles together when using the DMT profile policy.

Connectivity in DSLAM-only Network Architecture

To manage connections in a DSLAM-only network architecture, use the following sections as a guide.

Connecting in DSLAM-only Architecture

To make connections in a DSLAM-only network architecture, use the CNOM policy in Table 4-1. You can use the script `connect_cdm` to invoke the policy (see the following section, “Running the connect_cdm Script”).

Table 4-1 DSLAM-only Connection Policy

Policy	Description
`connectSubscriber_cdm`	Creates a PVC from a DSL port to the top trunk in a subtend tree

Caution When you run policies, be sure that the policy is one designated for the specified application. For example, the command listed in Table 4-1 is specifically designed for use in a CDM environment.

For information about the work order interface, refer to Appendix B, “Cisco Network Order Manager Policies.”

When the `connectSubscriber_cdm` policy is invoked, CNOM performs the following steps to create a connection from a DSL port to the top trunk of a subtended tree of DSLAMs.

1. CNOM identifies the DSLAM line port by querying for the line ID attribute on the interface, or by using the name supplied in the incoming work order.
2. CNOM configures the DSL port by applying profiles.
3. CNOM deploys the PVC objects on the appropriate ports in the DSLAMs in subtend path.

Note To create a connection to a DSLAM port that is outside your CEMF configuration, include the `-nsap` option when you invoke the `connect_cdm` script (see the following section).

4. CNOM generates appropriate VPI and VCI values for the intermediate PVCs.
5. CNOM invokes the connect action on the PVC objects to configure the PVCs in the nodes.

Running the connect_cdm Script

The connect_cdm script invokes the connectSubscriber_cdm policy to set up an end-to-end PVC or PVC connection across subtended DSLAMs. The script has the following command-line arguments:

```
connect_cdm [-auto] [-soft] {-line <lineID> | -path <ingress-path>}
[-nsap egressNSAPaddress] <profileName> [<ingressVPI> <ingressVCI>
<egressVPI> <egressVCI>] [-noSave | [-syslog] -save <file_path_name>]]
```

Table 4-2 Command Syntax for connect_cdm Script

Command Option	Description
-auto	Allows CDM to autoallocate VPI/VCI values for the connection. If you do not include this option, you must explicitly define the VPI/VCI values in *<ingressVPI>*, *<ingressVCI>*, *<egressVPI>*, and *<egressVCI>*.
-soft	Creates a soft PVC.
-line *<lineID>*	The line ID of the ingress port for the connection. If the port is not assigned a line ID, use the -path option to specify its path.
-path *<ingress-path>*	The full path of the ingress port for the connection. Use this option if the port is not assigned a line ID.
-nsap *egressNSAPaddress*	The network service access point (NSAP) address of the trunk port to connect the ingress port to. The trunk port can be on a DSLAM that is outside your CEMF configuration (for example, in another provider's network) or it can be part of your configuration. Use the standard NSAP address format. For example: 47.0091.8100.0000.0003.3265.6401.4000.0c98.0010.00
<profileName>	The name of the profile to apply to the connection.
<ingressVPI> <ingressVCI>[1]	The VPI and VCI values for the connection ingress port.
<egressVPI> <egressVCI>	The VPI and VCI values for the connection egress port.

[1] If you include -auto in the command line, the system assigns VPI/VCI values to the connection ingress and egress ports.

Hiding Connections When Using connect_cdm

When you create a CDM connection, you can prevent the connection from being displayed in the *connectionObjects* view (to improve system performance) by including the -hide option in the connect_cdm command line.

If the option Add PVC to SubtendPVC view is On when you run connect_cdm, a PVC object is added to the *SubtendPVC* view to represent the connection. To keep PVC objects from being added to this view (which improves system performance), turn off the option as described in the "Turning Off the Option Add PVC to SubtendPVC View" section on page 4-7. Note that this option is Off by default.

CNOM Internal Work Flow for connectSubscriber_cdm

CNOM performs the following steps to execute the `connectSubscriber_cdm` policy:

Initial State: `ddmGateway:DDM.connectSubscriber_cdm_checkIngressInfo`

1. Determine if the parameter sent is alias or containment path—`NOP`.
2. Find object:
   a. Path—Find the port object name through which connections are made on the DSLAM: `ResolveName`.
   b. Alias—Find the port to which the alias name maps: `FindMappedObject`.
3. Create an object for the connection: `CreateConnectionObject`.
4. Determine the egress port on the DSLAM: `GetTrunk`.
5. Set up transport parameters for the DSL Port: `ConfigureDSLPort`.
6. Connect the PVC, set VPI/VCI: `ConnectPVC`.

Connection Status

If a connection is:

- **Successful**—The **socketDaSet** command returns a success message and creates connection objects in the *connectionObjects* view to represent the connection. If you include the `-hide` option when you create the connection, connection objects are hidden in *connectionObjects*.

  At the root level of *connectionObjects*, an object with the same name as the `workOrder:DDMWorkOrder.connectionIdentifier` value appears. Expand this object to show the objects `CDMDmmConnectObj` and `ATM`.

  Under `CDMDmmConnectObj`, the PVCs set up in this connection are displayed in the CDM NI-2 and PVC windows. If the option Add PVC to SubtendPVC view is On, a PVC is also added to the *SubtendPVC* view. Status windows are available for the connection.

Note For a connection to a DSLAM trunk port that is outside the CEMF configuration (the port is identified by its NSAP address), CNOM creates a PVC object with a single virtual channel link (VCL) child under the ingress DSLAM port.

- **Unsuccessful**—The **socketDaSet** command returns the message “Failed to set the following.” Refer to log files *ddmGateway.audit*, *ddmGateway.log*, *CDMDmmCtlr.log*, and *ATMCMController.log* for more information.

Adding PVC Objects to the SubtendPVC View

By default, PVC objects are not displayed in the *SubtendPVC* view when they are created. This is because the option Add PVC to SubtendPVC view is turned off to improve system performance. Although it is not recommended, you can turn on the option if you want PVCs to be added to *SubtendPVC* when they are created. See the following sections for procedures on turning this option on and off.

Turning On the Option Add PVC to SubtendPVC View

This procedure describes how to turn on the option Add PVC to SubtendPVC view, so that PVCs are added to the *SubtendPVC* view when they are created.

Note The recommended and default setting for the option Add PVC to SubtendPVC view is off. This keeps PVC objects from being added to *SubtendPVC*, which improves system performance. You can run the `make_visible` script to display hidden connections.

Step 1 Log in as super user on the CNOM system.

Step 2 Open the file `/opt/cemf/config/CDMDmmCtlr/CDMDmmCtlrUserData.ini` in a text editor such as vi or emacs.

Step 3 In the `[NI2ConnectHelper]` section of the file, add the following line:

```
PVCConnect.AddToSubtendView=yes
```

Step 4 Save the file and exit the text editor.

Step 5 Stop and restart the CDMDmmCtlr process in CNOM:

```
./sysmgrClient -k CDMDmmCtlr
./sysmgrClient -x CDMDmmCtlr
```

From now on, PVCs created by CNOM are added to the *SubtendPVC* view.

Note Although it is not recommended, you can turn on the Add PVC to SubtendPVC view option for a particular PVC by providing the following attribute when you invoke the `ConnectPVC` primitive to create that PVC:

```
CDMDmmCtlr:DDMWorkOrder:addToSubtendPVC
```

Turning Off the Option Add PVC to SubtendPVC View

The default setting for the option Add PVC to SubtendPVC view is off, which keeps PVC objects from being added to *SubtendPVC* to improve system performance. If the option has been turned on, you can turn it off again as follows:

Step 1 Open the file `/opt/cemf/config/CDMDmmCtlr/CDMDmmCtlrUserData.ini` in a text editor like vi or emacs.

Step 2 In the `[NI2ConnectHelper]` section of the file, comment out the option as shown here:

```
# PVCConnect.AddToSubtendView=yes
```

Step 3 Save the file and exit the text editor.

Step 4 Stop and restart the CDMDmmCtlr process in CNOM:

```
./sysmgrClient -k CDMDmmCtlr
./sysmgrClient -x CDMDmmCtlr
```

From now on, PVCs created by CNOM are not added to the *SubtendPVC* view.

Disconnecting from DSLAM-only Architecture

To remove a connection in a DSLAM-only network architecture, use the CNOM policy in Table 4-3. You can use the `disconnect_cdm` script to invoke the policy. Specify the connection identifier of the connection you want to disconnect when you invoke the policy.

Table 4-3 DSLAM-only Disconnection Policy

Policy	Description
`disconnectSubscriber_cdm`	Disconnects a PVC

CDM Data Recovery

When CEMF is restored from backup, the CEMF database does not have information about any DSLAM or Cisco 6400 UAC connections created since the last backup. CNOM's data recovery feature logs all network orders that create or delete VC connections, and plays back the work orders to re-create or remove connections after the CEMF database is restored. This ensures that the information in the CEMF database accurately reflects the actual configuration on the DSLAM or UAC. For information about running data recovery, see the "Performing Data Recovery to Restore CNOM from Backup" section on page 3-6.

CNOM runs the following CDM policies to re-create the DSLAM configuration after a CEMF database restore.

- `restoreCDMConnection`—Restores the DSLAM configuration in the CEMF database and on CNOM after a system failure. The policy does not change the configuration on the DSLAM.
- `clearCDMConnection`—Removes non-existent CDM VCs from the CEMF database.

CHAPTER 5

Cisco 6400 UAC Architecture

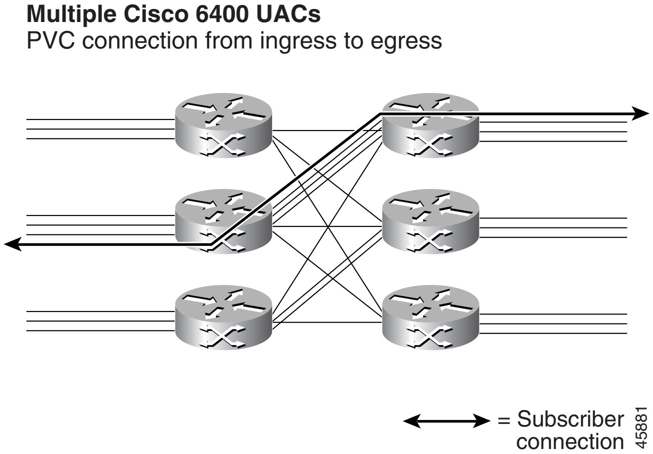
This chapter describes the connection attributes and processes of a Cisco Network Order Manager architecture consisting of a series of Cisco 6400 Universal Access Concentrators (UACs).

Configuring Cisco 6400 UACs

This section details the use of CNOM for managing PVCs in a network containing Cisco 6400 UAC devices. CNOM enables you to manage PVCs in a single UAC or between UACs. CNOM uses the ATM subscriber and service management features of the Cisco 6400 Service Connection Manager (SCM) to interact with the 6400 UACs. The SCM Network Order Adapter (NOA) specifies the ingress and egress PVCs used, and also the name of the connection template applied to the PVCs.

In addition to supporting stand-alone UACs, the Cisco 6400 UAC architecture supports clusters of Cisco 6400 UACs arranged in two layers—Remote Terminal (RT) and Competitive Local Exchange Carrier (CLEC)—with full interconnection between the layers. In a cluster topology, each cluster acts as a virtual switch, and CNOM must make PVC connections across the virtual switches. To do this, CNOM creates a pair of PVCs, one on each device, which makes a connection between a port on one Cisco 6400 UAC in a cluster to a port on another Cisco 6400 UAC in a cluster. See Figure 5-1.

Figure 5-1 Cisco 6400 UAC to Cisco 6400 UAC Configuration



CNOM uses the following types of connection objects to manage PVCs and subscriber services:

- *ATM cross-connections* (set up through the `connectSubscriberToATMService` policy) connect an ingress port to an egress port.
  - In a single UAC, the policy sets up the cross-connection within the device.
  - In an environment with multiple UACs, the policy creates a pair of PVCs, one on each device, to set up the ATM cross-connection between the RT UAC and the CLEC UAC chassis.
- *SCM service connections* (set up through the `connectSubscriberSCM` policy) connect a subscriber to a service on an NRP in the UAC.

  SCM service connections can exist only within a single UAC. You cannot set up a service connection from an ingress port on one UAC to an NRP on another.

To run CNOM SCM policies successfully, meet the following conditions:

- The Cisco 6400 UACs are deployed and commissioned.
- The ingress and egress ports are identified and configured with line IDs. See the "Configuring Line IDs for CNOM Modules" section on page A-3 for more details.
- For connections between UACs, the inter-machine tie ports are configured using the policy `set-IMT`.
- The services are configured and commissioned using the appropriate `createserviceprofile` and `createservicefromprofile` policies.

Prerequisites

Before you can use CNOM to configure Cisco 6400 UACs, you must install the following software:

- CEMF Release 3.0.4 and appropriate CEMF patches (see the *Release Notes for Cisco Network Order Manager Release 1.3*)
- Cisco 6400 Service Connection Manager (SCM)
- CNOM software

Refer to the installation guides for each product for installation procedures. The CNOM installation and upgrade procedures are described in Chapter 2, "Installing Cisco Network Order Manager."

Configuring the System

This section describes how to set up CNOM to manage connections in a network with multiple Cisco 6400 UACs. Connection management is possible only after you configure the following:

- Connection templates in SCM for the cross-connects
- Mapping objects for ingress and egress ports (only if CEMF names will not be supplied in the network orders)
- Inter-machine tie port information

The procedures for performing these configuration tasks are detailed in the following sections.

Connection Templates

An SCM connection template is a type of CEMF profile that contains ATM QoS settings to apply to a connection. By applying the profile (that is, by specifying the profile name in the connection command), you can apply those ATM QoS parameter settings to the connection rather than having to specify each parameter value separately. CNOM requires that at least one connection template is configured in SCM to facilitate connections in a network of multiple Cisco 6400 UACs.

Manage the connection templates using the SCM GUI (refer the *Cisco 6400 Service Connection Manager User Guide*), or using the CNOM network order interface.

To manage:

- SCM connection templates through the CNOM network order interface, use the policies `createConnectionTemplate`, `modifyConnectionTemplate`, and `deleteConnectionTemplate`.
- Utility scripts `create_qos`, `modify_qos`, and `delete_qos` to enable connection templates, use the command line on the CEMF server.

Connect and Disconnect Scripts

The following connect* and disconnect* perl scripts provide a wrapper around **socketDaSet** to provide a command line option to invoke the policies. You run these scripts from a CEMF shell.

- The `connect_atm` script invokes the policy `connectSubscriberToATMService`, which creates an ATM cross-connection between an ingress and egress port.
- The `disconnect_atm` script invokes the policy `disconnectSubscriberAndService`, which removes an existing ATM connection.
- The `connect_scm` script invokes the policy `connectSubscriberSCM`, which creates an SCM service connection used to connect an ingress port (or subscriber) to a service.
- The `disconnect_scm` script invokes the policy `disconnectSubscriberSCM`, which disconnects an ingress port (or subscriber) from a service.

Invoking these scripts without arguments displays the usage statement for the scripts.

These scripts generate a connection identifier using a config file `/opt/cemf/db/conncount` by default. The connection identifier is generated as string `CONN-<`*`conncount`*`>` where `<`*`conncount`*`>` is incrementally generated.

Note When you run a connect or disconnect script with the `-syslog` option, the system only records part of the work order in the syslog file. This is because the buffer on the CNOM system is too small to store the entire work order in the syslog record.

Connection Status

If a connection is:

- **Successful**—The **socketDaSet** command returns a success message and creates connection objects in the *connectionObjects* view to represent the connection. If you include the `-hide` option when you create the connection, connection objects are hidden in *connectionObjects*.

  At the root level of *connectionObjects*, an object with the same name as the `workOrder:DDMWorkOrder.connectionIdentifier` value appears. Expand this object to show the objects `C6400DMMServices` and `C6400DMMConnection`.

  - `C6400DMMServices`—Displays the ATM services created by the policy. Use SCM to access the windows to view and configure the services.
  - `C6400DMMConnection`—Displays the subscriber and the connections that were set up for this connection. The subscriber and connection objects are SCM objects and have subscriber and connection windows accessible from this view.

- **Unsuccessful**—The **socketDaSet** command returns the message "Failed to set the following." Refer to log files *ddmGateway.audit*, *ddmGateway.log*, and *C6400DMMCtrller.log* for more information.

To disconnect a connection made through the `connectSubscriberToATMService` policy, use the policy `disconnectSubscriberAndService`. The policy is assigned the connection identifier supplied for the connection in the attribute `workOrder:DDMWorkOrder.connectionIdentifier` and removes it.

Creating Mappings for ATM Interfaces

Network orders for creating connections in a multiple Cisco 6400 UAC network architecture supply names for the ingress and egress ATM interfaces. Use mapping objects to define the aliases available for use in network orders. If the names in the network order are not CEMF names, you must resolve each supplied name to be a CEMF name. To resolve supplied names to CEMF names, use mapping objects.

The mapping objects for a multiple Cisco 6400 UAC network architecture are called line IDs. Manage line IDs using the following scripts:

- `./set_line_ids <`*`ContainmentPath`*`> <`*`LineId,LineId,LineId`*`> <`*`SubdomainId`*`>`—Creates a list of line ID mapping objects, and sets the line ID attribute to be the comma-separated list of mappings. Previous mapping objects for the interface are removed.
- `./add_line_id <`*`ContainmentPath`*`> <`*`LineId`*`> <`*`SubdomainId`*`>`—Adds a line ID alias to an interface.
- `./remove_line_id <`*`ContainmentPath`*`> <`*`LineId`*`>`—Removes a single line ID alias from an interface. This step must be completed before you can remove a chassis.
- `./cleanup_line_id <`*`LineId`*`>`—Removes line IDs that were not deleted before you removed a chassis or uninstalled CNOM.

Note Use the `cleanup_line_Id` script only to remove line IDs that were not deleted before a chassis was removed or CNOM was uninstalled. For instance, if you attempt to use or remove a line ID from a removed chassis and receive an error message that you can not use that particular line ID, run the command **`./cleanup_line_Id [<`*`LineId`*`>] [<`*`ContainmentPath`*`>]`**.

- `./reset_line_ids <`*`ContainmentPath`*`>`—Removes line ID mapping objects for an interface, and clears the line ID attribute.

- `./get_line_id <ContainmentPath>`—Returns the value of the line ID attribute which may consist of a comma-separated list of line ID aliases.
- `./get_object_for_line <LineId>`—Returns the name of the interface with the specified line ID alias.

The values of the line ID aliases may contain any alphanumeric characters plus (-), (.), and (_).

Managing Inter-Machine Tie Links

In CNOM, the connections between two Cisco 6400 UAC devices are called inter-machine tie (IMT) links. You must set up IMT links in order for CNOM to be able to create ATM connections between two Cisco 6400 UACs. Each IMT link represents a physical connection between the UACs. When you create an ATM connection in CNOM, you can specify which IMT link the connection is to use or you can allow CNOM to decide.

Consider the following as you set up and manage IMT links:

- You must configure an IMT link for each physical connection between Cisco 6400 UACs that you want to make use of. For example, if there are three physical connections between a pair of UACs and you create two IMT links, CNOM will only be able to make use of the two physical connections represented by the IMT links.
- You can create multiple IMT links for a single physical connection, although this is not necessary.
- When you use the `connect_atm` script to create an ATM connection, you can specify which IMT link the connection is to use or you can let CNOM select the link (see the "Creating ATM Connections" section on page 5-7).
- You can rename an IMT link.

Creating and Removing IMT Links

CNOM uses IMT links to establish ATM connections between two Cisco 6400 UACs. To create an IMT link between two UACs, enter the following command:

```
./set-IMT <RT-Port-Containment-Path> <CLEC-Port-Containment-Path> <link_name>
```

Where:

- *<RT-Port-Containment-Path>* is the containment path of the tie link's ingress port.
- *<CLEC-Port-Containment-Path>* is the containment path of the tie link's egress port.
- *<link_name>* is a unique identifier to assign to the IMT link. When you create an ATM connection, you can assign the connection to an IMT by specifying its link name.

To remove an IMT link, enter the following command (where *<link_name>* is the IMT link to remove):

```
./unset-IMT <link_name>
```

Setting the Number of Connections

This script allows you to set the number of connections on a particular IMT link. Enter the following command:

```
./set-IMTcount <linkname> Counter
```

Where:

- *<linkname>* is the name of the IMT connection counter to set the connections on.
- Counter is the number of connections made on the IMT link.

Renaming IMT Links

This script allows you to create a new IMT name with a different name. Enter the following command:

```
./rename-IMT <old_name> <new_name>
```

Where:

- *<old_name>* is the original name for the IMT link.
- *<new_name>* is the new IMT link name. In addition to renaming the IMT link, this action resets the connection counter on this new IMT link and modifies all related connection objects to make certain that the "getConnection Data" script shows the correct new IMT name.

If all operations are successful, then this script deletes the old IMT link name.

Setting the Active IMT Link

When multiple IMT links exist between two Cisco 6400 UACs, you must designate one of the links as the active IMT link. When you create an ATM connection and do not assign the connection to a particular IMT link, CNOM assigns the connection to the active IMT link.

To set the active IMT link between a pair of UACs, use the following command (where *<link_name>* is the name of the IMT link to configure as the active link):

```
./setactive-IMT <link_name>
```

Getting Information about IMT Links

To retrieve information about IMT links, use the following command:

```
./getinfo-IMT [-link <link_name> | -chassis <chassis1> [<chassis2>] ]
```

Where:

- -link *<link_name>* retrieves information about the IMT link specified by *<link_name>*.
- -chassis *<chassis1>* retrieves information about all IMT links on the chassis specified by *<chassis1>*. You can use this option alone or with the *<chassis2>* option.
- *<chassis2>* is an optional second chassis. The system retrieves information about all IMT links that exist between both chassis.

Managing ATM Connections

This section describes how to create, delete, and query PVC connections across a pair of Cisco 6400 UACs.

Creating ATM Connections

Use the network order policy `connectSubscriberToATMService` to create a pair of PVCs that form an end-to-end ATM connection across two connected Cisco 6400 UACs. Using PVC information supplied in the network order, the policy locates the ingress and egress interfaces in CEMF, finds their inter-machine tie (IMT) link, and creates an end-to-end connection from the ingress interface to the egress interface through the IMT link. SCM autoallocates the PVCs used for tie links. If the ingress and egress interfaces are on the same UAC, CNOM makes a single cross-connection within that device.

The policy `connectSubscriberToATMService` requires that the network order contain the attributes listed in Table 5-1.

Table 5-1 Network Order Attributes for Network Order Policy connectSubscriberToATMService

Attribute	Description
`workOrder:DDMWorkOrder.connectionIdentifier`	A unique identifier for the connection.
`workOrder:DDMWorkOrder.ingressIdentifier`	A line ID value that resolves to an interface in the ingress Cisco 6400 UAC.
`workOrder:DDMWorkOrder.egressIdentifier`	A line ID value that resolves to an interface on the egress Cisco 6400 UAC.
`workOrder:DDMWorkOrder.serviceCharacteristics`	The name of an SCM connection template.
`workOrder:DDMWorkOrder.ingressVPI`	The VPI value to be used on the ingress interface.
`workOrder:DDMWorkOrder.ingressVCI`	The VCI value to be used on the ingress interface.
`workOrder:DDMWorkOrder.egressVPI`	The VPI value to be used on the egress interface.
`workOrder:DDMWorkOrder.egressVCI`	The VCI value to be used on the egress interface.
`workOrder:DDMWorkOrder.linkAction`	Optional IMT primitive action. Tells CNOM how to choose the TieLink.
`workOrder:DDMWorkOrder.linkName`	Optional TieLink name attribute.
`workOrder:DDMWorkOrder.saveForRestore`	Optional attribute. Tells the system not to save information for Data Recovery.
`workOrder:DDMWorkOrder.saveInSysLog`	Optional attribute. Save Data Recovery information to system log file.
`workOrder:DDMWorkOrder.saveInLogFile`	Optional attribute. Save Data Recovery information to specified log file.
`workOrder:DDMWorkOrder.logFileName`	Optional attribute. Defines log file name.
`workOrder:DDMWorkOrder.MakeInvisible`	Optional Attribute. Hide connection object.
`workOrder:DDM.finalWorkOrder`	Optional attribute which when set, causes the contents of the internal work order to be returned to the user. To view error messages, this attribute must be set.

You can run the following script `connect_atm` to invoke the connectSubscriberToATMService policy:

```
./connect_atm <ingressID> <egressID> <serviceChar> <ingressVPI> <ingressVCI> <egressVPI>
<egressVCI> [ConnId] [-link <TieLink_Name> | -linkAuto] [-noSave | [syslog] -save
<file_path_name>] [-hide]
```

Table 5-2 shows the options available for `connect_atm` script and the corresponding Network Order attributes.

ls

Table 5-2 Script Options, Corresponding Network Order Attributes and Their Actions

Script Option	Network Order Attributes Set by Option	Attribute Value	Action of Script Option	Action if Option is Not Set
`-noSave`	`workOrder:DDMWorkOrder.saveForRestore`	"NO"	System does not save information for data recovery.	Data recovery information is saved as it was configured in DDMGencUserData.ini file or as defined by `-save -syslog` options, if any.
`-save <file_name>`	`workOrder:DDMWorkOrder.saveInLogFile`	"1"	System saves data recovery information to the specified file.	Data recovery information is saved as configured in DDMGencUserData.ini file.
	`workOrder:DDMWorkOrder.LogFileName`	log file name		
`-syslog`	`workOrder:DDMWorkOrder.saveInSysLog`	"1"	System saves data recovery information to the system log file.	Data recovery information is saved as configured in DDMGencUserData.ini file.
`-hide`	`workOrder:DDMWorkOrder.MakeInvisible`	"YES"	System hides the connection object created by the connect script to improve system performance.	Connection object is shown in the ConnectionObjects view. Large numbers of connection objects in this ConnectionObjects view can reduce system perfomance.
`-link <Tie link name>`	`workOrder:DDMWorkOrder.linkAction`	"GetLinkInfoByName"	The operator configures CNOM to use a specific Tie Link between two Cisco 6400 chassis.	If you do not specify which IMT link should be used to create an ATM connection by using the -link, -linkAuto, or -linkActive option, then CNOM chooses the IMT link that is configured in the ini file. See the following note for details.
	`workOrder:DDMWorkOrder.linkName`	Tie Link Name		
`-linkAuto`	`workOrder:DDMWorkOrder.linkAction`	"GetAutoLinkByChassis"	CNOM automatically chooses the IMT link between RT and CLEC chassis with the least number of connections currently assigned.	
`-linkActive`	`workOrder:DDMWorkOrder.linkActive`	"GetActiveLinkByChassis"	CNOM assigns the IMT link that is currently configured to *active* state.	

Note If you do not specify which IMT link should be used to create ATM connection, then CNOM will look into <CEMF>/config/C6400DMMCtrller/C6400DMMCtrllerUserData.ini file to determine the behavior.
For example, this file may contain:

```
[LoadBalancing]
IMTaction=GetActiveLinkByChassis
```

Possible values for IMTaction are GetActiveLinkByChassis, GetAutoLinkByChassis, where:
- GetActiveLinkByChassis option will make CNOM to choose active IMT when creating connection
- GetAutoLinkByChassis option will make CNOM to choose IMT with the least number of connections on it (use LoadBalancing).

If the ini file is not configured, then the LoadBalancing feature will be used (and CNOM will choose the IMT link with the least number of connections on it). The ini file is not configured by default.

The default behavior of Data Recovery subsystem is defined in <CEMF>/config/DDMGenc/DDMGencUserData.ini file.

For example, if you use the DDMGencUserData.ini file, then Data Recovery information is saved in /opt/cemf/log/connectionsLog.log file only. Use the following script:

```
[ConfigSave]
ConfigSave.saveInFile=1
ConfigSave.logFileName=/opt/cemf/log/connectionsLog.log
ConfigSave.saveInSyslog=0
```

Script options {-save, -noSave, -syslog} (and corresponding Network Order attributes) override defaults defined in the DDMGencUserData.ini file.

If you do not use any options with the connect_atm script, then:

- Connections are created using the currently active Tie Link
- Connection objects are shown in the connectionObject view
- Data recovery information is saved in the /opt/cemf/log/connectionsLog.log file.

Note For more information about Common script options (-noSave, -save<file_name>, -syslog, -hide), see Common Arguments for Scripts section on page 3-3.

CNOM Internal Work Flow for connectSubscriberToATMServiceConnection

The following list summarizes the steps that CNOM performs to execute the `connectSubscriberToATMService` policy:

Initial State: `ddmGateway:DDMGenc:C6400DMM.findCLECInterface_CSAS`

1. Finds Interface Object: `FindMappedObject`
2. Finds Chassis Object: `GetAncestorPrimitive`
3. Gets RT Object: `FindMappedObject`
4. Finds RT Chassis: `GetAncestorPrimitive`
5. Gets CLEC Interface name: `GetContainmentPath`
6. Gets RT Interface name: `GetContainmentPath`
7. Creates Connection Object: `ObjectDeployPrimitive`
8. Creates Service Object: `ObjectDeployPrimitive`
9. Copies values to egress point: `Annotate`
    a. Find RT Egress and CLEC Ingress Interfaces: `FindTieLink`
    b. Copy values to RT Ingress: `Annotate`
10. Sets attributes of ATM Service: `SetupATMService`
11. Connects Subscriber: `ConnectSubscriber`
12. Copies RT values to Egress: `Annotate`
13. Copies RT values to Ingress: `Annotate`
14. Sets attributes of ATM Service: `SetupATMService`
15. Connects Subscriber: `ConnectSubscriber`

Querying Connections

Use the network order policy `getATMConnectionData` to gather information about existing connections. The network order policy `getATMConnectionData` requires the following attributes (Table 5-3).

Table 5-3 Attributes for Network Order Policy getATMConnectionData

Attribute	Description
`workOrder:DDMWorkOrder.connectionIdentifier`	A unique identifier for the connection.
`workOrder:DDM.finalWorkOrder`	Must be set in order for the internal work order to be returned to user. The internal work order contains the results of the query.

The `getATMConnectionData` policy returns the values of the attributes set on a connection object during execution of the network order policy `connectSubscriberToATMService`.

Deleting Connections

Use the network order policy `disconnectSubscriberAndService` to delete the PVCs associated with an end-to-end connection (Table 5-4). You can use the script `disconnect_atm` to invoke the policy.

Table 5-4 Attributes for Network Order Policy disconnectSubscriberAndService

Attribute	Description
`workOrder:DDMWorkOrder.connectionIdentifier`	The unique identifier assigned to the connection.
`workOrder:DDM.finalWorkOrder`	Optional attribute, which, when set, causes the contents of internal work order to be returned to user.

Managing Service Connections

This section provides information about managing service connections, which are used to connect a subscriber to a service on the Cisco 6400 UAC. CNOM provides policies for managing the following types of services:

- PPP over ATM (PPPoA) service
- L2TP service
- Bridging service
- IRB service
- RFC 1483 service

To manage services on a Cisco 6400 UAC, you must perform the following steps in CNOM, each of which is described in the sections that follow.

1. Create a service profile.
2. Create a service from a service profile.
3. Connect a subscriber to a service.

Creating Service Profiles

CNOM provides several policies for creating different types of service profiles. Each service profile describes the characteristics for a particular type of service, such as PPPoA or L2TP. After you create a service profile, you can run one of the `createservicefromprofile` policies to create a service from the settings in the profile.

CNOM provides the following scripts and policies for creating service profiles:

- The `create_B_profile` script invokes the `createBridgingServiceProfile` policy to create a bridging service profile.
- The `create_IRB_profile` script invokes the `createIRBServiceProfile` policy to create an IRB service profile.
- The `create_L2TP_profile` script invokes the `createL2TPServiceProfile` policy to create an L2TP service profile.
- The `create_PPPoA_profile` invokes the `createPPPoAServiceProfile` policy to create a PPP over ATM service profile.
- The `create_R_profile` script invokes the `createRFC1483ServiceProfile` policy to create an RFC 1483 service profile.

Following is a summary of the steps performed by each `createserviceprofile` policy:

1. Finds the containment path for the profile.

2. Creates a profile object.
3. Sets the profile attributes based on command-line or policy parameters.

For more information about these policies, and the corresponding policies for modifying and deleting service profiles, see Appendix B, “Cisco Network Order Manager Policies.”

Creating Services

CNOM provides several policies for creating a service from a service profile. Each policy for creating a service profile has a corresponding policy for creating a service from that profile. When you invoke a `createservicefromprofile` script or policy, you specify the name of the service profile to use to create the service. For example, you would use the policy `createL2TPServiceProfile` to create an L2TP service profile, and then use the corresponding policy `createL2TPServiceFromProfile` to create a service from that L2TP service profile.

CNOM provides the following scripts and policies for creating services from profiles:

- The `create_B_service` script invokes the `createBridgingServiceFromProfile` policy to create a bridging service from the specified profile.
- The `create_IRB_service` script invokes the `createIRBServiceFromProfile` policy to create an IRB service from the specified profile.
- The `create_L2TP_service` script invokes the `createL2TPServiceFromProfile` policy to create an L2TP service from the specified profile.
- The `create_PPPoA_service` invokes the `createPPPoAServiceFromProfile` policy to create a PPP over ATM service from the specified profile.
- The `create_R_service` script invokes the `createRFC1483ServiceFromProfile` policy to create an RFC 1483 service from the specified profile.

Following is a summary of the steps performed by each `createservicefromprofile` policy:

1. Finds the NRP and chassis to create the service on.
2. Finds the service profile to use to create the service.
3. Creates a service object.
4. Applies the service profile settings to the service object.
5. Commissions the service object.

For more information about these policies, and the corresponding policies for modifying and deleting services, see Appendix B, “Cisco Network Order Manager Policies.”

Connecting Subscribers to Services

The final step in managing SCM service connections is to connect a subscriber to a service. The policy `connectSubscriberSCM` sets up a connection between an ingress port and a service in a Cisco 6400 UAC. To run the policy successfully, meet the following conditions:

- Cisco 6400 UAC is deployed and commissioned.
- Service instances are deployed and commissioned.
- Service name is configured in `LocalDB:C6400DMM.serviceName` for the service.
- Connection Template objects are deployed and configured.
- Line IDs are configured for Cisco 6400 UAC ingress ports.

See the "Configuring Line IDs for CNOM Modules" section on page A-3 for more details.

Following is a summary of the steps performed by the `connectSubscriberSCM` policy:

1. Identifies the ingress port using the line ID or CEMF object name in the incoming work order.
2. Deploys the subscriber object and configures the subscriber values.
3. Allocates a valid VPI/VCI value if one was not provided by the work order.
4. Invokes a connect action on the subscriber object to perform the connection.

The following optional attributes are supported in work orders for the connection. The attributes are dependent on the type of services to which the connection is made:

- `workOrder:DDMWorkOrder.rfc1483SubIfIPAddress`
- `workOrder:DDMWorkOrder.rfc1483SubIfSubnetMask`
- `workOrder:DDMWorkOrder.vcIPAddress`
- `workOrder:DDMWorkOrder.rfc1483EncapsType`
- `workOrder:DDMWorkOrder.cpeSupportInArp`
- `workOrder:DDMWorkOrder.ptamdConnectionType`
- `workOrder:DDMWorkOrder.ptamdEncapsType`
- `workOrder:DDMWorkOrder.rbeSubIfIPAddress`
- `workOrder:DDMWorkOrder.rbeSubIfSubnetMask`

To disconnect a subscriber from a service, use the `disconnectSubscriberSCM` policy. You can invoke the policy through the `disconnect_scm` script.

CHAPTER 6

Configuring a WAN Environment

This chapter provides information about how the Cisco Network Order Manager can be used in a WAN environment. It contains the following sections:

- CNOM Connection Within a WAN Cloud, page 6-1
- CNOM QoS Profiles for WAN Configuration, page 6-2
- Connection Status, page 6-3

CNOM Connection Within a WAN Cloud

Use the CNOM policy `connectSubscriber_cwm` to set up a connection between ingress and egress ports on the appropriate nodes in a WAN cloud. To run the policy, satisfy the following conditions:

- CWM is installed and configured (on a separate server).
- ATM cloud must be deployed.
- WAN QoS profiles must be created before you attempt to configure connections in the WAN cloud.
- VPI/VCI ingress and egress values must be available.

The policy expects the following attributes in the work order:

- `workorder:DDMWorkOrderconnectionIdentifier`
- `workorder:DDMWorkOrder.ingressATMEndPointNodeName`
- `workorder:DDMWorkOrder.ingressATMEndPointIfShelf`
- `workorder:DDMWorkOrder.ingressATMEndPointSlot`
- `workorder:DDMWorkOrder.ingressATMEndPointPort`
- `workorder:DDMWorkOrder.ingressATMEndPointVPI`
- `workorder:DDMWorkOrder.ingressATMEndPointVCI`
- `workorder:DDMWorkOrder.egressATMEndPointNodeName`
- `workorder:DDMWorkOrder.egressATMEndPointIfShelf`
- `workorder:DDMWorkOrder.egressATMEndPointSlot`
- `workorder:DDMWorkOrder.egressATMEndPointPort`
- `workorder:DDMWorkOrder.egressATMEndPointVPI`
  `workorder:DDMWorkOrder.egressATMEndPointVCI`
- `workorder:DDMWorkOrder.atmServiceCharacteristics`

To set up a connection within a WAN Cloud, use the `connect_cwm` script:

```
./connect_cwm<Connection_ID><QoS_type><ingressNode><ingressSlot><ingressPort><ingressVPI>
<ingressVCI><egressNode><egressSlot><egressPort>[<egressVPI><egressVCI>]
```

CNOM QoS Profiles for WAN Configuration

You can create a WAN QoS profile in one of two ways:

- Graphical User Interface (GUI)
- CNOM policy

To create a WAN QoS profile with a policy, use the `create_cwm_QoS` command:

./create_cwm_qos <*ProfileName*> <*ProfileType*> [<***attribute=value***>]

Supported ProfileTypes are listed in Table 6-1.

Table 6-1 WAN QoS Profile Types

Profile Type	Description
cbr1	Constant bit rate (cell loss priority [CLP] not applicable)
vbr1	Variable bit rate (CLP not applicable)
vbr2	Variable bit rate
vbr3	Variable bit rate (CLP enabled)
abr-fs	Available bit rate (CLP enabled; ForeSight enabled)
ubr-1	Unspecified bit rate (CLP disabled)
ubr-2	Unspecified bit rate (CLP enabled)
abr-1	Available bit rate

The policy expects the following attributes in the work order.

Note If you do not assign a value to an attribute, the default value is used.

- `workOrder:DDMWorkOrder.connectionIdentifier`
- `workOrder:DDMWorkOrder.ldbsvConnSubType`
- `workOrder:DDMWorkOrder.profileName`
- `workOrder:DDMWorkOrder.CtrlState`
- `workOrder:DDMWorkOrder.ldbatmEndPointBCM`
- `workOrder:DDMWorkOrder.ldbatmEndPointCDVTZeroPlus1`
- `workOrder:DDMWorkOrder.ldbatmEndPointFGCRA`
- `workOrder:DDMWorkOrder.ldbatmEndPointFRTT`
- `workOrder:DDMWorkOrder.ldbatmEndPointICR`
- `workOrder:DDMWorkOrder.ldbatmEndPointICRTO`
- `workOrder:DDMWorkOrder.ldbatmEndPointMBS`
- `workOrder:DDMWorkOrder.ldbatmEndPointMCR`
- `workOrder:DDMWorkOrder.ldbatmEndPointMinAdjustPeriod`
- `workOrder:DDMWorkOrder.ldbatmEndPointNRM`
- `workOrder:DDMWorkOrder.ldbatmEndPointPCRZeroPlus1`
- `workOrder:DDMWorkOrder.ldbatmEndPointPercUtil`
- `workOrder:DDMWorkOrder.ldbatmEndPointPolicing`
- `workOrder:DDMWorkOrder.ldbatmEndPointRateDown`

- `workOrder:DDMWorkOrder.ldbatmEndPointRateUp`
- `workOrder:DDMWorkOrder.ldbatmEndPointSCRZeroPlus1`
- `workOrder:DDMWorkOrder.ldbatmEndPointTBE`
- `workOrder:DDMWorkOrder.ldbatmEndPointVSVD`
- `workOrder:DDMWorkOrder.ldbsvConnCellRouting`
- `workOrder:DDMWorkOrder.ldbsvConnClassOfService`
- `workOrder:DDMWorkOrder.ldbsvConnTrkAvoidType`
- `workOrder:DDMWorkOrder.ldbsvConnTrkAvoidZCS`
- `workOrder:DDMWorkOrder.ldbatmEndPointICR`

Connection Status

If a connection is:

- **Successful**—The **socketDaSet** command returns a success message and creates connection objects in the *connectionObjects* view to represent the connection. If you include the `-hide` option when you create the connection, connection objects are hidden in *connectionObjects*. An object with the same name as the `workOrder:DDMWorkOrder.connectionIdentifier` value exists at the root level of the *connectionObjects* view. Expanding this object displays an `ATM` object that represents the connection.
- **Unsuccessful**—The **socketDaSet** command returns the message “Failed to set the following.” Refer to log files *ddmGateway.audit*, *ddmGateway.log*, *CDMDmmCtlr.log*, and *ATMCMController.log* for more information.

To disconnect a connection made through the above policy, use the policy `disconnectSubscriber_with_CWM`. This policy takes the `connectionIdentifier` supplied in the earlier connection in the attribute `workOrder:DDMWorkOrder.connectionIdentifier` and removes the subscriber.

Caution When you run policies, be sure that the policy is one designated for the specified application. For example, the command listed above is specifically designed to disconnect an end-to-end connection in a WAN environment.

APPENDIX A

Cisco Network Order Manager System Configuration

This appendix describes how to configure CNOM. It contains several sections:

- CNOM Configuration, page A-1
- Profile Management, page A-2
- Setting Up a Subtended Hierarchy, page A-2
- Configuring Line IDs for CNOM Modules, page A-3
- Setting Quality of Service, page A-6

CNOM Configuration

To configure a CNOM system, you must complete the following:

- Installation—Install CEMF, any required patches, relevant element managers, and necessary CNOM components on applicable servers.
- Configuration—Create and configure profiles which you can later apply to connection segments (PVCs, DMT ports). Configure topology attributes to allow CNOM to navigate the network.
- Operation—Submit network orders to the *ddmGateway* process to bring the system live.

After installation, configure CNOM using script utilities supplied with the product to create mapping objects and set the various attributes in CEMF that CNOM uses during connection management. Examples of configuration tasks that are carried out are:

- Creating mapping objects to allow resolution from interface names received in network orders to CEMF object IDs. The line ID values used in the Cisco 6400 UAC architecture are examples of this.
- Configuring inter-machine tie ports in the multiple Cisco 6400 UAC architecture using the **set-IMT** command.

Configuration is normally required after any new device is installed in the network if the new device is to support connections managed by CNOM.

Profile Management

Network orders that create connections contain the attribute `serviceCharacteristics`, whose value corresponds to the names of profiles that will be applied by the various participating element managers. The settings in each profile should support the intended characteristics of the connection. So, an ADSL profile called *Gold* in CDM sets line speeds on a DSL port which match the speed settings in the *Gold* SCM connection template that is applied to a PVC cross-connect.

Each element manager must be configured to contain profiles corresponding to each possible value of `serviceCharacteristics`. For example, suppose you have the following transport speeds and DSL types:

- ADSL-1.5x384—ADSL transport with upstream/downstream speeds of 1.5 Mbps and 384 kbps
- ADSL-512x128—ADSL transport with upstream/downstream speeds of 512 kbps and 128 kbps
- SDSL-512—SDSL transport with speed 512 kbps
- SDSL-256—SDSL transport with speed 256 kbps

In this case you might set up DMT, ADSL, CAP and ATM QoS profiles named `ADSL-1.5x384` and `ADSL-512x128`, and SDSL and ATM QoS profiles named `SDSL-512` and `SDSL-256`. In the network order, you would also specify the name of the appropriate profile for the `serviceCharacteristics` attribute.

The primitive *ConfigureDSLPort*, used in the policy `connectSubscriber` applies the profiles of the appropriate type for the port being configured.

Setting Up a Subtended Hierarchy

Note This procedure is necessary only if you are running CDM Release 3.0.

The subtend hierarchy in CDM Release 3.0 is modeled in CEMF as a containment view called *Subtend*. All DSLAMs which are to be used by CNOM must appear in this view, even if they operate as standalone units. The view contains the chassis object for each DSLAM, and each chassis object contains its trunk and subtend ports. Chassis objects appear at either:

- The root of the tree—Indicating that they are not connected to the subtend port of any other DSLAM
- Under a subtend port of another DSLAM—Indicating that the trunk port is connected to the specified subtend port

The policy `setupSubtend` automatically sets up the DSLAM in a *Subtend* view and also sets the attributes to designate the network port and subtend ports. You can invoke the `setupSubtend` policy by running the **subtend** command as follows:

```
./subtend node_chassis_path [subtend_port_path]
```

Where:

- `node_chassis_path` is the containment path of the DSLAM node. If the DSLAM is not subtended, omit the following parameter.
- `subtend_port_path` is the port to connect the subtend node to. The DSLAM to connect the node to must already exist in the *Subtend* view. Omit this parameter if the node is not subtended.

Note When you upgrade from CNOM Release 1.03 or 1.05 or CDM Release 3.1 or later, you must remove the Subtend package before you begin the upgrade.

To position a node under a subtend path, the subtend port must already exist in the *Subtend* view. If the [subtend_port_path] parameter is not provided, the node is placed in the top level of the subtend view. For example, to set up a subtend view of three nodes (Node1/Node2/Node3), each with a single SONET trunk port and single SONET subtend port, run the following commands:

```
./subtend Physical:/Site-1/Node1-Shelf/Node1-Chassis
./subtend Physical:/Site-1/Node2-Shelf/Node2-Chassis subtend:/Node1-Chassis/SONET-10-2
./subtend Physical:/Site-1/Node3-Shelf/Node3-Chassis subtend:/Node2-Chassis/SONET-10-2
```

Note Run the **subtend** command from a CEMF shell.

Configuring Line IDs for CNOM Modules

Line IDs are used in CNOM to identify the ingress point (start) or egress point (end) of a connection. For example, line IDs are used to identify the ingress port for a DSLAM or a Cisco 6400 UAC connection. In the SCM ATM Connection policy, the line IDs are used to identify both an ingress and egress value. The line IDs are:

- Any string value associated with a port such as circuit ID, phone number, and so on.
- The identifier that is received in incoming work orders if a CEMF name is not supplied.

For example:

- The line ID for a DSLAM line port is a unique identifier assigned to the subscriber served by this line port (for example, the phone number 4085551212).
- For a Cisco 6400 UAC ingress port, the line ID is the set of DSLAMs being serviced by the UAC (for example, DSLAM-1, DSLAM-2, DSLAM-3). Each comma-separated value must be unique for the port and cannot be assigned to another port.

For performance, line IDs are managed in an invisible view, *ObjectMapView*, under the technology object `DDMTopology.lineID`. For each line ID assigned to a port, the system creates a map object whose name matches the line ID. The line ID object includes the following attributes:

- `LocalDB:DDMMappingObject.targetObject`—The line port which is assigned this line ID.
- `LocalDB:DDMMappingObject.subdomainId`—The subdomain (the set of CNOM devices) that the line port belongs to: DSLAM or C6400DMM.

In addition, the line port sets the value in its local attribute (`LocalDB:DDMTopology.lineID`).

When you assign the value DSLAM-1,DSLAM-2,DSLAM-3 to a line port, the resulting configuration in CEMF is as follows:

- Three line ID objects are created under `ObjectMapView:/DDMTopology` with the names DSLAM-1, DSLAM-2, and DSLAM-3
- The `targetObject` is set as the ingress port
- The ingress port's attribute `LocalDB:DDMTopology.lineID` is assigned the line ID value DSLAM-1,DSLAM-2,DSLAM-3

Line IDs are configured using policies which provide a more abstract form of mapping objects to arbitrary attribute values. The policies include:

- `./setMappingValues`—Sets a new set of mapping values for the object
- `./resetMappingValues`—Resets existing mapping values for an object
- `./addMappingValue`—Adds a new mapping value to the object

- `./removeMappingValue`—Removes a mapping value from the object
- `./findMappingTarget`—Identifies an object with a mapping value and returns the target object

The following command line scripts provide perl wrapper scripts to execute this policy:

- `./set_map_values`
- `./reset_map_values`
- `./add_map_value`
- `./remove_map_value`
- `./get_object_for_value`

Command line scripts are provided to configure line IDs for CNOM.

Note If the same line IDs are assigned to more than one port, policies fail.

Setting the Line ID of a DSLAM Port

Use the `./set_line_ids` script to assign a line ID to a DSLAM port to identify a subscriber line attached to the port. The script assigns a new ID to the port if none exists, or it overwrites an existing line ID with a new value. The command syntax is as follows:

```
./set_line_ids <port> <line_ID> DSLAM
```

Where:

- *<port>* is the containment path of the ingress or egress port whose line ID you are setting.
- *<line_ID>* is a unique identifier for the subscriber, such as a circuit ID or telephone number. `DSLAM` indicates that the line ID is assigned to a port in the DSLAM subdomain.

The following command assigns the line ID `408-555-1212` to the DSLAM port `Chelmsford/.../DMT-5-1` (where \ indicates that the command continues on a new line):

```
./set_line_ids ComponentManaged:/Chelmsford/ni2-90-shelf/ni2-90-chassis/D5/DMT-5-1 \
4085551212 DSLAM
```

Setting the Line ID for a Cisco UAC Port

To identify the set of DSLAMs served by the UAC, set the line ID for a Cisco 6400 UAC port. The command syntax is as follows:

```
./set_line_ids <port> <line_ID> C6400DMM
```

- For *<port>*, specify the containment path of the ingress or egress port whose line ID you are setting.
- For *<line_ID>*, specify the set of DSLAMs served by the port (include commas between each DSLAM name). `C6400DMM` indicates that the line ID is assigned to a port in the Cisco 6400 UAC subdomain.

The following command sets the line ID for the UAC port `sw1-shelf/.../ATM-3-1-0`
(where \ indicates that the command continues on a new line). The line ID indicates that the port serves DSLAM-1, DSLAM-2, and DSLAM-3.

```
./set_line_ids Cisco6400ServiceView:/sw1-shelf/sw1-chassis/sw1-nsp/OC3-3-1/ATM-3-1-0 \
DSLAM-1,DSLAM-2,DSLAM-3 C6400DMM
```

Note Run all of the following commands from the CEMF shell.

- The `./get_line_id <port>` script prints the line ID of the ingress port for the specified *port*. For example, for the port `sw1-shelf/.../ATM-3-1-0`, the script returns the value `DSLAM-1,DSLAM-2,DSLAM-3`.
- The `./get_object_for_line` script takes a line ID value and prints the containment path of the object that is assigned this value. For example, **`./get_object_for_line DSLAM-1`** returns the value:

  ```
  workOrder:DDMWorkOrder.objectName
  Cisco6400ServiceView:/sw1-shelf/sw1-chassis/sw1-nsp/OC3-3-1/ATM-3-1-0
  ```

- The `reset_line_ids` script resets the line ID for a specified port. For example, to reset the line ID for the port `sw1-shelf/.../ATM-3-1-0`, issue the command as follows:

  ```
  ./reset_line_ids Cisco6400ServiceView:/sw1-shelf/sw1-chassis/sw1-nsp/OC3-3-1/ATM-3-1-0
  ```

  If you then run `./get_line_id`, the script returns an empty value (because the line ID has been reset).
- The `./add_line_id` script assigns an additional line ID to the specified port's current line ID. For example, if the line ID is DSLAM-1,DSLAM-2,DSLAM-3 and you issue the following command, the line ID becomes DSLAM-1,DSLAM-2,DSLAM-3,DSLAM-4.

  ```
  ./add_line_id Cisco6400ServiceView:/sw1-shelf/sw1-chassis/sw1-nsp/OC3-3-1/ATM-3-1-0 \
  DSLAM-4 C6400DMM
  ```

- The `./remove_line_id` script removes a line ID from the port's current value. For example, the following command removes DSLAM-2, making the port's line ID value DSLAM-1,DSLAM-3,DSLAM-4:

  ```
  ./remove_line_id Cisco6400ServiceView:/sw1-shelf/sw1-chassis/sw1-nsp/OC3-3-1/ATM-3-1-0\
  DSLAM-2 C6400DMM
  ```

  If you run **`./get_object_for_line DSLAM-2`**, the script returns an error message stating that this value is no longer mapped to an object.

Note Use the **`cleanup_line_Id`** command to remove line IDs that were not deleted before you removed a chassis or uninstalled CNOM. For instance, if you attempt to use or remove a line ID from a removed chassis and receive an error message that you can not use that particular line ID, run the **`./cleanup_line_Id`** [<***`LineId`***>] [<***`ContainmentPath`***>] command.

Setting Quality of Service

For CNOM to successfully make connections, you must name at least one fully configured quality of service (QoS) through the network. To set a quality of service, perform the following steps:

Step 1 Create a CDM Manager DMT profile using an appropriate name.

Step 2 Create a CDM Manager DSL profile using an appropriate name.

Step 3 Create a CDM Manager ATM connection profile using an appropriate name.

Step 4 Create an SCM connection template using an appropriate name.

Note An appropriate name in the above procedure is any convention that functions within the network administrator for keeping a variety of QoS packages uniquely identified. For example, Gold, Silver, and Bronze are identifiers that can designate (in a hierarchical order) various QoS packages.

Tag each connection template with a profile name to tell CNOM which named QoS it represents. This name is used in the work order to identify the appropriate connection template for a connection.

APPENDIX B

Cisco Network Order Manager Policies

This appendix provides an overview of CNOM policies, a description of each policy, and a description of the CNOM internal work flow. It also includes a table that lists the scripts provided with CNOM for invoking policies. This chapter includes the following sections:

- DSLAM Profile and Policies, page B-1
- NI-2 Profile Policies, page B-2
- Cisco 6400 UAC Profile and Service Policies, page B-7
- Cisco WAN Manager Profile Policies, page B-22

DSLAM Profile and Policies

DSLAM profile policies consist of the DSL and ATM profiles that CNOM uses to configure a connection. You can set up the profiles in the CDM GUI, or by using the policies supported by the CDM NOA. The profile policies are responsible for deploying profile objects and configuring connection profile values for:

- Line IDs
- DSL
- ATM QoS
- PVC/SPVC
- VPI/VCI

DSLAM-only Architecture Policies

The policies `connectSubscriber_CDM` and `disconnectSubscriber_CDM` support DSLAM-only profile management.

The connect policy uses the following parameters:

- `workOrder:DDMWorkOrder.connectionIdentifier`
- `workOrder:DDMWorkOrder.ingressIdentifier`
- `workOrder:DDMWorkOrder.serviceCharacteristics`
- `workOrder:DDMWorkOrder.autoAllocate`
- `workOrder:DDMWorkOrder.setupSoftPVC`
- `workOrder:DDMWorkOrder.ingressVCI`

- `workOrder:DDMWorkOrder.ingressVPI`
- `workOrder:DDMWorkOrder.egressVCI`
- `workOrder:DDMWorkOrder.egressVPI`

The disconnect policy is assigned the profile name as the parameter for deleting the DSLAM profile.

Multiple Cisco 6400 UAC Architecture Policies

The policies `connectSubscriberToATMService` and `disconnectSubscriberAndService` support profile and service management in an environment of multiple Cisco 6400 UACs.

The connect policy uses the following parameters:

- `workOrder:DDMWorkOrder.connectionIdentifier`
- `workOrder:DDMWorkOrder.ingressIdentifier`
- `workOrder:DDMWorkOrder.egressIdentifier`
- `workOrder:DDMWorkOrder.serviceCharacteristics`
- `workOrder:DDMWorkOrder.ingressVPI`
- `workOrder:DDMWorkOrder.ingressVCI`

The disconnect policy is assigned the profile name as the parameter for disconnecting from a multiple Cisco 6400 UAC architecture profile.

NI-2 Profile Policies

The profiles for configuring NI-2 line cards consist of DSL parameter profiles and ATM QoS profiles. The profiles are either set up in the CDM GUI, or through the policies supported by the CDM NOA. The profile configuration policies manage the following:

- DMT profile with ADSL profile
- SDSL profile
- CAP profile
- ATM profile

The perl script `create_ni2_profile` provides a wrapper around **socketDaSet** to create a profile. Invoke the script as follows (where <DMT, CAP, SDSL, or ATMQoS> is the type of profile to create, and <*profileName*> is the name to assign the profile).

```
/opt/cemf/bin/create_ni2_profile <DMT|CAP|SDSL|ATMQoS> <profileName>
```

To check the arguments for a DMT profile, invoke the following script:

```
./create_ni2_profile DMT
```

Modify an existing profile using the scripts:

- `modify_ni2_profile`
- `delete_ni2_profile`

The delete script contains two parameters: profile type (DMT, CAP, SDSL, or ATMQoS) and profile name as follows:

- **`/opt/cemf/bin/delete_ni2_profile <DMT|CAP|SDSL|ATMQoS>`** ***`<profileName>`***

NI-2 DMT Profile

The policies `createNI2DMTProfile`, `modifyNI2DMTProfile`, and `deleteNI2DMTProfile` support DMT profile management. The internal work flow sections describe how CNOM executes the policies.

Note From the GUI, configure the DMT profile and the ADSL profile separately. From the DMT profile policy, configure the ADSL and DMT profiles together.

The create and modify policies use the following parameters:

- `workOrder:CEMCS-IF-ADSL-MIB.ChanConfRMaxInterleaveDelay`
- `workOrder:CEMCS-IF-ADSL-MIB.ChanConfCMaxInterleaveDelay`
- `workOrder:CEMCS-IF-ADSL-MIB.ChanConfRInterleaveMinTxRate`
- `workOrder:CEMCS-IF-ADSL-MIB.ChanConfCInterleaveMinTxRate`
- `workOrder:CEMCS-IF-ADSL-MIB.ChanConfRInterleaveMaxTxRate`
  `workOrder:CEMCS-IF-ADSL-MIB.ChanConfCInterleaveMaxTxRate`
- `workOrder:CEMCS-IF-ADSL-MIB.ConfRTargetSnrMgn`
- `workOrder:CEMCS-IF-ADSL-MIB.ConfCTargetSnrMgn`
- `workOrder:CEMCS-IF-DMT-MIB.dmtConfTrellis`
- `workOrder:CEMCS-IF-DMT-MIB.atucDmtConfOverheadFraming`
- `workOrder:CEMCS-IF-DMT-MIB.lineDmtConfOperatingMode`
- `workOrder:CEMCS-IF-DMT-MIB.lineDmtConfTrainingMode`
- `workOrder:CEMCS-IF-DMT-MIB.atucDmtConfBitSwapEnabled`
- `workOrder:CEMCS-IF-DMT-MIB.atucDmtConfBitSwapFrom`
- `workOrder:CEMCS-IF-DMT-MIB.atucDmtConfBitSwapTo`
- `workOrder:CEMCS-IF-DMT-MIB.aturDmtConfCodewordSize`
- `workOrder:CEMCS-IF-DMT-MIB.atucDmtConfCodewordSize`
- `workOrder:CEMCS-IF-DMT-MIB.aturDmtConfInterleaveFecSize`
- `workOrder:CEMCS-IF-DMT-MIB.atucDmtConfInterleaveFecSize`
- `workOrder:DDMWorkOrder.profileName`

The delete policy is assigned the profile name as the parameter for deleting both the ADSL and DMT profiles.

CNOM Internal Work Flow for createNI2DMTProfile

Initial State: `CDMDmmCtrlr:DSLAM.findDMTTechnologyObject_createNI2DMTProfile`

1. Find Technology Object: `ResolveName`
   a. If found, Create DMT Profile Object: `ObjectDeployPrimitive`
   b. If not found, Create Technology Object: `ObjectDeployPrimitive`
   – Create DMT Profile Object: `ObjectDeployPrimitive`
2. Create DMT Object: `ObjectDeployPrimitive`
3. Set Profile Attributes: `SetProfileAttribute`

CNOM Internal Work Flow for modifyNI2DMTProfile

Initial State: `CDMDmmCtrlr:DSLAM.findDMTTechnologyObject_modifyNI2DMTProfile`

1. Find DMTProfile by Name: `ResolveName`
2. Find DMT Object: `GetDescendantPrimitive`
3. Modify Attributes: `SetAttributesPrimitive`

CNOM Internal Work Flow for deleteNI2DMTProfile

Initial State: `CDMDmmCtrlr:DSLAM.findDMTTechnologyObject_deleteNI2DMTProfile`

1. Find DMTProfile by Name: `ResolveName`
2. Delete DMTProfile Object: `ObjectDeletePrimitive`

NI-2 CAP Profile

The policies `createNI2CAPProfile`, `modifyNI2CAPProfile`, and `deleteNI2CAPProfile` support CAP profile management. The create and modify policies are assigned the following parameters:

- `workOrder:DDMWorkOrder.profileName`
- `workOrder:CAPController-MIB.db-AtucCapConfTargetSnrMgn`
- `workOrder:CAPController-MIB.db-AtucCapConfUp17KBaudEnable`
- `workOrder:CAPController-MIB.db-AturCapConfPsdmLevel`
- `workOrder:CAPController-MIB.db-AtucCapConfMinTxRate`
- `workOrder:CAPController-MIB.db-CapConfCpeSignature`
- `workOrder:CAPController-MIB.db-AtucCapConfMaxTxRate`
- `workOrder:CAPController-MIB.db-AtucCapConfDown136KBaudEnable`
- `workOrder:CAPController-MIB.db-AtucCapConfUp68KBaudEnable`
- `workOrder:CAPController-MIB.db-AtucCapConfPsdmLevel`
- `workOrder:CAPController-MIB.db-AturCapConfMinTxRate`
- `workOrder:CAPController-MIB.db-AturCapConfTargetSnrMgn`
- `workOrder:CAPController-MIB.db-CapConfInterleaveDelay`
- `workOrder:CAPController-MIB.db-CapConfTrainingMode`
- `workOrder:CAPController-MIB.db-AturCapConfMaxTxRate`

The delete policy is assigned the profile name as the parameter for deleting the profile.

CNOM Internal Work Flow for createNI2CAPProfile

Initial State: `CDMDmmCtrlr:DSLAM.findCAPTechnologyObject_createNI2CAPProfile`

1. Find Technology Object: `ResolveName`
   a. If found, Create CAP Profile Object: `ObjectDeployPrimitive`
   b. If not found, Create Technology Object: `ObjectDeployPrimitive`
   – Create CAP Profile Object: `ObjectDeployPrimitive`
2. Create CAP Object: `ObjectDeployPrimitive`
3. Set Profile Attributes: `SetProfileAttribute`

CNOM Internal Work Flow for modifyNI2CAPProfile

Initial State: `CDMDmmCtrlr:DSLAM.findCAPProfileObject_modifyNI2CAPProfile`

1. Find CAP Profile by Name: `ResolveName`
2. Find CAP Object: `GetDescendantPrimitive`
3. Modify Attributes: `SetAttributesPrimitive`

CNOM Internal Work Flow for deleteNI2CAPProfile

Initial State: `CDMDmmCtrlr:DSLAM.findCAPProfileObject_deleteNI2CAPProfile`

1. Find CAP Profile by Name: `ResolveName`
2. Delete CAP Profile Object: `ObjectDeletePrimitive`

NI-2 SDSL Profile

The policies `createNI2SDSLProfile`, `modifyNI2SDSLProfile`, and `deleteNI2SDSLProfile` support SDSL profile management.

The create and modify policies are assigned the following parameters:

- `workOrder:DDMWorkOrder.profileName`
- `workOrder:CEMCS-SDSL-IF-MIB.sdslLineMaxRate`

The delete policy is assigned the profile name as the parameter for deleting the profile.

CNOM Internal Work Flow for createNI2SDSLProfile

Initial State: `CDMDmmCtrlr:DSLAM.findCAPTechnologyObject_createNI2SDSLProfile`

1. Find Technology Object: `ResolveName`
    a. If found, Create SDSL Profile Object: `ObjectDeployPrimitive`
    b. If not found, Create Technology Object: `ObjectDeployPrimitive`
    – Create SDSL Profile Object: `ObjectDeployPrimitive`
2. Create SDSL Object: `ObjectDeployPrimitive`
3. Set Profile Attributes: `SetProfileAttribute`

CNOM Internal Work Flow for modifyNI2SDSLProfile

Initial State: `CDMDmmCtrlr:DSLAM.findSDSLProfileObject_modifyNI2SDSLProfile`

1. Find SDSL Profile by Name: `ResolveName`
2. Find SDSL Object: `GetDescendantPrimitive`
3. Modify Attributes: `SetAttributesPrimitive`

CNOM Internal Work Flow for deleteNI2SDSLProfile

Initial State: `CDMDmmCtrlr:DSLAM.findSDSLProfileObject_deleteNI2SDSLProfile`

1. Find SDSL Profile by Name: `ResolveName`
2. Delete SDSL Profile Object: `ObjectDeletePrimitive`

NI-2 ATM QoS Profile

The policies `createNI2ATMQoSProfile`, `modifyNI2ATMQoSProfile`, and `deleteNI2ATMQoSProfile` support ATM QoS profile management. The create and modify policies take the following parameters:

- `workOrder:DDMWorkOrder.profileName`
- `workOrder:CEMCS-ATM-CONN-MIB.qoSCategoryTran`
- `workOrder:CEMCS-ATM-CONN-MIB.transCLP`
- `workOrder:CEMCS-ATM-CONN-MIB.transCDVT`
- `workOrder:CEMCS-ATM-CONN-MIB.transPCR`
- `workOrder:CEMCS-ATM-CONN-MIB.transMBS`
- `workOrder:CEMCS-ATM-CONN-MIB.transSCR`
- `workOrder:CEMCS-ATM-CONN-MIB.transMCR`
- `workOrder:CEMCS-ATM-CONN-MIB.qoSCategoryRecv`
- `workOrder:CEMCS-ATM-CONN-MIB.recvCLP`
- `workOrder:CEMCS-ATM-CONN-MIB.recvCDVT`
- `workOrder:CEMCS-ATM-CONN-MIB.recvPCR`
- `workOrder:CEMCS-ATM-CONN-MIB.recvMBS`
- `workOrder:CEMCS-ATM-CONN-MIB.recvSCR`
- `workOrder:CEMCS-ATM-CONN-MIB.recvMCR`

The delete policy is assigned the profile name as the parameter for deleting the profile.

CNOM Internal Work Flow for createNI2ATMQoSProfile

Initial State: `CDMDmmCtrlr:DSLAM.findATMQoSTechnologyObject_createNI2ATMQoSProfile`

1. Find Technology Object: `ResolveName`
   a. If found, Create ATM QoS Profile Object: `ObjectDeployPrimitive`
   b. If not found, Create Technology Object: `ObjectDeployPrimitive`
   – Create ATM QoS Profile Object: `ObjectDeployPrimitive`
2. Create ATM QoS Object: `ObjectDeployPrimitive`
3. Set Profile Attributes: `SetProfileAttribute`

CNOM Internal Work Flow for modifyNI2ATMQoSProfile

Initial State: `CDMDmmCtrlr:DSLAM.findATMQoSProfileObject_modifyNI2ATMQoSProfile`

1. Find ATM QoS Profile by Name: `ResolveName`
2. Find ATM QoS Object: `GetDescendantPrimitive`
3. Modify Attributes: `SetAttributesPrimitive`

CNOM Internal Work Flow for deleteNI2ATMQoSProfile

Initial State: `CDMDmmCtrlr:DSLAM.findATMQoSProfileObject_deleteNI2ATMQoSProfile`

1. Find ATM QoS Profile by Name: `ResolveName`
2. Delete ATM QoS Profile Object: `ObjectDeletePrimitive`

Cisco 6400 UAC Profile and Service Policies

Cisco 6400 UAC service profiles and service management are provided through the SCM GUI, and policies are provided by SCM NOA modules.

The profile policies are used for deploying service profile objects and configuring the service values. The service policies are used for deploying service objects, applying service profiles, and commissioning the service by configuring the node. The modify service policies are used for decommissioning the service to remove existing configuration from the node, applying the service profiles, and recommissioning the service to configure the node with new values. When service uplinks are used as in L2TP services, you must deploy and configure both the service object and service uplink objects.

The SCM NOA module supports the following service and profile management:

- PPPoA Service Management
- L2TP Service Management
- Bridging Service Management
- IRB Service Management
- RFC 1483 Service Management

For each of the above services, CNOM provides the following policies:

- `CreateProfile` policy—Creates a profile for the specified service type in the *profileContainment* view. Use the profile to configure a service through service management policies, or through the SCM dialogs.
- `ModifyProfile` policy—Modifies an existing profile for the specified service type in the *profileContainment* view.
- `DeleteProfile` policy—Deletes an existing profile.
- `CreateService` policy—Creates a service object in a given chassis, applies a specified profile to the service, and commissions the service for creating connections for the service.
- `ModifyService` policy—Modifies an existing service object by:
  - Decommissioning the service if it is already commissioned
  - Applying a service profile to the decommissioned service
  - Recommissioning the service
- `DeleteService` policy—Decommissions the service object if it is commissioned and deletes the service object from the view.

Manage connection templates using profile policies; do not deploy them as services.

PPPoA Service Management

Use the policy `createPPPoAServiceProfile` to create a PPPoA profile. The policy expects the following parameters:

- `workOrder:DDMWorkOrder.ProfileName`
- `workOrder:DDMWorkOrder.vtEthernetPortType`
- `workOrder:DDMWorkOrder.vtIPAddress`
- `workOrder:DDMWorkOrder.vtIPSubnetMask`
- `workOrder:DDMWorkOrder.pppipPeerDHCP`
- `workOrder:DDMWorkOrder.vtAuthenticationType`
- `workOrder:DDMWorkOrder.lowerIPAddress`
- `workOrder:DDMWorkOrder.higherIPAddress`
- `workOrder:DDMWorkOrder.pppipCustEncaps`

Use the policy `modifyPPPoAServiceProfile`, with the parameters above, to modify an existing profile. Only the profile object is updated. You must apply the profile to an existing service object to effect the change in a service object.

Use the policy `deletePPPoAServiceProfile` to delete the profile specified by `workOrder:DDMWorkOrder.ProfileName`.

Use the policy `createPPPoAServiceFromProfile` to create a service object in a given chassis. The policy expects the following parameters:

- `workOrder:DDMWorkOrder.NRPContainmentPath`—The containment path of the NRP object in AV under which the service is to be created.
- `workOrder:DDMWorkOrder.ProfileContainmentPath`—The containment path of the profile object (for example, `profileContainment:/Cisco6400.PPPIP/<`*`profile-name`*`>`).
- `workOrder:DDMWorkOrder.serviceName`—The name of the service. This name is also asserted in the attribute `LocalDB:C6400DMM.serviceName` and is used by CNOM to identify the service in connection management.

Use the policy `modifyPPPoAServiceFromProfile`, with the parameters above, to modify an existing service.

The policy `deletePPPoAService` decommissions the specified service and deletes it from SCM.

The following scripts provide a wrapper around the policies for service management:

- `create_PPPoA_profile`
- `modify_PPPoA_profile`
- `delete_PPPoA_profile`
- `create_PPPoA_service`
- `modify_PPPoA_service`
- `delete_PPPoA_service`

CNOM Internal Work Flow for createPPPOAServiceProfile

Initial State: `C6400DMMCtrller:C6400DMM.findTechnologyObject_createPPPOAServiceProfile`

1. Find the Technology Object Name: `ResolveName`
   a. If found, Create Profile Object: `ObjectDeployPrimitive`
   b. If not found, Create Technology Object: `ObjectDeployPrimitive`
   – Create Profile Object: `ObjectDeployPrimitive`

2. Create PPPOA Profile Object: `ObjectDeployPrimitive`
3. Set PPPOA Profile Attributes: `SetProfileAttribute`

CNOM Internal Work Flow for createPPPOAServiceFromProfile

Initial State: `C6400DMMCtrller:C6400DMM.findNRP_createPPPOAServiceFromProfile`

1. Find NRP by Name: `ResolveName`
2. Find Cisco 6400 UAC Chassis from NRP Name: `GetAncestorPrimitive`
3. Find Profile Object: `ResolveName`
4. Create Service Object: `ObjectDeployPrimitive`
5. Apply Service Profile: `ApplyProfile`
6. Commission Service: `InvokeActionPrimitive`

CNOM Internal Work Flow for deletePPPOAService

Initial State: `C6400DMMCtrller:C6400DMM.findServiceObject_deletePPPOAService`

1. Find Service Object: `FindObject`
2. Determine Commission State: `Annotate`
   a. If commissioned—Decommission Service: `InvokeActionService`
   b. If decommissioned—Continue to Step 3
3. Delete Service Object: `ObjectDeletePrimitive`

CNOM Internal Work Flow for deletePPPOAServiceProfile

Initial State: `C6400DMMCtrller:C6400DMM.findServiceProfile_deletePPPOAServiceProfile`

1. Find Service Profile Object by Name: `ResolveName`
2. Delete Service Profile Object: `ObjectDeletePrimitive`

CNOM Internal Work Flow for modifyPPPOAServiceProfile

Initial State: `C6400DMMCtrller:C6400DMM.findServiceProfile_modifyPPPOAServiceProfile`

1. Find Service Profile Object by Name: `ResolveName`
2. Find Service Profile Object: `GetDescendantPrimitive`
3. Update Profile Services Attributes: `SetAttributesPrimitive`

CNOM Internal Work Flow for modifyPPPOAServiceFromProfile

Initial State:
`C6400DMMCtrller:C6400DMM.findSourceServiceProfile_modifyPPPOAServiceFromProfile`

1. Find Service Profile: `ResolveName`
2. Find Service Object: `FindObject`
3. Find NRP Object: `ResolveName`

4. Determine Commission State: `Annotate`
   a. If commissioned—Decommission Service: `InvokeActionService`
   b. If decommissioned—Continue to Step 5
5. Apply Service Profile: `ApplyProfile`
6. Set Service Parameters: `SetAttributesPrimitive`
7. Commission Service: `InvokeActionPrimitive`

L2TP Service Management

Use the policy `createL2TPServiceProfile` to create an L2TP profile. The policy expects the following parameters:

- `workOrder:DDMWorkOrder.ProfileName`
- `workOrder:DDMWorkOrder.l2tpDomainName`
- `workOrder:DDMWorkOrder.l2tpTunnelNumber`
- `workOrder:DDMWorkOrder.l2tpCustEncaps`
- `workOrder:DDMWorkOrder.l2tpIPAddress`
- `workOrder:DDMWorkOrder.vtAuthenticationType`
- `workOrder:DDMWorkOrder.l2tpAuthName`
- `workOrder:DDMWorkOrder.l2tpAuthPassword`
- `workOrder:DDMWorkOrder.vcEncapsType`
- `workOrder:DDMWorkOrder.vcIPAddress`
- `workOrder:DDMWorkOrder.subIfIPAddress`
- `workOrder:DDMWorkOrder.subIfSubnetMask`
- `workOrder:CiscoGenericEM-MIB.QoSCategoryTrans`
- `workOrder:CiscoGenericEM-MIB.transCDVT`
- `workOrder:CiscoGenericEM-MIB.transMBS`
- `workOrder:CiscoGenericEM-MIB.transPCR`
- `workOrder:CiscoGenericEM-MIB.transSCR`
- `workOrder:CiscoGenericEM-MIB.transMCR`
- `workOrder:CiscoGenericEM-MIB.QoSCategoryRecv`
- `workOrder:CiscoGenericEM-MIB.recvCDVT`
- `workOrder:CiscoGenericEM-MIB.recvMBS`
- `workOrder:CiscoGenericEM-MIB.recvPCR`
- `workOrder:CiscoGenericEM-MIB.recvSCR`
- `workOrder:CiscoGenericEM-MIB.recvMCR`
- `workOrder:DDMWorkOrder.autoVpiVci`

Use the policy `modifyL2TPServiceProfile`, with the parameters above, to modify an existing profile in the *profileContainment* view. Only the profile object is updated. You must apply the profile to an existing service object to effect the change in a service object.

Use the policy `deleteL2TPServiceProfile` to delete the profile specified by the parameter `workOrder:DDMWorkOrder.ProfileName`.

Use the policy `createL2TPServiceFromProfile` to create a service object in a given chassis. The policy expects the following parameters:

- `workOrder:DDMWorkOrder.NRPContainmentPath`—The containment path of the NRP object in AV under which the service is created.
- `workOrder:DDMWorkOrder.ProfileContainmentPath`—The containment path of the profile object (for example, `profileContainment:/Cisco6400.L2TP/<`*`profile-name`*`>`).
- `workOrder:DDMWorkOrder.serviceName`—The name of the service. This name is also asserted in the attribute `LocalDB:C6400DMM.serviceName` and is used by CNOM to identify the service in connection management.
- `workOrder:DDMWorkOrder.ATMPortContainmentPath`—The containment path of the ATM Line Port for the uplink tunneling.
  - `workOrder:DDMWorkOrder.UplinkVPI`—The uplink VPI value.
  - `workOrder:DDMWorkOrder.UplinkVCI`—The uplink VCI value.
  - `workOrder:C6400SSControl-MIB.allowOverSub`—Indicates if the L2TP tunnel allows oversubscription. The possible values are: Allow and Prevent.

Use the policy `modifyL2TPServiceFromProfile`, with the parameters above, to modify an existing service.

The following scripts provide a wrapper around the policies for service management:

- `create_L2TP_profile`
- `modify_L2TP_profile`
- `delete_L2TP_profile`
- `create_L2TP_service`
- `modify_L2TP_service`
- `delete_L2TP_service`

Note If you create a profile using scripts, you must also delete the profile by using scripts. If you create a profile through the GUI, you must also delete it through the GUI.

CNOM Internal Work Flow for createL2TPServiceProfile

Initial State: `C6400DMMCtrller:C6400DMM.findTechnologyObject_createL2TPServiceProfile`

1. Find the Technology Object Name: `ResolveName`
   a. If found, Create Profile Object: `ObjectDeployPrimitive`
   b. If not found, Create Technology Object: `ObjectDeployPrimitive`
      – Create Profile Object: `ObjectDeployPrimitive`
2. Create Uplink Profile Object and set its attributes: `ObjectDeployPrimitive`
3. Create L2TP Profile Object and set its attributes: `ObjectDeployPrimitive`
4. Set L2TP Profile Attributes: `SetProfileAttribute`

CNOM Internal Work Flow for createL2TPServiceFromProfile

Initial State: `C6400DMMCtrller:C6400DMM.findNRP_createL2TPServiceFromProfile`

1. Find NRP by Name: `ResolveName`
2. Find Cisco 6400 UAC Chassis from NRP Name: `GetAncestorPrimitive`
3. Find Uplink Port: `ResolveName`
4. Find Profile Object: `ResolveName`
5. Create Service Object: `ObjectDeployPrimitive`
6. Create Uplink Object: `ObjectDeployPrimitive`
7. Apply Service Profile: `ApplyProfile`
8. Apply Uplink Profile: `ApplyProfile`
9. Commission Service: `InvokeActionPrimitive`

CNOM Internal Work Flow for deleteL2TPService

Initial State: `C6400DMMCtrller:C6400DMM.findServiceObject_deleteL2TPService`

1. Find Service Object: `FindObject`
2. Determine Commission State: Annotate
   a. If commissioned—Decommission Service: `InvokeActionService`
   b. If decommissioned—Continue to Step 3
3. Delete Service Object: `ObjectDeletePrimitive`

CNOM Internal Work Flow for deleteL2TPServiceProfile

Initial State: `C6400DMMCtrller:C6400DMM.findServiceProfile_deleteL2TPServiceProfile`

1. Find Service Profile Object by Name: `ResolveName`
2. Delete Service Profile Object: `ObjectDeletePrimitive`

CNOM Internal Work Flow for modifyL2TPServiceProfile

Initial State: `C6400DMMCtrller:C6400DMM.findServiceProfile_modifyL2TPServiceProfile`

1. Find Object Profile by Name: `ResolveName`
2. Find Profile Uplink Object: `GetDescendantPrimitive`
3. Find Service Profile Object: `GetDescendantPrimitive`
4. Update Profile Uplink Attributes: `SetAttributesPrimitive`
5. Update Profile Services Attributes: `SetAttributesPrimitive`

CNOM Internal Work Flow for modifyL2TPServiceFromProfile

Initial State:
`C6400DMMCtrller:C6400DMM.findSourceServiceFromProfile_modifyL2TPServiceFromProfile`

1. Find Service Profile: `ResolveName`
2. Find Service Object: `FindObject`
3. Find Uplink Object: `GetDescendantPrimitive`
4. Find NRP Object: `ResolveName`
5. Find Uplink Port: `ResolveName`
6. Determine Commission State: Annotate
   a. If commissioned—Decommission Service: `InvokeActionService`
   b. If decommissioned—Continue to Step 7
7. Apply Service Profile: `ApplyProfile`
8. Apply Uplink Profile: `ApplyProfile`
9. Set Service Parameters: `SetAttributesPrimitive`
10. Set Uplink Parameters: `SetAttributesPrimitive`
11. Commission Service: `InvokeActionPrimitive`

Bridging Service Management

Use the policy `createBridgingServiceProfile` to create a bridged services profile. The policy expects the following parameters:

- `workOrder:DDMWorkOrder.ProfileName`
- `workOrder:DDMWorkOrder.bridgeProtocol`
- `workOrder:DDMWorkOrder.bridgeGroup`
- `workOrder:DDMWorkOrder.vcEncapsType`
- `workOrder:DDMWorkOrder.subPolicyARP`
- `workOrder:DDMWorkOrder.subPolicyBroadcast`
- `workOrder:DDMWorkOrder.subPolicyMulticast`
- `workOrder:DDMWorkOrder.subPolicyUnknownDest`
- `workOrder:DDMWorkOrder.subPolicySTP`
- `workOrder:DDMWorkOrder.subPolicyCDP`
- `workOrder:CiscoGenericEM-MIB.qoSCategoryTrans`
- `workOrder:CiscoGenericEM-MIB.transCDVT`
- `workOrder:CiscoGenericEM-MIB.transMBS`
- `workOrder:CiscoGenericEM-MIB.transPCR`
- `workOrder:CiscoGenericEM-MIB.transSCR`
- `workOrder:CiscoGenericEM-MIB.transMCR`
- `workOrder:CiscoGenericEM-MIB.qoSCategoryRecv`
- `workOrder:CiscoGenericEM-MIB.recvCDVT`
- `workOrder:CiscoGenericEM-MIB.recvMBS`
- `workOrder:CiscoGenericEM-MIB.recvPCR`
- `workOrder:CiscoGenericEM-MIB.recvSCR`

- `workOrder:CiscoGenericEM-MIB.recvMCR`
- `workOrder:DDMWorkOrder.autoVpiVci`

Use the policy `modifyBridgingServiceProfile`, with the parameters above, to modify an existing profile in the *profileContainment* view. Only the profile object is updated. You must apply the profile to an existing service object to effect the change in a service object.

Use the policy `deleteBridgingServiceProfile` to delete the profile specified by the parameter `workOrder:DDMWorkOrder.ProfileName`.

Use the policy `createBridgingServiceFromProfile` to create a service object in a given chassis. The policy expects the following parameters:

- `workOrder:DDMWorkOrder.NRPContainmentPath`—The containment path of the NRP object in AV under which the service is to be created.
- `workOrder:DDMWorkOrder.ProfileContainmentPath`—The containment path of the profile object (for example, `profileContainment:/Cisco6400.BB/<`*`profile-name`*`>`).
- `workOrder:DDMWorkOrder.serviceName`—The name of the service. This name is also asserted in the attribute `LocalDB:C6400DMM.serviceName` and is used by CNOM to identify the service in connection management.
- `workOrder:DDMWorkOrder.ATMPortContainmentPath`—The containment path of the ATM Line Port for the uplink.
  - `workOrder:DDMWorkOrder.UplinkVPI`—The uplink VPI value.
  - `workOrder:DDMWorkOrder.UplinkVCI`—The uplink VCI value.
  - `workOrder:C6400SSControl-MIB.allowOverSub`—Indicates if the service allows oversubscription. The possible value are: Allow and Prevent.

Use the policy `modifyBridgingServiceFromProfile`, with the parameters above to modify an existing service.

The following scripts provide a wrapper around the policies for service management:

- `create_B_profile`
- `modify_B_profile`
- `delete_B_profile`
- `create_B_service`
- `modify_B_service`
- `delete_B_service`

Note If you create a profile using scripts, you must delete the profile by using scripts. If you create a profile through the GUI, you must also delete it through the GUI.

CNOM Internal Work Flow for createBridgingServiceProfile

Initial State: `C6400DMMCtrller:C6400DMM.findTechnologyObject_createBridgingServiceProfile`

1. Find the Containment Object for the Profile: `ResolveName`
   a. If found, create Profile Object: `ObjectDeployPrimitive`
   b. If not found, create Containment Object: `ObjectDeployPrimitive`
   – Create Profile Object: `ObjectDeployPrimitive`
2. Create Uplink Profile Object and set its attributes: `ObjectDeployPrimitive`

3. Create Bridging Service Profile Object and set its attributes: `ObjectDeployPrimitive`
4. Set Bridging Profile Attributes: `SetProfileAttribute`

CNOM Internal Work Flow for createBridgingServiceFromProfile

Initial State: `C6400DMMCtrller:C6400DMM.findNRP_createBridgingServiceFromProfile`

1. Find NRP Object: `ResolveName`
2. Find Cisco 6400 UAC Chassis Object: `GetAncestorPrimitive`
3. Find Uplink Port Object: `ResolveName`
4. Find Profile Object: `ResolveName`
5. Create Service Object: `ObjectDeployPrimitive`
6. Create Uplink Service Object: `ObjectDeployPrimitive`
7. Apply Service Object: `ApplyProfile`
8. Apply Uplink Service Object: `ApplyProfile`
9. Commission Service: `InvokeActionPrimitive`

CNOM Internal Work Flow for deleteBridgingService

Initial State: `C6400DMMCtrller:C6400DMM.findServiceObject_deleteBridgingService`

1. Find Service Object: `FindObject`
2. Determine Commission State: Annotate
   a. If commissioned—Decommission Service: `InvokeActionService`
   b. If decommissioned—Continue to Step 3
3. Delete Service Object: `ObjectDeletePrimitive`

CNOM Internal Work Flow for deleteBridgingServiceProfile

Initial State: `C6400DMMCtrller:C6400DMM.findServiceProfile_deleteBridgingServiceProfile`

1. Find Profile Object by Name: `ResolveName`
2. Delete Profile Object: `ObjectDeletePrimitive`

CNOM Internal Work Flow for modifyBridgingServiceProfile

Initial State: `C6400DMMCtrller:C6400DMM.findServiceProfile_modifyBridgingServiceProfile`

1. Find Object Profile by Name: `ResolveName`
2. Find Profile Uplink Object: `GetDescendantPrimitive`
3. Find Service Profile Object: `GetDescendantPrimitive`
4. Update Profile Uplink Attributes: `SetAttributesPrimitive`
5. Update Profile Services Attributes: `SetAttributesPrimitive`

CNOM Internal Work Flow for modifyBridgingServiceFromProfile

Initial State:
`C6400DMMCtrller:C6400DMM.findSourceServiceProfile_modifyBridgingServiceFromProfile`

1. Find Service Profile Object by name: `ResolveName`
2. Find Service Object: `FindObject`
3. Find Uplink Object: `GetDescendantPrimitive`
4. Find NRP Object: `ResolveName`
5. Find Uplink Port Object: `ResolveName`
6. Determine State of Uplink Port Object: `Annotate`
    a. If commissioned—Decommission the Object: `InvokeActionService`
    - Apply Profile changes: `applyProfile`
    - Apply Uplink Profile changes: `applyProfile`
    - Set Service Parameters: `setAttributesPrimitive`
    - Set Uplink Parameters: `setAttributesPrimitive`
    - Recommission Service: `InvokeActionPrimitive`

    b. If decommissioned—Continue to Step 7
7. Apply Service Profile: `ApplyProfile`
8. Apply Uplink Profile changes: `ApplyProfile`
9. Set Service Parameters: `SetAttributesPrimitive`
10. Set Uplink Parameters: `SetAttributesPrimitive`
11. Commission Service: `InvokeActionPrimitive`

IRB Service Management

Use the policy `createIRBServiceProfile` to create an IRB profile. The policy expects the following parameters:

- `workOrder:DDMWorkOrder.ProfileName`
- `workOrder:DDMWorkOrder.bridgeProtocol`
- `workOrder:DDMWorkOrder.bridgeGroup`
- `workOrder:DDMWorkOrder.vcEncapsType`
- `workOrder:DDMWorkOrder.subPolicyARP`
- `workOrder:DDMWorkOrder.subPolicyBroadcast`
- `workOrder:DDMWorkOrder.subPolicyMulticast`
- `workOrder:DDMWorkOrder.subPolicyUnknownDest`
- `workOrder:DDMWorkOrder.subPolicySTP`
- `workOrder:DDMWorkOrder.subPolicyCDP`
- `workOrder:DDMWorkOrder.bviIPAddress`
- `workOrder:DDMWorkOrder.bviSubnetMask`
- `workOrder:CiscoGenericEM-MIB.qoSCategoryTrans`
- `workOrder:CiscoGenericEM-MIB.transCDVT`
- `workOrder:CiscoGenericEM-MIB.transMBS`

- workOrder:CiscoGenericEM-MIB.transPCR
- workOrder:CiscoGenericEM-MIB.transSCR
- workOrder:CiscoGenericEM-MIB.transMCR
- workOrder:CiscoGenericEM-MIB.qoSCategoryRecv
- workOrder:CiscoGenericEM-MIB.recvCDVT
- workOrder:CiscoGenericEM-MIB.recvMBS
- workOrder:CiscoGenericEM-MIB.recvPCR
- workOrder:CiscoGenericEM-MIB.recvSCR
- workOrder:CiscoGenericEM-MIB.recvMCR
- workOrder:DDMWorkOrder.autoVpiVci

Use the policy modifyIRBServiceProfile, with the parameters above to modify an existing profile in the *profileContainment* view. Only the profile object is updated. You must apply the profile to an existing service object to effect the change in a service.

Use the policy deleteIRBServiceProfile to delete the profile specified by the parameter workOrder:DDMWorkOrder.ProfileName.

Use the policy createIRBServiceFromProfile to create a service object in a given chassis. The policy expects the following parameters:

- workOrder:DDMWorkOrder.NRPContainmentPath—The containment path of the NRP object in AV under which the service is created.
- workOrder:DDMWorkOrder.ProfileContainmentPath—The containment path of the profile object (for example, profileContainment:/Cisco6400.BR/*<profile-name>*).
- workOrder:DDMWorkOrder.serviceName—The name of the service. This name is also asserted in the attribute LocalDB:C6400DMM.serviceName and is used by CNOM to identify the service in connection management.
- workOrder:DDMWorkOrder.ATMPortContainmentPath—The containment path of the ATM line port for the uplink.
  - workOrder:DDMWorkOrder.UplinkVPI—The uplink VPI value.
  - workOrder:DDMWorkOrder.UplinkVCI—The uplink VCI value.
  - workOrder:C6400SSControl-MIB.allowOverSub—Indicates if the service allows oversubscription. The possible value are: Allow and Prevent.

Use the policy modifyIRBServiceFromProfile, with the following parameters, to modify an existing service:

- workOrder:DDMWorkOrder.bviMACAddress
- workOrder:DDMWorkOrder.brIPRoute

The following scripts provide a wrapper around the policies for service management:

- create_IRB_profile
- modify_IRB_profile
- delete_IRB_profile
- create_IRB_service
- modify_IRB_service
- delete_IRB_service

Note If you create a profile using scripts, you must delete the profile by using scripts. If you create a profile through the GUI, you must also delete the profile through the GUI.

CNOM Internal Work Flow for createIRBServiceProfile

Initial State: `C6400DMMCtrller:C6400DMM.findTechnologyObject_createIRBServiceProfile`

1. Find the Technology Object Name: `ResolveName`
    a. If found, create Profile Object: `ObjectDeployPrimitive`
    b. If not found, create Technology Object: `ObjectDeployPrimitive`
    – Create Profile Object: `ObjectDeployPrimitive`
2. Create Uplink Profile Object and set its attributes: `ObjectDeployPrimitive`
3. Create IRB Profile Object and set its attributes: `ObjectDeployPrimitive`
4. Set IRB Profile Attributes: `SetProfileAttribute`

CNOM Internal Work Flow for createIRBServiceFromProfile

Initial State: `C6400DMMCtrller:C6400DMM.findNRP_createIRBServiceFromProfile`

1. Find NRP by Name: `ResolveName`
2. Find Cisco 6400 UAC Chassis from NRP Name: `GetAncestorPrimitive`
3. Find Uplink Port: `ResolveName`
4. Find Profile Object: `ResolveName`
5. Create Service Object: `ObjectDeployPrimitive`
6. Create Uplink Object: `ObjectDeployPrimitive`
7. Apply Service Profile: `ApplyProfile`
8. Apply Uplink Profile: `ApplyProfile`
9. Commission Service: `InvokeActionPrimitive`

CNOM Internal Work Flow for deleteIRBService

Initial State: `C6400DMMCtrller:C6400DMM.findServiceObject_deleteIRBService`

1. Find Service Object: `FindObject`
2. Determine Commission State: `Annotate`
    a. If commissioned—Decommission Service: `InvokeActionService`
    b. If decommissioned—Continue to Step 3
3. Delete Service Object: `ObjectDeletePrimitive`

CNOM Internal Work Flow for deleteIRBServiceProfile

Initial State: `C6400DMMCtrller:C6400DMM.findServiceProfile_deleteIRBServiceProfile`

1. Find Service Profile Object by Name: `ResolveName`
2. Delete Service Profile Object: `ObjectDeletePrimitive`

CNOM Internal Work Flow for modifyIRBServiceProfile

Initial State: `C6400DMMCtrller:C6400DMM.findServiceProfile_modifyIRBServiceProfile`

1. Find Profile Object by Name: `ResolveName`
2. Find Profile Uplink Object: `GetDescendantPrimitive`
3. Find Service Profile Object: `GetDescendantPrimitive`
4. Update Profile Uplink Attributes: `SetAttributesPrimitive`
5. Update Profile Services Attributes: `SetAttributesPrimitive`

CNOM Internal Work Flow for modifyIRBServiceFromProfile

Initial State: `C6400DMMCtrller:C6400DMM.findSourceServiceProfile_modifyIRBServiceFromProfile`

1. Find Service Profile: `ResolveName`
2. Find Service Object: `FindObject`
3. Find Uplink Object: `GetDescendantPrimitive`
4. Find NRP Object: `ResolveName`
5. Find Uplink Port: `ResolveName`
6. Determine Commissioned State: `Annotate`
   a. If commissioned—Decommission Service: `InvokeActionService`
   b. If decommissioned—Continue to Step 7
7. Apply Service Profile: `ApplyProfile`
8. Apply Uplink Profile changes: `ApplyProfile`
9. Set Service Parameters: `SetAttributesPrimitive`
10. Set Uplink Parameters: `SetAttributesPrimitive`
11. Commission Service: `InvokeActionPrimitive`

RFC 1483 Service Management

Use the policy `createRFC1483ServiceProfile` to create an RFC 1483 profile. The policy expects the following parameters:

- `workOrder:DDMWorkOrder.ProfileName`
- `workOrder:DDMWorkOrder.subIfIPAddress`
- `workOrder:DDMWorkOrder.subIfSubnetMask`
- `workOrder:DDMWorkOrder.vcEncapsType`
- `workOrder:DDMWorkOrder.vcIPAddress`
- `workOrder:CiscoGenericEM-MIB.qoSCategoryTrans`
- `workOrder:CiscoGenericEM-MIB.transCDVT`
- `workOrder:CiscoGenericEM-MIB.transMBS`
- `workOrder:CiscoGenericEM-MIB.transPCR`
- `workOrder:CiscoGenericEM-MIB.transSCR`
- `workOrder:CiscoGenericEM-MIB.transMCR`
- `workOrder:CiscoGenericEM-MIB.qoSCategoryRecv`

- workOrder:CiscoGenericEM-MIB.recvCDVT
- workOrder:CiscoGenericEM-MIB.recvMBS
- workOrder:CiscoGenericEM-MIB.recvPCR
- workOrder:CiscoGenericEM-MIB.recvSCR
- workOrder:CiscoGenericEM-MIB.recvMCR
- workOrder:DDMWorkOrder.autoVpiVci

Use the policy modifyRFC1483ServiceProfile, with the parameters above to modify an existing profile in the *profileContainment* view. Only the profile object is updated. You must apply the profile to an existing service object to effect the change in a service.

Use the policy deleteRFC1483ServiceProfile to delete the profile specified by the parameter workOrder:DDMWorkOrder.ProfileName.

Use the policy createRFC1483ServiceFromProfile to create a service object in a given chassis. The policy expects the following parameters:

- workOrder:DDMWorkOrder.NRPContainmentPath—The containment path of the NRP object in AV under which the service is to be created.
- workOrder:DDMWorkOrder.ProfileContainmentPath—The containment path of the profile object (for example, profileContainment:/Cisco6400.BR/<*profile-name*>).
- workOrder:DDMWorkOrder.serviceName—The name of the service. This name is also asserted in the attribute LocalDB:C6400DMM.serviceName and is used by CNOM to identify the service in connection management.
- workOrder:DDMWorkOrder.ATMPortContainmentPath—The containment path of the ATM line port for the uplink:
  - workOrder:DDMWorkOrder.UplinkVPI—The uplink VPI value
  - workOrder:DDMWorkOrder.UplinkVCI—The uplink VCI value
  - workOrder:DDMWorkOrder.subIfIPAddress
  - workOrder:DDMWorkOrder.subIfSubnetMask

Use the policy modifyRFC1483ServiceFromProfile, with the parameters above to modify an existing service.

The following scripts provide a wrapper around the policies for service management:

- create_R_profile
- modify_R_profile
- delete_R_profile
- create_R_service
- modify_R_service
- delete_R_service

CNOM Internal Work Flow for createRFC1483ServiceProfile

Initial State: C6400DMMCtrller:C6400DMM.findTechnologyObject_createRFC1483ServiceProfile

1. Find the Technology Object Name: ResolveName
   a. If found, create Profile Object: ObjectDeployPrimitive
   b. If not found, create Technology Object: ObjectDeployPrimitive
   – Create Profile Object: ObjectDeployPrimitive

2. Create Uplink Profile Object and set its attributes: `ObjectDeployPrimitive`
3. Create RFC1483 Profile Object and set its attributes: `ObjectDeployPrimitive`
4. Set RFC1483 Profile Attributes: `SetProfileAttribute`

CNOM Internal Work Flow for createRFC1483ServiceFromProfile

Initial State: `C6400DMMCtrller:C6400DMM.findNRP_createRFC1483ServiceFromProfile`

1. Find NRP by Name: `ResolveName`
2. Find Cisco 6400 UAC Chassis from NRP Name: `GetAncestorPrimitive`
3. Find Uplink Port: `ResolveName`
4. Find Profile Object: `ResolveName`
5. Create Service Object: `ObjectDeployPrimitive`
6. Create Uplink Object: `ObjectDeployPrimitive`
7. Apply Service Profile: `ApplyProfile`
8. Apply Uplink Profile: `ApplyProfile`
9. Commission Service: `InvokeActionPrimitive`

CNOM Internal Work Flow for deleteRFC1483Service

Initial State: `C6400DMMCtrller:C6400DMM.findServiceObject_deleteRFC1483Service`

1. Find Service Object: `FindObject`
2. Determine Commission State: `Annotate`
   a. If commissioned—Decommission Service: `InvokeActionService`
   b. If decommissioned—Continue to Step 3
3. Delete Service Object: `ObjectDeletePrimitive`

CNOM Internal Work Flow for deleteRFC1483ServiceProfile

Initial State: `C6400DMMCtrller:C6400DMM.findServiceProfile_deleteRFC1483ServiceProfile`

1. Find Service Profile Object by Name: `ResolveName`
2. Delete Service Profile Object: `ObjectDeletePrimitive`

CNOM Internal Work Flow for modifyRFC1483ServiceProfile

Initial State: `C6400DMMCtrller:C6400DMM.findServiceProfile_modifyRFC1483ServiceProfile`

1. Find Profile Object by Name: `ResolveName`
2. Find Profile Uplink Object: `GetDescendantPrimitive`
3. Find Service Profile Object: `GetDescendantPrimitive`
4. Update Profile Uplink Attributes: `SetAttributesPrimitive`
5. Update Profile Services Attributes: `SetAttributesPrimitive`

CNOM Internal Work Flow for modifyRFC1483ServiceFromProfile

Initial State:
`C6400DMMCtrller:C6400DMM.findSourceServiceProfile_modifyRFC1483ServiceFromProfile`

1. Find Service Profile: `ResolveName`
2. Find Service Object: `FindObject`
3. Find Uplink Object: `GetDescendantPrimitive`
4. Find NRP Object: `ResolveName`
5. Find Uplink Port: `ResolveName`
6. Determine Commissioned State: `Annotate`
   a. If commissioned—Decommission Service: `InvokeActionService`
   b. If decommissioned—Continue to Step 7
7. Apply Service Profile: `ApplyProfile`
8. Apply Uplink Profile changes: `ApplyProfile`
9. Set Service Parameters: `SetAttributesPrimitive`
10. Set Uplink Parameters: `SetAttributesPrimitive`
11. Commission Service: `InvokeActionPrimitive`

Cisco WAN Manager Profile Policies

Cisco Wan Manager (CWM) profiles consist of WAN QoS parameter profiles and ATM QoS profiles. You can set up the profiles in CWM GUI, or through the policies supported by the CWM NOA.

WAN QoS Profile Policies

There are two policies that support WAN QoS profile management:

- `create_cwmQoS`
- `remove_cwm_QoS`

Use the policy `create_cwm_QoS` to create a WAN QoS profile. The policy expects the following attributes in the work order:

- `workOrder:DDMWorkOrder.connectionIdentifier`
- `workOrder:DDMWorkOrder.ldbsvConnSubType`
- `workOrder:DDMWorkOrder.profileName`
- `workOrder:DDMWorkOrder.CtrlState`
- `workOrder:DDMWorkOrder.ldbatmEndPointBCM`
- `workOrder:DDMWorkOrder.ldbatmEndPointCDVTZeroPlus1`
- `workOrder:DDMWorkOrder.ldbatmEndPointFGCRA`
- `workOrder:DDMWorkOrder.ldbatmEndPointFRTT`
- `workOrder:DDMWorkOrder.ldbatmEndPointICR`
- `workOrder:DDMWorkOrder.ldbatmEndPointICRTO`
- `workOrder:DDMWorkOrder.ldbatmEndPointMBS`

- `workOrder:DDMWorkOrder.ldbatmEndPointMCR`
- `workOrder:DDMWorkOrder.ldbatmEndPointMinAdjustPeriod`
- `workOrder:DDMWorkOrder.ldbatmEndPointNRM`
- `workOrder:DDMWorkOrder.ldbatmEndPointPCRZeroPlus1`
- `workOrder:DDMWorkOrder.ldbatmEndPointPercUtil`
- `workOrder:DDMWorkOrder.ldbatmEndPointPolicing`
- `workOrder:DDMWorkOrder.ldbatmEndPointRateDown`
- `workOrder:DDMWorkOrder.ldbatmEndPointRateUp`
- `workOrder:DDMWorkOrder.ldbatmEndPointSCRZeroPlus1`
- `workOrder:DDMWorkOrder.ldbatmEndPointTBE`
- `workOrder:DDMWorkOrder.ldbatmEndPointVSVD`
- `workOrder:DDMWorkOrder.ldbsvConnCellRouting`
- `workOrder:DDMWorkOrder.ldbsvConnClassOfService`
- `workOrder:DDMWorkOrder.ldbsvConnTrkAvoidType`
- `workOrder:DDMWorkOrder.ldbsvConnTrkAvoidZCS`
- `workOrder:DDMWorkOrder.ldbatmEndPointICR`

To create a WAN QoS profile (`ATMDMMQOS` object) use the following script:

```
./create_cwm_qos <ProfileName> <ProfileType> [<attribute=value>]
```

The supported profile types are listed in Table B-1.

Table B-1 WAN QoS Profile Types

Profile Type	Description
cbr1	Constant bit rate (cell loss priority [CLP] not applicable)
vbr1	Variable bit rate (CLP not applicable)
vbr2	Variable bit rate
vbr3	Variable bit rate (CLP enabled)
abr-fs	Available bit rate (CLP enabled; ForeSight enabled)
ubr-1	Unspecified bit rate (CLP disabled)
ubr-2	Unspecified bit rate (CLP enabled)
abr-1	Available bit rate

Caution You must adhere to valid attribute ranges when creating a WAN QoS profile (Table B-2).

Table B-2 WAN QoS Valid Attribute Ranges

Attribute Name	Type	Valid Attribute Ranges
`ldbsvConnSubType`	integer	(1 to 10,200) as the following: cbr1(1), vbr1(2), vbr2(3), vbr3(4), abr-fs(5), fr-fs(6), fr(7), ubr-1(8), ubr-2(9), abr-1(10), unknown(200)
`ldbsvConnClassOfService`	integer	(0 to 15)
`ldbsvConnTrkAvoidType`	pick-list	(none(1), satellite(2), terrestrial(3))

Table B-2 WAN QoS Valid Attribute Ranges (continued)

Attribute Name	Type	Valid Attribute Ranges
`ldbsvConnTrkAvoidZCS`	pick-list	(false(1), true(2))
`ldbsvConnCellRouting`	pick-list	(enable(1), disable(2))
`ldbatmEndPointPCRZeroPlus1`	integer	(10 to 1412832)
`ldbatmEndPointCDVTZeroPlus1`	integer	(0 to 5000000)
`ldbatmEndPointPercUtil`	integer	(1 to 100)
`ldbatmEndPointFGCRA`	pick-list	(enable, disable)
`ldbatmEndPointSCRZeroPlus1`	integer	(7 to 1412832)
`ldbatmEndPointMBS`	integer	(1 to 5000000)
`ldbatmEndPointBCM`	pick-list	(enable, disable)
`ldbatmEndPointICR`	integer	(0 to 1412832)
`ldbatmEndPointRateUp`	integer	(0 to 1412832)
`ldbatmEndPointRateDown`	integer	(1 to 32768)
`ldbatmEndPointMinAdjustPeriod`	integer	(1 to 255)
`ldbatmEndPointNRM`	integer	(2 to 256)
`ldbatmEndPointTBE`	integer	(0 to 1048320)
`ldbatmEndPointFRTT`	integer	(0 to 16700)
`ldbatmEndPointVSVD`	pick-list	(enable, disable)
`ldbatmEndPointPolicing`	pick-list	(policingVbr1(1), Vbr2(2), Vbr3(3), Pcrpic(4), none(5))
`ldbatmEndPointMCR`	integer	(0 to 1412832)
`ldbatmEndPointICRTO`	integer	(62 to 255000)

If you do not assign a value to an attribute, the default value is used.

Use the policy `remove_cwm_QoS` to remove a WAN QoS profile. The policy expects the following attributes in the work order:

- `workOrder:DDMWorkOrder.connectionIdentifier`
- `workOrder:DDMWorkOrder.profileName`

To remove a WAN QoS profile, use the following script:

```
./remove_cwm_qos <ProfileName>
```

WAN Connection Management

Use WAN Connection Management to create and remove connections in:

- a WAN cloud

WAN Cloud Connection

The policy `connectSubscriber_cwm` sets up a connection between ingress and egress ports on the appropriate nodes in a WAN cloud. The policy expects the following attributes in the work order:

- `workOrder:DDMWorkOrder.connectionIdentifier`
- `workOrder:DDMWorkOrder.ingressATMEndPointNodeName`
- `workOrder:DDMWorkOrder.ingressATMEndPointIfShelf`
- `workOrder:DDMWorkOrder.ingressATMEndPointSlot`
- `workOrder:DDMWorkOrder.ingressATMEndPointPort`
- `workOrder:DDMWorkOrder.ingressATMEndPointVpi`
- `workOrder:DDMWorkOrder.ingressATMEndPointVci`
- `workOrder:DDMWorkOrder.egressATMEndPointNodeName`
- `workOrder:DDMWorkOrder.egressATMEndPointIfShelf`
- `workOrder:DDMWorkOrder.egressATMEndPointSlot`
- `workOrder:DDMWorkOrder.egressATMEndPointPort`
- `workOrder:DDMWorkOrder.egressATMEndPointVpi`
- `workOrder:DDMWorkOrder.egressATMEndPointVci`
- `workOrder:DDMWorkOrder.atmServiceCharacteristics`

To run the policy successfully, satisfy the following conditions:

- CWM is installed and properly configured (on separate server)
- The ATM cloud is deployed
- Create WAN QoS profiles prior to creating any connections in a WAN cloud
- Ingress and egress VPI/VCI values are available

To set up the connection inside the WAN cloud, use the following script:

```
./connect_cwm <Connection_ID> <QoS type> <ingressNode> <ingressSlot> <ingressPort>
<ingressVpi> <ingressVci> <egressNode> <egressSlot> <egressPort> [<egressVpi> <egressVci>]
```

Disconnect CWM Connection

The policy `disconnect_cwm` eliminates a connection in a WAN cloud and removes it from CNOM view.

The policy expects the following attribute in the work order:

- `workOrder:DDMWorkOrder.connectionIdentifier`

You can also run this policy by using the script `disconnect_cwm` as follows:

```
./disconnect_cwm <Connection_ID>
```

APPENDIX C

Cisco DSL Manager Error Code Reference

This appendix details the error messages that the Cisco Network Order Manager may receive from Cisco DSL Manager (CDM) if there are problems processing the work order. Error messages are returned as attribute values in the internal work order. The values of attributes in the internal work order are only returned if the attribute `workOrder:DDM.finalWorkOrder` is set in initiating external work order.

Either or both of the following attributes are set in the internal work order if an error condition occurs.

- `primitiveErrorMessage`—Contains the error code generated by the first primitive to report an error
- `policyErrorMessage`—Contains an error message that is set inside the policy

Table C-1 lists error messages for each element manager or network manager that CNOM communicates with during policy execution.

Table C-1 CDM Error Codes, Possible Causes, and Corrective Actions

Error Code	Possible Cause	Corrective Action
CDM_101 Unable to get ingress port from work order.	Invalid work order. The attribute `workOrder:DDMWorkOrder.ingressTP` is not present in the work order.	Make sure you are invoking a valid policy.
CDM_102 Missing DSL Service Characteristics in work order.	Invalid work order. The attribute `workOrder:DDMWorkOrder.serviceCharacteristics` is not present in the work order.	Verify that the attribute is passed to the CDM Network Order Adapter (NOA) with the profile name.
CDM_103 Missing Connection Identifier value in work order.	Invalid work order. The attribute `workOrder:DDMWorkOrder.connectionIdentifier` is not present in the work order.	Verify that the attribute is specified with a unique connection ID value.
CDM_104 `CDMDMM:GetTrunk` Unable to get ingress port from work order.	Invalid work order. The attribute `workOrder:DDMWorkOrder.ingressTP` is not present in the work order.	Make sure you are invoking a valid policy.
CDM_105 Unable to get containment path of the chassis object in *Subtend* view.	The CDM NOA module looks in the *Subtend* view for details on DSLAM connectivity. All chassis objects appear in the *Subtend* view as follows: • Standalone chassis (which are not part of a subtend configuration) appear at the top level. • Subtended chassis appear under the network port to which they are connected.	Make sure the chassis belonging to the line port is present in the *Subtend* view. To set up a subtend configuration, use the **subtend** command.

Table C-1 CDM Error Codes, Possible Causes, and Corrective Actions (continued)

Error Code	Possible Cause	Corrective Action
CDM_106 Unable to get the chassis ID of the top level chassis in the subtend path.	Internal Error.	None.
CDM_107 Unable to get trunk port for the top level chassis in the subtend path.	The CDM NOA module looks in the *Subtend* view for details on DSLAM connectivity. All chassis objects appear in the *Subtend* view as follows: • Standalone chassis (which are not part of a subtend configuration) appear at the top level. • Subtended chassis appear under the network port to which they are connected.	Make sure the trunk port of the chassis is configured correctly. To do this, specify the value 1 for the attribute `<`*`Trunk-Port-Name`*`>` `LocalDB:LCM-Subtend-MIB.isTrunk`. To set up the configuration, run the **subtend** command. To set up a subtend configuration, use the **subtend** command.
CDM_108 Unable to get connection object ID from work order.	Invalid policy.	Check the policy for aliasing.
CDM_109 Invalid connection object ID in work order.	Invalid policy.	Check the policy.
CDM_110 Unable to get ingress port object ID from work order.	Invalid policy.	Check the policy.
CDM_111 Invalid ingress port object ID in work order.	Invalid policy.	Check the policy.
CDM_112 Unable to get egress port object ID from work order.	Invalid policy.	Check the policy.
CDM_113 Invalid egress port object ID in work order.	Invalid policy.	Check the policy.
CDM_114 Unable to get ingress VPI value from work order.	The attribute `workOrder:DDMWOrkOrder.ingressVPI` is missing from the work order.	Make sure the attribute is present in the work order.
CDM_115 Unable to get ingress VCI value from work order.	The attribute `workOrder:DDMWOrkOrder.ingressVCI` is missing from the work order.	Make sure the attribute is present in the work order.
CDM_116 Unable to get egress VPI value from work order.	The attribute `workOrder:DDMWOrkOrder.egressVPI` is missing from the work order.	Make sure the attribute is present in the work order.
CDM_117 Unable to get egress VCI value from work order.	The attribute `workOrder:DDMWOrkOrder.egressVCI` is missing from the work order.	Make sure the attribute is present in the work order.

Table C-1 CDM Error Codes, Possible Causes, and Corrective Actions (continued)

Error Code	Possible Cause	Corrective Action
CDM_118 Unable to get ATM service characteristics from work order.	The attribute `workOrder:DDMWorkOrder.serviceCharacteristics` is missing from the work order.	Verify that the attribute is passed to the CDM NOA along with the profile name.
CDM_119 Unable to get trunk port of a chassis in subtend path.	The CDM NOA module looks in the *Subtend* view for details on DSLAM connectivity. All chassis objects appear in the *Subtend* view as follows: • Standalone chassis (which are not part of a subtend configuration) appear at the top level. • Subtended chassis appear under the network port to which they are connected.	Make sure the trunk port of the chassis is configured correctly. To do this, specify the value 1 for the attribute `<Trunk-Port-Name> LocalDB:LCM-Subtend-MIB.isTrunk`. To set up a subtend configuration, run the **subtend** command.
CDM_120 Unable to get containment path of the chassis object.	Internal Error.	None.
CDM_121 Unable to get object IDs of chassis in the subtend containment path.	Internal Error.	Make sure the containment path is valid.
CDM_122 Unknown chassis was recognized in subtend path.	The CDM NOA supports only NI-2 chassis in *Subtend* view.	Make sure there are no chassis which are not DSLAMs in the *Subtend* view.
CDM_124 Invalid VPI/VCI combination.	The VPI/VCI cannot be 0/0.	Make sure the VPI/VCI values for both ingress and egress attributes are valid. Check the log files for more details.
CDM_125 A connection was not set correctly.	The CDM NOA cannot successfully set up the connection across the subtend path.	Check the *CDMDmmCtlr.log* file for more details.
CDM_126 Unable to deploy `CDMDmmConnection` object.	The CDM NOA cannot create a `CDMDmmConnection` object under `connectionObjects:/<workOrder:DDMWorkOrder.connectionIdentifier>`.	Check the *CDMDmmCtlr.log* file for more details.
CDM_127 Unable to get connection object from work order.	Internal Error.	None.
CDM_200 Unable to get class inheritance for ingress port.	The CDM NOA cannot identify the ingress port's class model due to an invalid port object ID.	None.
CDM_201 Ingress port type is not supported in CDM DMM.	The CDM NOA supports only CAP, DMT, and SDSL for NI-2.	Make sure the line ID is set correctly for the port by checking `get_object_for_line <line-id>`. The object returned by the node must be of one of the above types.

Table C-1 CDM Error Codes, Possible Causes, and Corrective Actions (continued)

Error Code	Possible Cause	Corrective Action
CDM_202 Unable to get chassis object for ingress port.	The given ingress port object is not represented correctly in the *ComponentManaged* view. Either the ingress port is not a valid NI-2 port or the object no longer exists.	Make sure the line ID is set correctly for the port by checking `get_object_for_line` *`<line-id>`*. The object returned by the node must be of one of the above types.
CDM_203 Unable to get VCL attributes from the ingress and egress ports. Make sure the ports are valid DSL ports.	The CDM NOA cannot obtain virtual channel link (VCL) details from the ingress and egress ports to deploy a PVC object. The object may not be a valid type supported by CDM, or it may not exist in the CEMF database any more. The ingress port must be a valid CDM line port or subtend port and the egress port must be a valid trunk port for NI-2.	Make sure the objects are valid. Check the *CDMDmmCtlr.log* file for more details.
CDM_204 Unable to get VPI/VCI values from the PVC object. PVC connect did not succeed. Check log files for more details.	The CDM ATM Connection Manager cannot set up the connection. Possible reasons: • VPI/VCI values are out of range. NI-2 modules support a VPI range of 0 to 255 and a VCI range of 32 to 65535 by default. • VPI/VCI values have been allocated on the egress or ingress side. • Connection parameters such as the ATM QoS parameters or network service access point (NSAP) address are not valid. • Private Network-to-Network Interface (PNNI) is not configured for the node, and the CDM Domain Manager Module (DMM) cannot set up a soft PVC. • Unable to communicate with the node due to invalid passwords. • Loss of communication with the node.	Check the attribute `workOrder:DDMWorkOrder.primitiveErrorMessage` for more details. Check the *CDMDmmCtlr.log* and *ATMCMController.log* files for more details.
CDM_205 Unable to get PVC commission state from the PVC object. PVC connect did not succeed. Check log files for more details.	The CDM ATM Connection Manager cannot set up the connection. Possible reasons: • VPI/VCI values are out of range. NI-2 modules support a VPI range of 0 to 255 and a VCI range of 32 to 65535 by default. • VPI/VCI values have been allocated on the egress or ingress side. • Connection parameters such as the ATM QoS parameters or NSAP address are not valid. • Unable to communicate with the node due to invalid passwords. • Loss of communication with the node.	Check the attribute `workOrder:DDMWorkOrder.primitiveErrorMessage` for more details. Check the *CDMDmmCtlr.log* and *ATMCMController.log* files for more details.

Table C-1 CDM Error Codes, Possible Causes, and Corrective Actions (continued)

Error Code	Possible Cause	Corrective Action
CDM_206 PVC object is not in normal state. PVC connect did not succeed. Element Manager reported the result: \n.	The CDM ATM ConnectionManager cannot set up the connection. Possible reasons: • VPI/VCI values are out of range. NI-2 modules support a VPI range of 0 to 255 and a VCI range of 32 to 65535 by default. • VPI/VCI values have been allocated on the egress or ingress side. • Connection parameters such as the ATM QoS parameters or NSAP address are not valid. • PNNI is not configured for the node, and the CDM DMM cannot set up a soft PVC. • Unable to communicate with the node due to invalid passwords. • Loss of communication with the node.	Check the attribute `workOrder:DDMWorkOrder.primitiveErrorMessage` for more details. Check the *CDMDmmCtlr.log* and *ATMCMController.log* files for more details.
CDM_207 Unable to get VPI/VCI values from PVC object.	The CDM ATM Connection Manager cannot set up the connection. Possible reasons: • VPI/VCI values are out of range. NI-2 modules support a VPI range of 0 to 255 and a VCI range of 32 to 65535 by default. • VPI/VCI values have been allocated on the egress or ingress side. • Connection parameters such as the ATM QoS parameters or NSAP address are not valid. • PNNI is not configured for the node, and the CDM DMM cannot set up a soft PVC. • Unable to communicate with the node due to invalid passwords. • Loss of communication with the node.	Check the attribute `workOrder:DDMWorkOrder.primitiveErrorMessage` for more details. Check the *CDMDmmCtlr.log* and *ATMCMController.log* files for more details.
CDM_208 Unable to get the NSAP address of the trunk port of the top level chassis. Make sure the value is set correctly.	The DMM module cannot retrieve the SNMP attribute with the NSAP address of the trunk port of the top node in a subtend environment. The CDM NOA uses this information to set up soft PVCs between the DSLAMs.	Make sure the node's ATM addresses are configured correctly.
CDM_209 Unable to complete deployment for PVC object.	The CDM NOA cannot deploy the PVC object for the ATM connection. Possible reasons: • The object name is not unique, possibly because the VPI/VCI is already being used in the node. • Connection parameters are not valid.	Check the *CDMDmmCtlr.log* and *ATMCMController.log* files for more details.
CDM_210 Deleting all PVCs failed.	The CDM NOA cannot delete all of the PVCs when attempting to remove the ATM connection.	Check the *CDMDmmCtlr.log* and *ATMCMController.log* files for more details.

Table C-1 CDM Error Codes, Possible Causes, and Corrective Actions (continued)

Error Code	Possible Cause	Corrective Action
CDM_211 Unable to apply ATM Profile for chassis. Element Manager returned the following report: \n.	The CDM NOA cannot apply the ATM profile.	Make sure the ATM profile is present and the connection parameters for the work order are valid.
CDM_212 Unable to connect PVC for work order. Element Manager reported the following result: /n	The CDM ATM Connection Manager cannot set up the connection. Possible reasons: • VPI/VCI values are out of range. NI-2 modules support a VPI range of 0 to 255 and a VCI range of 32 to 65535 by default. • VPI/VCI values have been allocated on the egress or ingress side. • The connection parameters such as the ATM QoS parameters or NSAP address are not valid. • PNNI is not configured for the node, and the CDM DMM cannot set up a soft PVC. • Unable to communicate with the node due to invalid passwords. • Loss of communication with the node.	Check the attribute `workOrder:DDMWorkOrder.primitiveErrorMessage` for more details. Check the CDMDmmCtlr.log and *ATMCMController.log* files for more details.
CDM_213 Unable to add PVC object to `CDMDMMconnection` object in *connectionObjects* view.	The CDM NOA cannot add a PVC object to the *connectionObject* view for display and disconnect.	Check the *CDMDmmCtlr.log* file for more details.
CDM_214 Unable to deploy PVC objects for work order.	The CDM NOA cannot add a PVC object to the *connectionObject* view for display or disconnect.	Check the *CDMDmmCtlr.log* file for more details.
CDM_215 Unable to check the SPVC Connection Status.	The CDM NOA cannot create an SPVC in the node. Possible reasons: • VPI/VCI values are out of range. NI-2 modules support a VPI range of 0 to 255 and a VCI range of 32 to 65535 by default. • VPI/VCI values have been allocated on the egress or ingress side. • The connection parameters such as the ATM QoS parameters or NSAP address are not valid. • PNNI is not configured for the node, and the CDM DMM cannot set up a soft PVC. • Unable to communicate with the node due to invalid passwords. • Loss of communication with the node.	Check the attribute `workOrder:DDMWorkOrder.primitiveErrorMessage` for more details. Check the *CDMDmmCtlr.log* and *ATMCMController.log* files for more details.

Table C-1 CDM Error Codes, Possible Causes, and Corrective Actions (continued)

Error Code	Possible Cause	Corrective Action
CDM_216 SPVC Connection did not succeed in the node.	The CDM NOA cannot create an SPVC in the node. Possible reasons: • VPI/VCI values are out of range. NI-2 modules support a VPI range of 0 to 255 and a VCI range of 32 to 65535 by default. • VPI/VCI values have been allocated on the egress or ingress side. • The connection parameters such as the ATM QoS parameters or NSAP address are not valid. • PNNI is not configured for the node, and the CDM DMM cannot set up a soft PVC. • One or more interfaces in the subtend path are down. • Unable to communicate with the node due to invalid passwords. • Loss of communication with the node.	Check the attribute `workOrder:DDMWorkOrder.primitiveErrorMessage` for more details. Check the *CDMDmmCtlr.log* and *ATMCMController.log* files for more details.
CDM_301 Ingress Port is of unknown type.	The CDM NOA supports only Carrier Amplitude Phase (CAP), Discrete MultiTone (DMT), and single-line digital subscriber line (SDSL).	Make sure the line ID is set correctly for the port. Make sure the ingress port is valid. Check `get_object_for_line` *`<line-id>`*. The object returned by the node must be of one of the types listed at left.
CDM_308 Unable to get subscriber object for ingress port.	Internal Error.	None.
CDM_311 Unable to delete the subscriber object for ingress port. Element Manager reported the following: \n	The CDM NOA cannot delete the PVC.	Check the *CDMDmmCtlr.log* and *C6100v30Ctrl.log* files for more details.
CDM_354 Unable to get value for a profile attribute.	The given profile object is not assigned proper values for attributes.	Check the *CDMDmmCtlr.log* file for more details.
CDM_401 Chassis containment path is not provided in the work order.	The chassis containment path is expected in the work order attribute `workOrder:DDMWorkOrder.nodeContainmentPath`.	Make sure the attribute is set correctly.
CDM_402 Chassis containment path is not in proper format.	The chassis containment path is expected in the work order attribute `workOrder:DDMWorkOrder.nodeContainmentPath` as the value `<treeName>:/<objectName>/<objectName>`	Make sure the attribute is set correctly.

Table C-1 CDM Error Codes, Possible Causes, and Corrective Actions (continued)

Error Code	Possible Cause	Corrective Action
CDM_403 Subtend port containment path not in proper format.	The port containment path is expected in the work order attribute `workOrder:DDMWorkOrder.subtendPortContainmentPath` as the value `<treeName>:/<objectName>/<objectName>`	Make sure the attribute is set correctly.
CDM_404 Chassis or port containment path is invalid. Check the containment paths.	The chassis and port containment paths are expected in work order attributes `workOrder:DDMWorkOrder.chassisContainmentPath` and `workOrder:DDMWorkOrder.subtendPortContainmentPath` as the value `<treeName>:/<objectName>/<objectName>`	Make sure the attributes are set correctly.
CDM_405 Given containment path is not a valid chassis object.	The chassis containment path object in the attribute `workOrder:DDMWorkOrder.chassisContainmentPath` was not an NI-2 chassis object.	Make sure the attribute represents a chassis object.
CDM_407 Unable to get the NI-2 module for the chassis. Make sure the chassis has been commissioned.	The chassis did not contain an NI-2 module in the *ComponentManaged* view.	Make sure the chassis has been commissioned and the NI-2 module has been discovered correctly.
CDM_408 Too many NI-2 modules for the chassis.	The CDM NOA found more than one NI-2 module for the given chassis.	None.
CDM_409 Unable to get the NI-2 ports for the chassis. Make sure the chassis has been commissioned.	The chassis did not contain an NI-2 module in the *ComponentManaged* view.	Make sure the chassis has been commissioned and the NI-2 module has been discovered correctly.
CDM_412 Unable to identify trunk and subtend ports for NI-2 chassis. Check log files.	The CDM NOA uses the port number to identify the trunk port (with port number 1) and subtend port. The CDM NOA is not able to retrieve this information from CEMF database.	None.
CDM_414 Unable to mark the trunk and subtend port for the chassis.	The CDM NOA uses port numbers to identify the trunk and subtend ports and marks the port using the attribute `LocalDB:CDM-Subtend-MIB.isTrunk`.	None.
CDM_416 Unable to add the chassis and the ports to subtend tree. Make sure the subtending port if any is present in the view already.	The CDM NOA cannot add the chassis to the *Subtend* view, possibly because the parent subtending port is not present in the view.	Make sure the subtending port is present in the view.

Table C-1 CDM Error Codes, Possible Causes, and Corrective Actions (continued)

Error Code	Possible Cause	Corrective Action
Policy Error Messages		
CDM_501 Unable to identify the DSL port from the line ID for work order.	The CDM NOA cannot identify the DSL port for a given line ID.	Use `get_object_for_line <line-id>` to make sure the line ID is correctly set up for the port. This should return success with valid line ID value.
CDM_502 Unable to get trunk port of the top level chassis in subtend path.	The CDM NOA cannot identify the egress end point for the DSLAM subdomain.	Check the primitive error message in `workOrder:DDMWorkOrder.primitiveErrorMessage` for more details.
CDM_503 Unable to create connection object in *connectionObjects* view.	The CDM NOA cannot create the connection object with the name given in `workOrder:DDMWorkOrder.connectionIdentifier` in the *connectionObjects* view.	Make sure the connection identifier is unique and does not already exist in the *connectionObjects* view.
CDM_504 Unable to configure the DSL port with DSL service characteristics.	The CDM NOA cannot apply DSL service characteristics for the given port. Possible reasons: • Profile does not exist for the given name. • Given parameters are not valid for the port. • Unable to communicate with the node.	Check the primitive error message in `workOrder:DDMWorkOrder.primitiveErrorMessage` for more details.
CDM_505 Unable to set up PVCs in the subtend path.	The CDM NOA cannot set up a PVC in any of the nodes in the subtend path or could not set up a soft PVC in the leaf node.	Check the primitive error message in `workOrder:DDMWorkOrder.primitiveErrorMessage` for more details.
CDM_506 Unable to identify the object underneath the *connectionObjects* view with the name provided in `workOrder:DDMWorkOrder.connectionIdentifier`.	The CDM NOA cannot get the connection objects to initiate disconnect for the given connection identifier.	Make sure the connection identifier is valid.
CDM_507 Unable to delete PVC objects in subtend path.	The CDM NOA cannot delete the PVC in the node and/or CEMF database. Possible reasons: • PVC was already deleted from the node. • Unable to communicate with the node.	Check the primitive error message in `workOrder:DDMWorkOrder.primitiveErrorMessage` for more details.
CDM_508 Unable to delete the connection object. Check ddmGateway.log for more details.	The CDM NOA cannot delete the connection object from the view.	Check the *ddmGateway.log* file for more details.
CDM_524 Unable to modify the profile object. Check log for errors.	The CDM NOA cannot update the parameters in the profile object.	Check the *CDMDmmCtlr.log* file for more details.

Table C-1 CDM Error Codes, Possible Causes, and Corrective Actions (continued)

Error Code	Possible Cause	Corrective Action
CDM_525 Unable to create the profile parent `profileContainment:/Cisco` for adding profile objects.	The CDM NOA uses the technology name Cisco for NI-2 ATM QoS profiles. The CDM NOA could not add the technology object for adding the appropriate profile.	Check the log for errors.
CDM_526 Unable to create the profile object under parent `profileContainment:/Cisco`.	The CDM NOA cannot create the profiles for the CDM NI-2 module. A profile may already exist with this name.	Make sure a profile with this name does not already exist by using `get_ni2_profile ATMQoS <`*`profileName`*`>`.
CDM_527 Unable to create object for profile.	The CDM NOA cannot create the profile for the NI-2 module.	Check the *CDMDmmCtlr.log* file for more details.
CDM_528 Unable to find the profile object in *profileContainment* view under Cisco object.	The CDM NOA cannot find the profile in the *profileContainment* view.	Make sure the profile exists by using `get_ni2_profile ATMQoS <`*`profileName`*`>`.
CDM_529 Invalid profile.	Internal Error.	Internal Error. Profile must have an ATM `CiscoVirtualCircuit` object under the profile object.
CDM_530 Unable to modify the profile object with new values.	The CDM NOA cannot update the profile object.	Check the *CDMDmmCtlr.log* file for more details.
CDM_531 Unable to delete the object.	The CDM NOA cannot delete the NI-2 profile from the CEMF database.	Check the *CDMDmmCtlr.log* file for more details.
CDM_532 Unable to create the profile parent `profileContainment:/CAPTech` for adding profile objects.	The CDM NOA uses the technology name Cisco for NI-2 CAP profiles. The CDM NOA cannot add the technology object for adding the appropriate profile.	None.
CDM_533 Unable to create the profile object under parent `profileContainment:/CAPTech`. Make sure the profile name is unique.	The CDM NOA cannot create the profiles for the CDM NI-2 module. A profile may already exist with this name.	Make sure a profile with this name does not already exist by using `get_ni2_profile CAP <`*`profileName`*`>`.
CDM_534 Unable to find the profile object in *profileContainment* view under `CAPTech` object.	The CDM NOA cannot find the profile in the *profileContainment* view.	Make sure the profile exists by using `get_ni2_profile CAP <`*`profileName`*`>`.

Table C-1 CDM Error Codes, Possible Causes, and Corrective Actions (continued)

Error Code	Possible Cause	Corrective Action
CDM_535 Invalid profile.	Internal Error.	Internal Error. The profile must have an ATM `CiscoCAPFunctionality` object under the profile object.
CDM_536 Unable to create the profile parent `profileContainment:/SDSLTech` for adding profile objects.	The CDM NOA uses the technology name Cisco for NI-2 SDSL profiles. The CDM NOA cannot add the technology object for adding the appropriate profile.	Check the log for errors.
CDM_537 Unable to create the profile object under parent `profileContainment:/SDSLTech`.	The CDM NOA cannot create the profiles for the CDM NI-2 module. A profile may already exist with this name.	Use `get_ni2_profile SDSL <`*`profileName`*`>` to make sure a profile with this name does not already exist.
CDM_538 Unable to find the profile object in *profileContainment* view under `SDSLTech` object.	The CDM NOA cannot find the profile in the *profileContainment* view.	Make sure the profile exists using `get_ni2_profile SDSL <`*`profileName`*`>`.
CDM_539 Invalid profile.	Internal Error.	Internal Error. Profile must have a `CiscoSDSLFunctionality` object under the profile object.
CDM_540 Unable to create the profile parent `profileContainment:/ADSLIfManager` for adding profile objects.	The CDM NOA uses the technology name Cisco for NI-2 ADSL (subset of DMT) profiles. The CDM NOA cannot add the technology object for adding the appropriate profile.	Check the log for errors.
CDM_541 Unable to create the profile parent `profileContainment:/DMTTech` for adding profile objects.	The CDM NOA uses the technology name Cisco for NI-2 DMT profiles. The CDM NOA cannot add the technology object for adding the appropriate profile.	Check the log for errors.
CDM_542 Unable to create the profile object under parent `profileContainment:/ADSLIfManager`.	The CDM NOA cannot create the profiles for the CDM NI-2 module. A profile may already exist with this name.	Use `get_ni2_profile DMT <`*`profileName`*`>` to make sure a profile with this name does not already exist.
CDM_544 Unable to create the profile object under parent `profileContainment:/DMTTech`.	The CDM NOA cannot create the profiles for the CDM NI-2 module. A profile may already exist with this name.	Make sure the profile name is unique.

Table C-1 CDM Error Codes, Possible Causes, and Corrective Actions (continued)

Error Code	Possible Cause	Corrective Action
CDM_545 Unable to find the profile object in *profileContainment* view under `ADSLIfManager` object.	The CDM NOA cannot find the profile in the *profileContainment* view.	Make sure the profile exists by using `get_ni2_profile DMT <`*`profileName`*`>`.
CDM_546 Unable to find the profile object in *profileContainment* view under `DMTTech` object.	The CDM NOA cannot find the profile in the *profileContainment* view.	Make sure the profile exists by using `get_ni2_profile DMT <`*`profileName`*`>`.
CDM_547 Invalid profile. Profile must have a `CiscoADSLFunctionality` object underneath the profile object.	Internal Error.	None.
CDM_548 Invalid profile.	Internal Error.	Profile must have a `CiscoDMTFunctionality` object under the profile object.
CDM_550 Unable to modify the ADSL profile object with new values.	The CDM NOA cannot update the profile object.	Check the *CDMDmmCtlr.log* file for more details.
CDM_551 Unable to modify the DMT profile object with new values.	The CDM NOA cannot update the profile object.	Check the *CDMDmmCtlr.log* file for more details.
CDM_552 Unable to find the ADSL profile object in *profileContainment* view.	The CDM NOA cannot find the profile in the *profileContainment* view.	Make sure the profile exists by using `get_ni2_profile DMT <`*`profileName`*`>`.
CDM_553 Unable to find the DMT profile object in *profileContainment* view.	The CDM NOA cannot find the profile in the *profileContainment* view.	Make sure the profile exists by using `get_ni2_profile DMT <`*`profileName`*`>`.
CDM_554 Unable to delete the ADSL profile object.	The CDM NOA cannot delete the profile from the *profileContainment* view.	Check the *CDMDmmCtlr.log* file for more details.
CDM_555 Unable to delete the DMT profile object.	The CDM NOA cannot delete the profile from the *profileContainment* view.	Check the *CDMDmmCtlr.log* file for more details.
CDM_556 Unable to delete profile objects.	The CDM NOA cannot delete the profile from the *profileContainment* view.	Check the *CDMDmmCtlr.log* file for more details.
CDM_557 Unable to set up subtend.	The CDM NOA cannot set up subtending for the given node.	Check the attribute `workOrder:DDMWorkOrder.primitiveErrorMessage` for more details.

Table C-1 CDM Error Codes, Possible Causes, and Corrective Actions (continued)

Error Code	Possible Cause	Corrective Action
CDM_568 Identified port object does not belong to DSLAM subdomain.	The CDM NOA expects the line ports for the given line ID to be assigned the subdomain value DSLAM.	Use `get_object_for_line` *<line-id>* to make sure the line ID is set to the subdomain value of DSLAM.
CDM_569 No valid `L` attribute in the work order.	Ingress input must be provided either as line ID in the attribute `workOrder:DDMWorkOrder.ingressIdentifier` or as port containment path in the attribute `workOrder:DDMWorkOrder.ingressContainmentPath`.	Make sure that one of these attributes is present in the work order.
CDM_702 Error parsing string.	Corrupted attributes passed to the `restorePVC` primitive.	Check that the correct Data Recovery log information is supplied.
CDM_703 Unable to resolve TP paths.	Trying to restore a connection that refers to a nonexisting (removed) chassis.	Make sure the chassis exists.
CDM_704 Ingress/Egress paths cannot be the same.	Ingress and egress paths must be different. In the case of `restorePVC`, this error message may be the result of an internal error.	Check the log files.
CDM_705 Unable to tokenize.	Internal error.	Check the log files.
CDM_711 Unable to resolve attribute IDs for query.	Internal error.	None.
CDM_713 Query returned invalid object.	Internal error.	Check the log files.
CDM_714 Unable to get `ATMCMController:CEMCS-ATM-Conn_MIB.destNsapAddr` attribute. Check attribute lists.	No NSAP address attribute passed to the `restorePVC` primitive.	Check that the correct Data Recovery log information is supplied.
CDM_715 SNMP attribute missing. Check VCL attributes list.	Incorrect SNMP attributes were passed to the `restorePVC` primitive.	Check that the correct Data Recovery log information is supplied.
CDM_716 `DeployPVCobject` failed.	The PVC object cannot be deployed.	Check that the port object in *ComponentManaged* view includes a PVC object with the same name.
CDM_718 Unable to apply the QoS profile.	The ATM QOS profile cannot be applied. The profile may not exist.	Check that the profile exists.

Table C-1 CDM Error Codes, Possible Causes, and Corrective Actions (continued)

Error Code	Possible Cause	Corrective Action
CDM_721 Unable to add PVC object to connection object.	The PVC object cannot be added to the *connectionObjects* view.	Check that there are no objects with the same name.
CDM_751 Wrong PVC type.	An incorrect PVC type was supplied. There are two PVC types supported: • PVC • SPVC	Check that the correct Data Recovery log information was supplied.

APPENDIX D

Cisco Service Connection Manager Error Code Reference

This appendix details the error messages that the Cisco Network Order Manager may receive from Service Connection Manager (SCM) if there are problems processing the work order. Error messages are returned as attribute values in the internal work order. The values of attributes in the internal work order are only returned if the attribute `workOrder:DDM.finalWorkOrder` is set in initiating external work order.

Either or both of the following attributes are set in the internal work order if an error occurs.

- `primitiveErrorMessage`—Contains the error code generated by the first primitive to report an error
- `policyErrorMessage`—Contains an error message that is set inside the policy

Table D-1 lists error messages that relate to each element manager or network manager that CNOM communicates with during policy execution.

Table D-1 SCM Error Codes, Possible Causes, and Corrective Action

Error Code	Possible Cause	Corrective Action
SCM_101 Unable to get connection identifier string from the work order.	The attribute `workOrder:DDMWorkOrder.connectionIdentifier` is not present in the work order.	Make sure the attribute is supplied with a value to the work order.
SCM_102 The service name for connection is not present in the work order.	The service name is not present in the work order.	Make sure the attribute `workOrder:DDMWorkOrder.egressIdentifier` is set with the service name in the work order.
SCM_103 The ATM service characteristics name is not present in work order.	The service characteristics name is not present in the work order.	Make sure the attribute `workOrder:DDMWorkOrder.service Characteristics` is set in the ATM connection template name in the work order.
SCM_105 There are no identified ingress ports or service instances for this work order.	For a given ingress side (identified from a line ID associated with a Cisco 6400 UAC port), the node did not detect a service instance identified by `workOrder:DDMWorkOrder.egressIdentifier`.	Make sure there are available services that can support this connection for the given ingress port. Check the ingress name and service name. Check the *C6400DMMCtrller.log* file for more details.
	The NOA could not set up a connection from the ingress port to the service instances identified by `workOrder:DDMWorkOrder.egressIdentifier`.	

Table D-1 SCM Error Codes, Possible Causes, and Corrective Action (continued)

Error Code	Possible Cause	Corrective Action
SCM_106 Unable to get gateway CNOM connection object from work order attribute.	Internal Error.	None.
SCM_107 Unable to get service characteristics name for connection template from work order.	The service characteristics name is not present in the work order.	Make sure the attribute `workOrder:DDMWorkOrder.service Characteristics` is set as part of the ATM connection template name in the work order.
SCM_108 Unable to find connection template object for given service characteristics name.	CNOM cannot find an ATM QoS object with the given name in the SCM *Cisco6400ConnectionTemplate* view.	Expand the view *Cisco6400ConnectionTemplate* and check for an object with the given name.
SCM_109 Unable to get ingress TP from work order attribute.	Internal Error.	None.
SCM_110 Unable to get service instance object from work order attribute.	Internal Error.	None.
SCM_111 Unable to get NRP object from service.	Use the NRP object in the service instance for connection in case of SSG service. If the service is not configured and commissioned, CNOM fails and returns this error.	Make sure the NRP object is configured in the service instance and the service instance is fully configured and commissioned.
SCM_201 Unable to get ingress ports for connection for a given region or a Cisco 6400 UAC.	Given an egress identifier, CNOM identifies a possible set of ingress ports in a given chassis or region to perform connection. If you are unable to identify at least one ingress port, CNOM reports this error.	Make sure you configured the regions correctly for the Cisco 6400 UAC.
SCM_202 Unable to get service instances matching service name.	CNOM found a mismatch between the given service name in the work order attribute `workOrder:DDMWorkOrder.egressIdentifier` and the value of attribute `LocalDB:C6400DMM.serviceName` for services to identify the service for connection.	Make sure the attribute values are configured correctly and that the work order is set with correct values.
SCM_203 Unable to get NSP parent for the ingress port.	Internal Error.	Make sure the ingress port is a Cisco 6400 UAC port.
SCM_204 Unable to get NSP for the service instance.	Internal Error.	Make sure the service instance is a valid Cisco 6400 UAC service.

Table D-1 SCM Error Codes, Possible Causes, and Corrective Action (continued)

Error Code	Possible Cause	Corrective Action
SCM_205 Unable to get chassis object for the service instance.	Internal Error.	Make sure the service instance is a valid Cisco 6400 UAC service.
SCM_206 Unable to complete query for work order.	Internal Error.	None.
SCM_210 Unable to deploy subscriber object for egress tie-port.	The deployment context failed while it was creating the subscriber object.	Check the *C6400DMMCtrller.log* file for more details.
SCM_211 Unable to deploy subscriber object for ingress tie-port.	The deployment context failed while it was creating the subscriber object.	Check the *C6400DMMCtrller.log* file for more details.
SCM_212 Unable to get object ID of the deployed subscriber object from context.	Internal Error.	None.
SCM_213 Unable to get object ID of subscriber and `C6400DMMConnection` object from deployment context.	Internal Error.	None.
SCM_214 Unable to get object ID of `C6400DMMConnection` object from deployment context.	Internal Error.	None.
SCM_215 Unable to deploy subscriber and `C6400DMMConnection` objects for ingress TP.	The deployment context failed while it was creating the subscriber object.	Check the *C6400DMMCtrller.log* file for more details.
SCM_216 Unable to deploy `C6400DMMConnection` object under gateway connection object. Check log for errors.	The deployment context failed while it was creating the subscriber object.	Check the *C6400DMMCtrller.log* file for more details.
SCM_217 Unable to delete subscriber and connection objects for ingress TP.	The deployment context failed while it was deleting the subscriber object.	Check the *C6400DMMCtrller.log* file for more details.

Table D-1 SCM Error Codes, Possible Causes, and Corrective Action (continued)

Error Code	Possible Cause	Corrective Action
SCM_218 Unable to connect subscriber to service. Check action result for more details.	The connect action from SCM to the node failed. Possible reasons: • Unable to Telnet to the node due to network issue or incorrect passwords. • The VPI/VCI values are incorrect or have been already allocated. • The traffic resource parameters are not valid for the connection. • The service to which the connection is made has not been fully configured and commissioned.	Check the value of the attribute `workOrder:DDMWorkOrder.connectActionReport` for a list of the commands sent to the node and the response of the node.
SCM_219 Unable to get connection ID from action report. Cannot delete the connection.	Internal Error.	None.
SCM_220 Unable to get chassis parent object for deployment from work order.	The chassis parent object under which the ATM service is deployed is not present in the work order.	None.
SCM_222 Unable to get ATM Port for service instance deployment.	The ATM port object on which the ATM service is created is not present in the work order.	None.
SCM_223 Unable to get the egress VPI from work order.	The work order does not contain the attribute `workOrder:DDMWorkOrder.egressVPI`.	Make sure the attribute and its value are included in the work order.
SCM_224 Unable to get the egress VCI from the work order. Make sure the work order is set correctly.	The work order does not contain the attribute `workOrder:DDMWorkOrder.egressVCI`.	Make sure the attribute and its value are included in the work order.
SCM_225 Unable to get NOA parent object for deployment from work order.	Internal Error.	None.
SCM_226 Unable to get the name of the chassis object.	Internal Error.	None.

Table D-1 SCM Error Codes, Possible Causes, and Corrective Action (continued)

Error Code	Possible Cause	Corrective Action
SCM_227 Unable to completely deploy the ATM service. Check log files.	CNOM was unable to create the ATM service object for the connection.	Check the *C6400DMMCtrller.log* file for more details.
SCM_501 Unable to create SCM connection template.	CNOM was unable to create the ATM QoS object under the SCM *Cisco6400ConnectionTemplate* view. A connection template with the specified name may already exist.	Make sure the connection template name is unique. Check this by expanding the view *Cisco6400ConnectionTemplate* and checking for an object with the given name.
SCM_502 Unable to find SCM connection template with the given name.	CNOM was unable to find the ATM QoS object in the SCM *Cisco6400ConnectionTemplate* view.	Make sure the connection template exists by expanding the *Cisco6400ConnectionTemplate* view and checking for an object with the given name.
SCM_503 Unable to create connection object.	Unable to create a connection object with the name provided in the work order attribute `workOrder:DDMWorkOrder.connectionIdentifier` in the *connectionObjects* view.	Make sure a connection already exists for the specified name and is unique. Connection identifiers must be unique and cannot be reused unless the earlier connection is torn down.
SCM_504 Unable to deploy `C6400DMMServices` object.	The deployment failed when it attempted to create the `C6400DMMServices` object under the connection object in the *connectionObjects* view.	Refer to the *6400DMMCtrller.log* file.
SCM_505 Unable to identify the CLEC port from line ID.	The policy cannot convert the line ID to a valid line port for a Cisco 6400 UAC. To set line IDs for a port, use the `set_line_ids` and `add_line_id` commands.	Make sure the line ID is set correctly and configured uniquely for a port using `get_object_for_line`. The command accepts the line ID as a parameter and returns the containment path of the port with that line ID. If the command returns a: • Failure—The line ID is not assigned to any port. • Success—Check the containment path to verify that the port is valid and the subdomain ID is set to C6400DMM.
SCM_506 Invalid CLEC Port. Unable to get the Cisco 6400 UAC chassis object for the CLEC port.	The port returned from the egress identifier line ID value is not a valid Cisco 6400 UAC port, or the port no longer exists.	Make sure the line ID is set correctly and configured uniquely for a port using `get_object_for_line`. The command accepts the line ID as a parameter and returns the containment path of the object with that line ID. If the command returns a: • Failure—The object is no longer valid. • Success—Check the containment path to ensure that this is a valid Cisco 6400 UAC port.

Table D-1 SCM Error Codes, Possible Causes, and Corrective Action (continued)

Error Code	Possible Cause	Corrective Action
SCM_507 Unable to identify RT ingress port from ingress line ID.	The policy cannot convert the given line ID to a valid line port for a Cisco 6400 UAC. To set line IDs for a port, use the `set_line_ids` and `add_line_id` commands.	Make sure the line ID is set correctly and uniquely for a port using `get_object_for_line`. The command takes the line ID as a parameter and returns the containment path of the object with that line ID. If the command returns a: • Failure—The object is no longer valid. • Success—Check the containment path to ensure that this is a valid Cisco 6400 UAC port.
SCM_508 Invalid RT Ingress Port. Unable to get Cisco 6400 UAC chassis for the port object.	The port returned from the egress identifier line ID value is not a valid Cisco 6400 UAC port, or the port no longer exists.	Make sure the line ID is set correctly and uniquely for a port using `get_object_for_line`. The command takes the line ID as a parameter and returns the containment path of the object with that line ID. If the command returns a: • Failure—The object is no longer valid. • Success—Check the containment path to ensure that this is a valid Cisco 6400 UAC port.
SCM_509 Internal error. Policy is invalid.	Work order attributes are either missing from the policy or assigned invalid values.	Make sure all of the work order attributes are present in the policy and valid values are provided for the attributes.
SCM_510 Unable to find IMT port for work order.	The policy identified two Cisco 6400 UAC chassis (an RT chassis and a CLEC chassis) for connection but did not find an inter-machine tie (IMT) connection between the two chassis.	Make sure the IMT port is configured correctly for the RT chassis. Run `set_IMT` to set up the IMT connection between the two chassis.
SCM_511 Unable to deploy ATM service for the chassis.	The ATM service cannot be deployed for the chassis.	SCM policy deploys an ATM service with the name `<`*`ChassisName`*`>.<`*`ConnectionIdentifier`*`>-ATM Service`, where *`ChassisName`* is the name of the chassis object in the service view *Cisco6400ServiceView* and *`ConnectionIdentifier`* is the value of the work order attribute `workOrder:DDMWorkOrder.connection Identifier`. Make sure the service view already includes a service deployed with this name. Check the *C6400DMMCtrller.log* file for more details.

Table D-1 SCM Error Codes, Possible Causes, and Corrective Action (continued)

Error Code	Possible Cause	Corrective Action
SCM_513 Unable to connect subscriber to ATM service.	The NOA failed to connect an ingress VPI/VCI pair to an egress VPI/VCI pair.	Check the work order attributes `workOrder:DDMWorkOrder.primitiveErrorMessage` and `workOrder:DDMWorkOrder.connectActionReport` for more details. The `primitiveErrorMessage` attribute contains a specific error message that is encountered by the `ConnectSubscriber` primitive when it tries to deploy a subscriber object and connects the subscriber to the given ATM service. The `connectActionReport` attribute contains the details of the IOS commands that were sent to the node and possible errors in the Cisco IOS connection.
SCM_514 Unable to find connection object with the connection identifier in *connectionObjects* view.	CNOM cannot remove the connection because the connection object is missing from the *connectionObjects* view. Possible reasons: • Connection name is invalid. • Connection was already deleted in CNOM.	Check for the connection object in the *connectionObjects* view. The connection object has the same name as the value of the work order attribute `workOrder:DDMWorkOrder.connectionIdentifier`.
SCM_515 Unable to delete connection object from *connectionObject* view.	CNOM cannot delete the connection object from the *connectionObjects* view. Possible reasons: • Since the work order was started, the connection has been deleted. • Internal error. A possible controller is in the process of restarting.	Make sure the connection object exists in the *connectionObjects* view. Wait for a period and retry the deletion.
SCM_516 Unable to delete `C6400DMMService` object from *connectionObjects* view.	CNOM cannot delete the `DMMService` parent object in the *connectionObjects* view. Possible reasons: • Since the work order was started, the connection has been deleted. • Internal error. A controller might be in the process of restarting.	Make sure the SCM NOA service object exists in the *connectionObjects* view. Wait for a period and retry the deletion.
SCM_517 Unable to disconnect subscriber from service.	CNOM cannot disconnect an existing cross-connect between a subscriber and a service. Possible reasons: • The connection was deleted in the node. • Unable to Telnet to the node due to a network problem or invalid passwords.	Check for more details in the work order status attributes `workOrder:DDMWorkOrder.primitiveErrorMessage` and `workOrder:DDMWorkOrder.connectActionReport`.
SCM_518 Unable to find the CLEC port from the containment path.	The given containment path for the port is not valid and cannot be changed into an object ID for setting IMT configuration.	Make sure the containment path is valid. Use the utility `ObjectUtilsTestRig` to verify that the containment path is valid by running `ObjectUtilsTestRig getObjectsWithinPath <`*`containment-path`*`>` in the CEMF shell. The utility returns a valid object ID for all containment objects in the path.

Table D-1 SCM Error Codes, Possible Causes, and Corrective Action (continued)

Error Code	Possible Cause	Corrective Action
SCM_519 Invalid CLEC Port. Unable to get Cisco 6400 UAC chassis for CLEC port	CNOM cannot identify the Cisco 6400 UAC chassis object for the CLEC port. To determine the port's chassis object, CNOM traces the ancestor of the port (third-level ancestor) in the SCM *Cisco6400Service* view.	Make sure this is a valid SCM line port.
SCM_520 Unable to find RT IMT port from the containment path.	The given containment path for the port is not valid and cannot be changed into an object ID for setting the IMT configuration.	To make sure the containment path is valid, run `ObjectUtilsTestRig getObjectsWithinPath` *`<containment-path>`* in a CEMF shell. The utility returns a valid object ID for all containment objects in the path.
SCM_521 Unable to set IMT configuration.	CNOM tried to set the attributes `LocalDB:DDMTopology.connectedTP` and `LocalDB:C6400DMM.clecChassisID` for an RT port, or it tried to set the attribute `LocalDB:DDMTopology.connectedTP` for a CLEC port and was unable to do so. Possible reasons: • Invalid attribute values • Invalid attribute names • Invalid policy	Check the *C6400DMMCtrller.log* file for more details.
SCM_522 Unable to find the ingress port to initiate connection.	The ingress port for the connection cannot be identified. Possible reasons: • `workOrder:DDMWorkOrder.ingressIdentifier` or `workOrder:DDMWorkOrder.ingress ContainmentPath` is not set in work order • Invalid line ID • Invalid containment path	Use `get_object_for_line` *`<line-id>`* to make sure the given line ID is valid. To make sure the containment path is valid, run `ObjectUtilsTestRig getObjectsWithinPath` *`<containment-path>`* in a CEMF shell. The utility returns a valid object ID for all containment objects in the path.
SCM_523 Unable to identify valid services for connection.	For a given ingress side (identified from a line ID associated with Cisco 6400 UAC port), the node does not detect a single service instance identified by `workOrder:DDMWorkOrder.egressIdentifier`.	Make sure services are available to support this connection for the given ingress port. Check the *C6400DMMCtrller.log* file for more details.
	SCM NOA could not setup a connection from the ingress port to the service instances identified by `workOrder:DDMWorkOrder.egressIdentifier`.	

Table D-1 SCM Error Codes, Possible Causes, and Corrective Action (continued)

Error Code	Possible Cause	Corrective Action
SCM_524 Unable to connect subscriber to service.	CNOM cannot connect an ingress VPI/VCI pair to a given service.	Check the work order attributes `workOrder:DDMWorkOrder.primitiveErrorMessage` and `workOrder:DDMWorkOrder.connectActionReport` for more details. The `primitiveErrorMessage` attribute contains more specific error messages that are encountered by the `ConnectSubscriber` primitive when it tries to deploy a subscriber object and connects the subscriber to the given ATM service. The `connectActionReport` attribute contains the details of the Cisco IOS commands that were sent to the node and possible errors in the Cisco IOS connection.
SCM_525 Unable to find a connection object with the given connection identifier.	CNOM cannot remove the connection because the connection object is missing from the *connectionObjects* view. The connection object has the same name as the value of the attribute `workOrder:DDMWorkOrder.connectionIdentifier`. Possible reasons: • Invalid connection name. • Connection was deleted in CNOM.	Make sure the connection identifier is valid. Check for the connection object in the *connectionObjects* view.
SCM_526 Unable to disconnect subscriber from service.	CNOM cannot disconnect an existing cross-connect between a subscriber and a service. Possible reasons: • The connection was deleted in the node. • Unable to Telnet to the node either due to a network problem or invalid passwords.	Check for more details in the attributes `workOrder:DDMWorkOrder.primitiveErrorMessage` and `workOrder:DDMWorkOrder.connectActionReport` in the work order.
SCM_527 Unable to delete connection object from *connectionObject* view.	CNOM cannot delete the connection object from the *connectionObjects* view. Possible reasons: • Since the work order was started, the connection has been deleted. • Internal error; a controller is in the process of restarting.	Make sure the connection object exists in the *connectionObjects* view. Wait for a period and retry the deletion.
SCM_528 Unable to create connection object in *connection* view.	CNOM cannot create a connection object with the name specified in the work order attribute `workOrder:DDMWorkOrder.connectionIdentifier` in the *connectionObjects* view.	Make sure that another connection is not already using this name. Connection identifiers must be unique and cannot be reused unless a previous connection with that ID has already been deleted.

Table D-1 SCM Error Codes, Possible Causes, and Corrective Action (continued)

Error Code	Possible Cause	Corrective Action
SCM_529 Unable to find ingress port or egress services to initiate connection.	For a given ingress side (identified either from a line ID associated with a Cisco 6400 UAC port or from a `connectedTP` attribute of a DSLAM port), the node could not find a single service instance identified by `workOrder:DDMWorkOrder.egressIdentifier`.	Make sure services are available to support this connection for the given ingress port. Check the *C6400DMMCtrller.log* file for more details.
	DMM could not set up a connection from the ingress port to the service instances identified by `workOrder:DDMWorkOrder.egressIdentifier`.	
scm_553 Identified ingress port does not belong to C6400DMM subdomain.	The line ID was not created in the Cisco 6400 subdomain.	Use `get_object_for_line` *`<line-id>`* to make sure the given line ID is valid. Issue the command `set_line_ids` to re-create the line ID, and be sure to specify the keyword `C6400DMM` to create the line ID in the Cisco 6400 subdomain.

APPENDIX E

Cisco Network Order Manager Reference

This appendix provides an overview of the commands associated with CNOM, the policies each command invokes, and a description of command usage. See Table E-1.

Table E-1 Cisco Network Order Manager Command Reference

Script	Policy Invoked	Action
SCM Commands and Scripts		
`connect_atm`	`connectSubscriberToATMService`	Create ATM connections on Cisco 6400 UACs, CEMF, and CNOM.
`connect_scm`	`connectSubscriberSCM`	Create other types of connections on Cisco 6400 UACs, CEMF, and CNOM.
`disconnect_atm`	`disconnectSubscriberAndService`	Disconnect ATM connection.
`getConnectionData`	`getATMConnectionData`	Retrieve ATM connection information.
Create Profile Commands and Scripts		
`create_B_profile`	`createBridgingServiceProfile`	Create a bridging service profile.
`create_IRB_profile`	`createIRBServiceProfile`	Create an IRB service profile.
`create_L2TP_profile`	`createL2TPServiceProfile`	Create an L2TP service profile.
`create_PPPoA_profile`	`createPPPoAServiceProfile`	Create a PPP over ATM service profile.
`create_R_profile`	`createRFC1483ServiceProfile`	Create an RFC 1483 IRB profile.
Create Service Commands and Scripts		
`create_B_service`	`createBridgingServiceFromProfile`	Create a bridging service.
`create_IRB_service`	`createIRBServiceFromProfile`	Create an IRB service from a profile.
`create_L2TP_service`	`createL2TPServiceFromProfile`	Create an L2TP service from a profile.
`create_PPPoA_service`	`createPPPoAServiceFromProfile`	Create a PPP over ATM service from a profile.
`create_R_service`	`createRFC1483ServiceFromProfile`	Create an RFC 1483 service from a profile.
Modify Profile Commands and Scripts		
`modify_B_profile`	`modifyBridgingServiceProfile`	Modify a bridging service profile.
`modify_IRB_profile`	`modifyIRBServiceProfile`	Modify an IRB service profile.
`modify_L2TP_profile`	`modifyL2TPServiceProfile`	Modify an L2TP service profile.
`modify_PPPoA_profile`	`modifyPPPoAServiceProfile`	Modify a PPP over ATM service profile.
`modify_R_profile`	`modifyRFC1483ServiceProfile`	Modify an RFC 1483 service profile.

Table E-1 Cisco Network Order Manager Command Reference (continued)

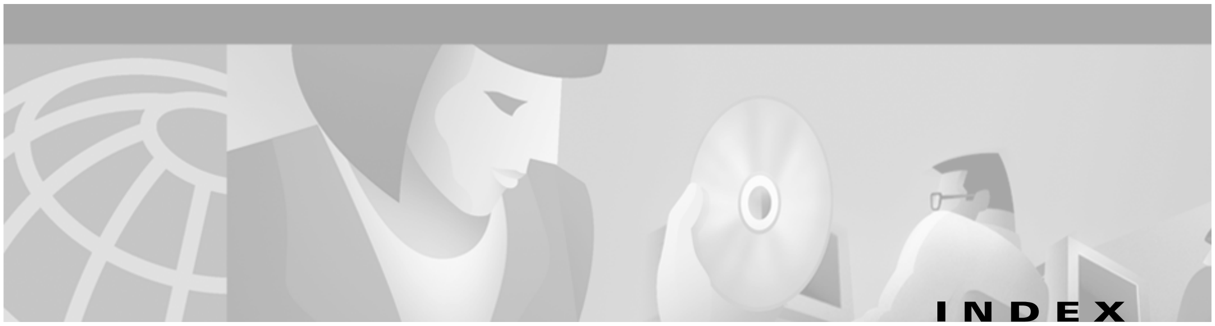
Script	Policy Invoked	Action
Modify Service Commands and Scripts		
modify_B_service	modifyBridgingServiceFromProfile	Apply profile to change a bridging service instance.
modify_IRB_service	modifyIRBServiceFromProfile	Apply profile to change an IRB service instance.
modify_L2TP_service	modifyL2TPServiceFromProfile	Apply profile to change an L2TP service instance.
modify_PPPoA_service	modifyPPPoAServiceFromProfile	Apply profile to change a PPP over ATM service instance.
modify_R_service	modifyRFC1483ServiceFromProfile	Apply profile to change an RFC 1483 service instance.
Delete Profile Commands and Scripts		
delete_B_profile	deleteBridgingServiceProfile	Delete a bridging service profile.
delete_IRB_profile	deleteIRBServiceProfile	Delete an IRB service profile.
delete_L2TP_profile	deleteL2TPServiceProfile	Delete an L2TP service profile.
delete_PPPoA_profile	deletePPPoAServiceProfile	Delete a PPP over ATM service profile.
delete_R_profile	deleteRFC1483ServiceProfile	Delete an RFC 1483 service profile.
Delete Service Commands and Scripts		
delete_B_service	deleteBridgingService	Delete the specified bridging service.
delete_IRB_service	deleteIRBService	Delete the specified IRB service.
delete_L2TP_service	deleteL2TPService	Delete the specified L2TP service.
delete_PPPoA_service	deletePPPoAService	Delete the specified PPP over ATM service.
delete_R_service	deleteRFC1483Service	Delete the specified RFC 1483 service.
Delete Template Commands and Scripts		
create_qos	createConnectionTemplate	Create a QoS connection template.
delete_qos	deleteConnectionTemplate	Delete the specified QoS connection template.
modify_qos	modifyConnectionTemplate	Modify a QoS connection template.
IMT Commands and Scripts		
set-IMT	setupTieLink	Create inter-machine tie (IMT) link.
unset-IMT	unsetupTieLink	Remove IMT link.
rename-IMT	renameTieLink	Rename IMT link and set new IMT name to all relative connections.
setactive-IMT	setActiveTieLink	Select which IMT link that you want to make active between two chassis.
set-IMTcount	setTieLinkCounter	Set the connections counter for how many PVCs in this IMT link.
CDM Commands and Scripts		
clear_cdm	clearCDMConnection	Clear a CDM connection in CEMF and CNOM. Nodes remain unaltered.
connect_cdm	connectSubscriber_cdm	Create a subscriber PVC and apply a QoS for the PVC between the ingress port and the egress trunk port. CEMF and CNOM.

Table E-1 Cisco Network Order Manager Command Reference (continued)

Script	Policy Invoked	Action
create_ni2_profile	createNI2ATMQoSProfile, createNI2DMTProfile createNI2SDSLProfile createNI2CAPProfile	Create DSLAM interface configuration profiles for DMT, ADSL, SDSL, CAP, and ATM QoS in the CDM system.
delete_ni2_profile	deleteNI2ATMQoSProfile deleteNI2DMTProfile deleteNI2SDSLProfile deleteNI2CAPProfile	Remove the NI-2 profile from the CDM system.
disconnect_cdm	disconnectSubscriber_cdm	Disconnect a CDM connection and remove it from the DSLAM node, CEMF, and CNOM.
get_line_id	—	Provide a means of determining which line ID is set to a specific ingress port.
get_object_for_line	—	Provide the containment path for the ingress port of a line ID.
modify_ni2_profile	modifyNI2ATMQoSProfile modifyNI2DMTProfile modifyNI2SDSLProfile modifyNI2CAPProfile	Modify an NI-2 ATM QoS profile.
set_line_ids	—	Give a line ID name to the ingress port of the DSLAM, which is used by the work order system to execute other scripts.
CWM Commands and Scripts		
connect_cwm	connectSubscriber_cwm	Create a WAN connection on the CWM machine, node (through CWM) and CNOM.
connect_with_CWM	connectSubscriber_with_CWM	—
create_cwm_qos	create_cwm_QoS	Create a WAN QoS profile.
disconnect_cwm	disconnectSubscriber_cwm	Disconnect a WAN connection on the CWM machine node (by using CWM) and CNOM.
disconnect_with_CWM	disconnectSubscriber_with_CWM	—
set_wan_port	set_WAN_port_attributes	Set WAN node, shelf, slot, and port attributes.
remove_cwm_qos	remove_cwm_QoS	Delete a WAN QoS profile.
General Commands and Scripts		
add_line_id	add_map_value	Add a line ID for a given full object path.
add_map_value	addMappingValue	—
auditLogFile	auditLogFile	Write a comment to a recovery log file.
cleanup_line_id	removeAnyObject	Remove line IDs that were not deleted before you removed a chassis or uninstalled CNOM.
get_line_id	—	Get an attribute for a given object.
get_object_for_line	Script get_object_for_value	Get a path for a line ID value.
get_object_for_value	findMappedTarget	Get a line ID for a path.
getRestoreWO	—	Restore the CNOM configuration by replaying a recovery log file.
initSequence	InitSequence	—

Table E-1 Cisco Network Order Manager Command Reference (continued)

Script	Policy Invoked	Action
map_values	setMappingValues	—
remove_line_id	remove_map_value	Remove a line ID for a given path and line ID.
remove_map_value	removeMappingValue	—
reset_line_ids	reset_map_values	Reset a line ID.
reset_map_values	resetMappingValues	—
set_line_ids	map_values	Set a line ID for a given path.
socketDaSet	Invokes socketDaSet	Set attribute(s) for a given object.



INDEX