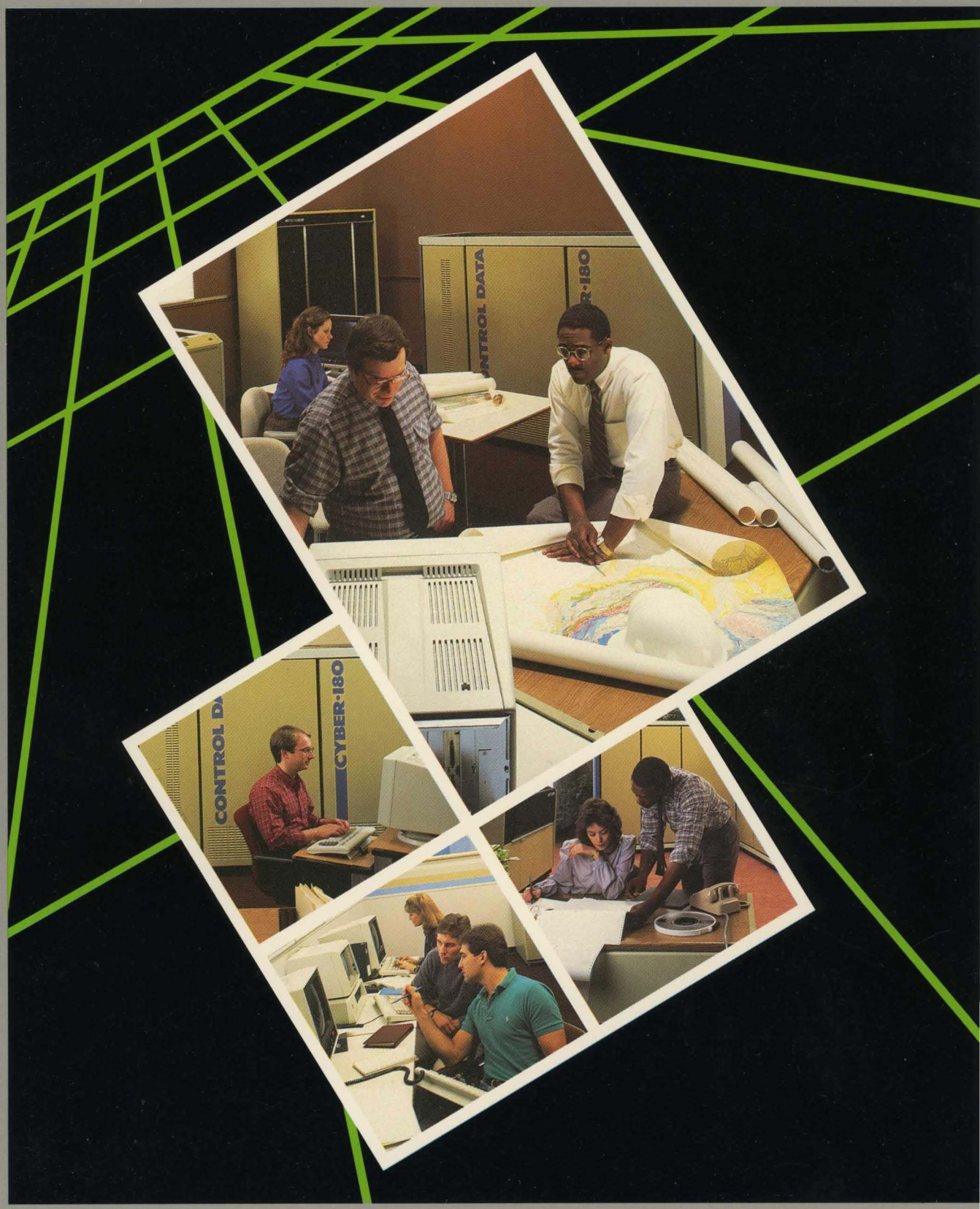
NOS/VE
Network Management

CONTROL DATA



Usage

60463916

NOS/VE Network Management

Usage

This product is intended for use only as described in this document. Control Data cannot be responsible for the proper functioning of undescribed features and parameters.

Publication Number 60463916

Manual History

Revision	System Version	PSR Level	Date
A	1.2.2	678	April 1987
B	1.2.3	688	September 1987
C	1.3.1	700	April 1988
D	1.4.1	716	December 1988

Revision D of this manual, printed in December 1988, reflects the release of NOS/VE Version 1.4.1 at PSR level 716.

Changes to the manual for this release include the following:

- Added new application installation commands to define, delete, activate, and deactivate the Simple Mail Transfer Protocol/VE and the Network File System/VE applications.
- Added new MANAGE_APPLICATION_DEFINITIONS utility. This utility contains the application installation commands that define and delete applications.
- Replaced references to the UPGRADE_SOFTWARE utility in chapter 5 with references to the MANAGE_APPLICATION_DEFINITIONS utility.
- Added new parameters to the CHANGE_NAM_ATTRIBUTES command:
  - ADDITIONAL_LOGIN_PROMPTS
  - MAXIMUM_LOGIN_ATTEMPTS
  - PREFERRED_PROTOCOL_STACK
- Removed old parameters from the CHANGE_NAM_ATTRIBUTES command:
  - ICA_ERROR_RESET_THRESHOLD
  - MCI_ERROR_RESET_THRESHOLD
- Added new chapter 8 to describe the MANAGE_STORE_FORWARD_NETWORK utility.
- Added new appendix D to describe the error messages associated with the MANAGE_STORE_FORWARD_NETWORK utility.

In addition, miscellaneous editorial and technical changes and corrections are made.

Contents

Figures

Tables

About This Manual

This manual explains how to create and manage network application definitions, and manage network services on the CONTROL DATA® Network Operating System/Virtual Environment (NOS/VE) using the CDC® Control Data Distributed Communications Network (CDCNET).

Audience

This manual is for the system analyst who is responsible for network management on NOS/VE. It is assumed that you are either an experienced NOS/VE analyst or you have completed the sequence of NOS/VE hardware and software courses (including the NOS/VE Analysis course) and the CDCNET courses offered by Control Data. It is also assumed that you have a working knowledge of the System Command Language (SCL) as described in the NOS/VE System Usage manual.

Conventions

This manual documents commands and subcommands using a format similar to the format used in the NOS/VE Commands and Functions manual. In addition, the following conventions are used:

Boldface	In command format descriptions, command names, subcommand names, and required parameters are shown in boldface type.
Italics	Optional parameters in a command or subcommand format description are shown in italics.
Vertical bar	A technical change is indicated by a vertical bar in the margin next to the change.
Numbers	All numbers are decimal unless otherwise noted.

Submitting Comments

There is a comment sheet at the back of this manual. You can use it to give us your opinion of the manual's usability, to suggest specific improvements, and to report errors. Mail your comments to:

Control Data Corporation
Technology and Publications Division ARH219
4201 North Lexington Avenue
St. Paul, Minnesota 55126-6198

Please indicate whether you would like a response.

If you have access to SOLVER, the Control Data online facility for reporting problems, you can use it to submit comments about the manual. When entering your comments, use NV0 (zero) as the product identifier. Include the name and publication number of the manual.

If you have questions about the packaging and/or distribution of a printed manual, write to:

Control Data Corporation
Literature and Distribution Services
308 North Dale Street
St. Paul, Minnesota 55103

Or call (612) 292-2101. If you are a Control Data employee, call (612) 292-2100.

CYBER Software Support Hotline

Control Data's CYBER Software Support maintains a hotline to assist you if you have trouble using our products. If you need help not provided in the documentation, or find the product does not perform as described, call us at one of the following numbers. A support analyst will work with you.

From the USA and Canada: (800) 345-9903

From other countries: (612) 851-4131

Introduction to Network Management 1

Introduction to Network Management 1

NOS/VE network management, as described in this manual, refers to the performance of network-related activities on the NOS/VE operating system. NOS/VE network management functions include:

- Maintenance of network applications, as described in chapters 2 through 5.
- Activation and deactivation of certain host-based parts of the CDCNET network software, as described in chapter 6.
- Site-tailoring of NAM/VE attributes, as described in chapter 7.
- Controlling the action of the Queue File Transfer Facility (QTF) in its store-and-forward function, as described in chapter 8.

Although NOS/VE network management responsibilities include maintenance of some host-based CDCNET software, they do not include CDCNET management as such. For information on CDCNET management and administration, refer to the appropriate CDCNET documentation.

Network Application Programs

NAM/VE is part of the NOS/VE operating system and provides the interface between the NOS/VE operating system and CDCNET. All network application programs run under NAM/VE.

Network applications provide various network services such as interactive and batch terminal support for NOS/VE. (Note that in a dual-state environment, interactive terminal support can also be provided through the NOS or NOS/BE partner system.) In a multihost network, network services can also be established to transfer jobs and files between mainframes in the network.

CDCNET Management Entities

CDCNET software includes a number of application programs that support CDCNET functions. These programs are called Management Entities (MEs).

The NOS/VE network manager is responsible for activating certain MEs that reside on the NOS/VE host. Chapter 6 describes these MEs and their commands.

NAM/VE Attributes

You can control aspects of NAM/VE operation that affect performance and the use of resources with the CHANGE_NAM_ATTRIBUTES and DISPLAY_NAM_ATTRIBUTES commands. Chapter 7 describes these commands in detail.

MANAGE_STORE_FORWARD_NETWORK Utility

A site administrator can use the MANAGE_STORE_FORWARD_NETWORK utility to create a store-and-forward network file. This network file contains mainframe name substitutions required to forward a queued file from one NOS/VE system to another NOS/VE system through a non-VE system. Chapter 8 describes the MANAGE_STORE_FORWARD_NETWORK utility in detail.

Understanding Network Applications 2

Understanding Network Applications 2

Introduction

This chapter explains network application concepts and describes the system-defined network applications that execute on NOS/VE.

The default application definition for each system-defined network application described in this chapter is listed in appendix C. The NOS/VE commands used for installing the system-defined network applications are listed in chapter 5. See chapter 3 for instructions on how to use the MANAGE_NETWORK_APPLICATIONS (MANNA) utility to change the default network application definitions.

NOTE

You should be familiar with the concepts and terms presented in this chapter before using the MANAGE_NETWORK_APPLICATIONS utility.

Network Application Concepts

Within a CDCNET configuration, various types of communications media enable you to physically interconnect NOS/VE and other mainframe hosts, terminals, peripheral devices, and CDCNET device interfaces (DIs) into networks. DIs facilitate communication among mainframes, terminals, and peripheral devices in a network. For more information about DIs, see the CDCNET manuals listed in appendix B.

A network application can be structured as a single program, a sequence of programs, or a set of concurrent programs. In this manual, each execution of an application is referred to as an instance of that application.

Within a multihost network, NAM/VE allows an application to communicate with another application on any mainframe host connected to the network. NAM/VE also provides application-to-application communication within the same NOS/VE system. Figure 2-1 illustrates both concepts. The following sections explain general network application concepts.

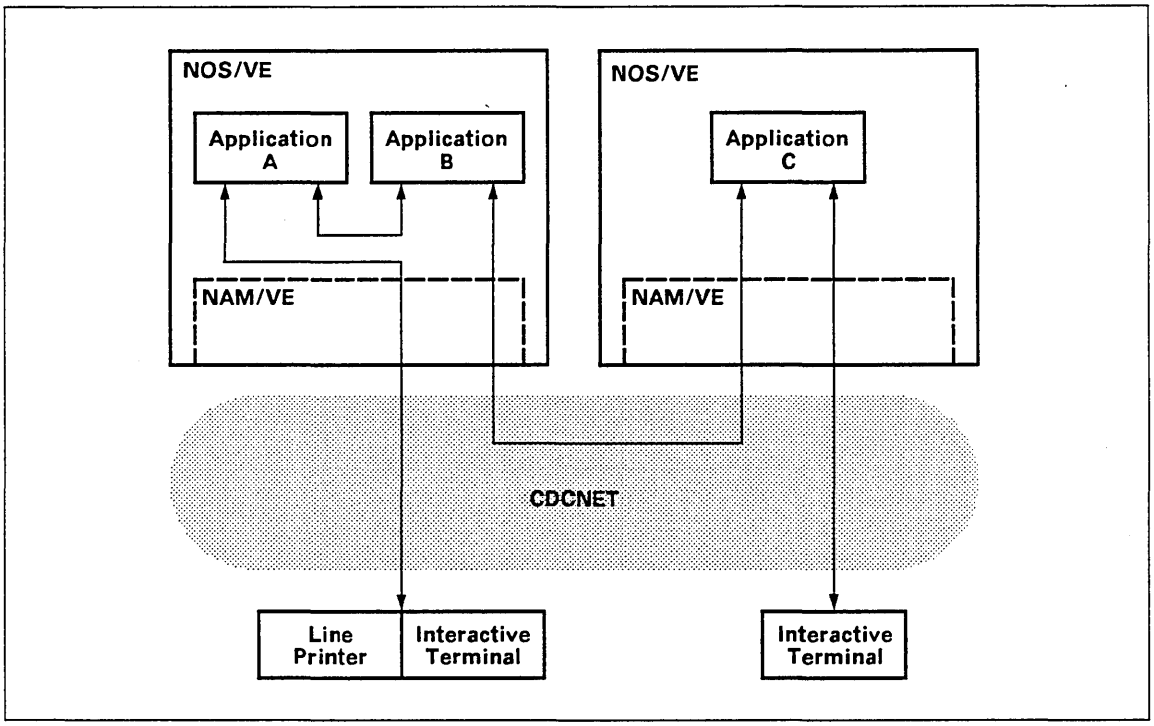


Figure 2-1. Network Connections

Client and Server Network Applications

A network application is classified as either a client application or a server application by the role the application plays in establishing a network connection. The client/server classification is relevant only to the creation of a network connection. Once the connection is successfully established, all applications have the same privileges. However, the applications may choose to retain their client or server characteristics and restrict operations accordingly. The client/server classification applies only to a particular network connection. An instance of an application that uses multiple connections may be the client for one connection and the server for a different connection.

A client application is any application authorized to create a network connection to access a service provided by a server application. The server application can either accept or reject a connection request. If the request is accepted, the server application provides a service to the client application via the network connection.

For example, when an interactive terminal user enters the CDCNET command:

```
create_connection nosve
```

a CDCNET client application in the DI connected to the user's terminal executes the command, thereby requesting a connection to the Timesharing server application with the registered title of NOSVE. An interactive job starts in the NOS/VE host. This job is an instance of the Timesharing server application.

The rest of this example illustrates the concept of an instance of an application being the server for one network connection and the client for another connection. Assume the interactive terminal user wants to use the Permanent File Transfer Facility (PTF) to transfer files to or from another host in the network. In this case, the interactive job (instance of the Timesharing application) requests a network connection to the PTF server application in the other host. The instance of the Timesharing application which is the server for the first connection is now the client for the second connection.

Single-Connection Applications

A single-connection application is an application that supports one connection for each instance of the application. Since several instances of an application can be active at one time, a single-connection server application can provide service to several clients simultaneously, as shown in figure 2-2.

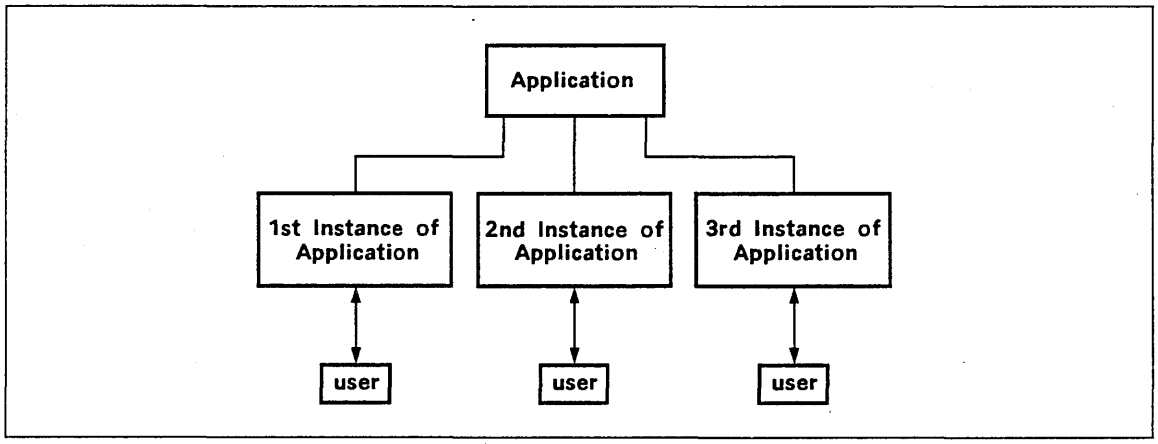


Figure 2-2. Single-Connection Application

Multiconnection Applications

A multiconnection application supports multiple connections for each instance of the application. In this case, there is usually only one instance active at any given time (see figure 2-3). A multiconnection application is used if connections to the application must interact (for example, to exchange messages between connections) or if operations must be synchronized between connections (for example, to update a data base).

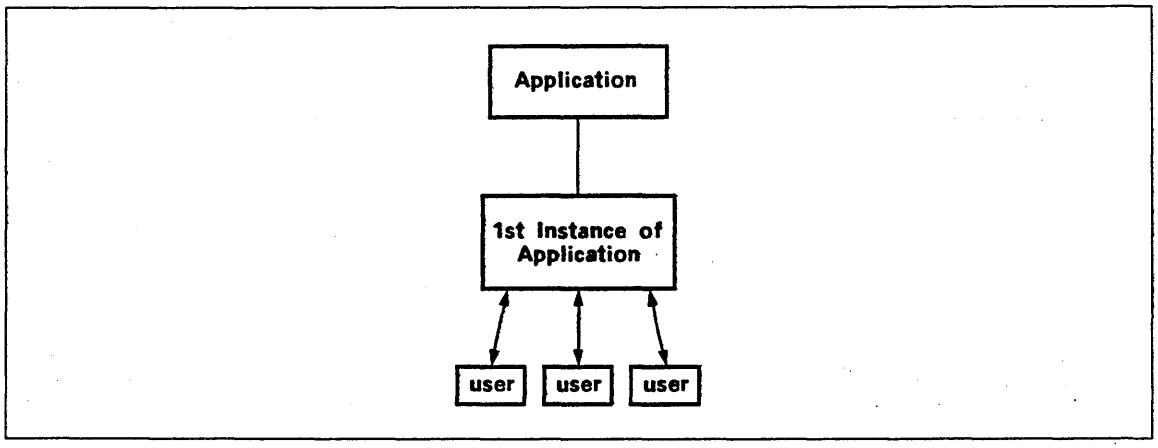


Figure 2-3. Multiconnection Application

Network Address Components

Each application that can access CDCNET has a network address. This address locates the application within the network. The address is used to route messages to the application. A network address consists of three parts:

- A network identifier
- A system identifier
- An application identifier

The network identifier and the system identifier uniquely identify the NOS/VE system within the CDCNET network. The network identifier is defined during network configuration. (For more information on defining a network configuration, refer to the NOS/VE System Performance and Maintenance manual, Volume 2.) The NOS/VE system identifier is created automatically by NAM/VE from the CPU model and serial numbers when NAM/VE is activated.

The application identifier indicates a specific application within the NOS/VE system. When you define client and server applications, you can specify either fixed application identifiers or you can allow NOS/VE to assign variable identifiers. Fixed and variable application identifiers are explained in chapter 3.

Server Application Job

A server application job is defined as the job in which an instance of a server application occurs. It is also the job that provides the server application's service to a client connection.

A server application must be activated before any client can establish a connection to the server. A server application is activated by the ACTIVATE_SERVER subcommand. See chapter 4 for a description of this MANNA subcommand.

For a server application to service a client connection, the following must be true:

- The network address at which the server application resides must be active (open) so a connection request can be received.
- An instance of the server application (a server application job) must be executing.

NAM/VE supports two types of server applications: NAM-initiated server and user-initiated server. The server application types are specified on the NAM_INITIATED parameter of the DEFINE_SERVER subcommand. This subcommand is described in chapter 4. These two types of server applications differ in the way in which the server job is initiated and the conditions under which the server's network address becomes active. The following two sections explain these differences.

User-Initiated Server Applications

For a user-initiated server, the server application job is initiated by a command for a specific application that is executed by an authorized user. The server job then notifies NAM/VE of its existence as an instance of a server application and waits for connection requests. When at least one instance of the server job is executing, the network address of a user-initiated server job is active.

Except for the Timesharing application, all of the system-defined applications are user-initiated. The activation commands that initiate server application jobs are described in chapter 5, Installation Commands.

NAM-Initiated Server Applications

For NAM-initiated server applications, the server's network address is active when the server's state is activated by an ACTIVATE_SERVER subcommand. When a NAM-initiated server is defined by a DEFINE_SERVER subcommand, a sequence of SCL commands is specified that initiates the server application job, once the job is submitted by NAM/VE to the system. (See the SERVER_JOB server application attribute described in table 3-4.)

NAM/VE waits for a connection request for the server application. Then, based on whether the server application is a single-connection or multiconnection application, NAM/VE submits a server job to the system.

For a single-connection server application, NAM/VE initiates a server job for each connection request. (The Timesharing application is an example.) For a multiconnection server application, NAM/VE initiates a server job only when the server jobs currently executing cannot service more connections.

The following table summarizes the major differences between user-initiated and NAM-initiated server jobs.

Table 2-1. User-Initiated and NAM-Initiated Server Job Comparison

User-Initiated Server Job	NAM-Inititated Server Job
Validation information is specified in a LOGIN command executed by the user.	Client can supply the validation information.
Server jobs are initiated by a command executed by the user.	NAM/VE initiates server jobs as needed to service clients.
The content within the server job is unknown to NAM/VE.	The content within the server job is specified to NAM/VE via the MANAGE_NETWORK_APPLICATIONS utility.
A server job must be initiated before a client's request can be serviced.	A client must initiate a connection before NAM/VE executes a server job to service the connection.
NAM/VE registers a server's titles in the network directory when the first instance of the server job is initiated.	NAM/VE registers a server's titles in the network directory when the server application is activated.

Network Application Validation

In a NOS/VE system, users are assigned passwords to access the system. Each user is uniquely identified by a family and user name, and is assigned certain privileges and capabilities.

In a like manner, a network application is uniquely identified by an application name and network address, and can also be assigned a set of privileges and capabilities.

NAM/VE provides two levels of application validation.

The first level of application validation is through the MANAGE_NETWORK_APPLICATIONS utility. You use this utility to verify that a program is an authorized network application and to allow the program to use the network address and other attributes of the application. The MANAGE_NETWORK_APPLICATIONS utility gives you the following client and server attributes that specify how the validation is performed.

Server Attributes:

SERVER_CAPABILITY
SERVER_RING
SERVER_SYSTEM_PRIVILEGE

Client Attributes:

CLIENT_CAPABILITY
CLIENT_RING
CLIENT_SYSTEM_PRIVILEGE

NOTE

The SERVER_SYSTEM_PRIVILEGE and CLIENT_SYSTEM_PRIVILEGE attributes are not operational for the current release of NOS/VE.

See chapter 3 for more information about these attributes.

The second level of application validation must take place for network applications to communicate through NAM/VE. This validation takes place in one of two ways:

- When a NAM-initiated server job is submitted to the NOS/VE system, the server application validates the job. This is called *server job validation.*
- When a client application requests a connection to a server application, the server application validates the connection request. This is called *connection request validation.*

The next two sections discuss this second level of application validation.

Server Job Validation

Like all NOS/VE jobs, the server job must validate itself to the system. The server job validation:

- Identifies the user for whom the server job is executed.
- Determines the activities to be performed within the job.
- Establishes the limits on resources the server job uses.

The validation information for server jobs is obtained from the LOGIN command. However, the source of information for the LOGIN command varies, depending on whether the server job is user-initiated or NAM-initiated.

For a user-initiated server job, you must specify validation information in a LOGIN command. The LOGIN command must also be the first command in the job file.

For a NAM-initiated server job, the user validation information is also contained in the LOGIN command. It too must be the first command in the job. However, the MANAGE_NETWORK_APPLICATIONS utility provides the following two options:

- Validation information can be imbedded in the server job definition.
- Validation information can be obtained by NAM/VE from the client prior to initiating the server job.

For more information on server job validation, see the description of the SERVER_JOB_VALIDATION_SOURCE attribute in table 3-4.

NOTE

If the validation source is the server job, the first command in the job must be a LOGIN command. If the validation source is the client, the server job cannot contain a LOGIN command.

Client-Supplied Validation

If the client application supplies the user validation information, NAM/VE initiates a dialog with the user as follows:

```
Enter validation for service access.
User:
Password:
Family:
```

NAM/VE concatenates the login responses to form a parameter list for a LOGIN command. This parameter list follows standard SCL syntax rules. If the client does not supply a family name, NAM/VE uses the system default family. If the client specifies more than one parameter in a single response, NAM/VE assumes that all required parameters are entered and stops prompting.

In addition to supplying the user name, password, and family name, the server application allows the user to enter other information during the login dialog (for example, user job name, job class, account, project, and job execution ring). When all responses are entered, NAM/VE creates a LOGIN command to be included as the first command in the server job it submits to NOS/VE. Next, NAM/VE validates the user's

identity based on the LOGIN command parameters, and the server job runs with the user's identity. The user specified by the client application is billed directly for resources used by the server job, and the user must be validated for all server job activities. In addition, the user's user prolog is executed in the server job.

If NAM/VE detects errors in the LOGIN parameter list, it issues an error message and restarts the dialog. After three unsuccessful attempts, NAM/VE issues an error message and rejects the connection request.

Connection Request Validation

A server application validates a client application's authority to connect to services provided by the server. However, for convenience, NAM/VE provides two optional forms of connection request validation: client validation and client address validation. The application developer can use either of these validation capabilities, or the developer can write his or her own validation code within the program. In addition to using the NAM/VE validations, a programmer can write supplementary validation requirements.

NAM/VE Client Validation

When NAM/VE validates clients, NAM/VE accepts the client's connection request on behalf of the server application. NAM/VE initiates an interactive dialog to obtain the client's identity.

The dialog is the same as the server job validation dialog. However, in client validation dialog, only the USER, PASSWORD, and FAMILY parameters are valid. All other parameters are ignored.

Once the client's identity is validated, NAM/VE checks the client's entry in the NOS/VE validation file to verify that the client is authorized to access the requested server application. If the connection request is validated, NAM/VE assigns the connection to the server application, and the application begins processing. If the connection request is not validated, NAM/VE rejects the connection.

NOTE

For server job and client validation, the NOS/VE family validation file serves as the validation data base. The ADMINISTER_VALIDATIONS utility is used to manage this data base. The NOS/VE validation file is maintained by the family administrator. For more information, see the NOS/VE User Validation manual listed in appendix B.

NAM/VE Client Address Validation

NAM/VE can validate a connection request using the client application's network address. When you define a server application, you can specify a list of network addresses authorized to connect to the server application with the the CLIENT_ADDRESS server application attribute described in table 3-4. To validate a connection request, NAM/VE obtains the client application's network address and verifies it against the list of authorized network addresses. If the client application is at a network address in the list, NAM/VE assigns the connection to the server application and the server application begins processing. If the client application is at a network address that is not in the list, NAM/VE rejects the connection and the server application receives no indication of the attempted connection.

Network Directory

A client application specifies a server application's network address when initiating a connection request. However, you may choose to have a server's network address vary with each activation of that server. To allow the client to locate the server, NAM/VE provides a network directory that maps one or more logical titles to the server's network address. The client need only specify a server's title to obtain the appropriate network address. The following discussion explains how you can assign and create server titles and how the network performs directory searches.

Assigning Server Titles

Assigning titles to server applications serves two purposes:

- It allows the client to identify a server by a logical name that represents the service provided.
- It eliminates any need for the client to know where a server application executes. A server may be moved from one network location to another without requiring clients to change the server's identity.

When two or more server applications register identical titles, NAM/VE assumes the applications provide identical services. Hence, when a client application translates a title, the network address returned may be the address of any server for which that title is registered. Therefore, the title you assign must uniquely identify the service provided. Neither the system nor the MANNA utility enforces this constraint. The network administrator must ensure that all server applications with the same title provide the same service.

NOTE

Before determining the network address of the server application, the CDCNET command CREATE_CONNECTION converts all letters in the specified server application title to uppercase. Only those characters allowed by SCL in name values are allowed. Therefore, when you assign a title to a server application that accepts connections from CDCNET terminals, specify the titles using only uppercase letters and other characters allowed in SCL names.

Title Management

The MANAGE_NETWORK_APPLICATIONS utility allows you to manage server titles directly or to delegate title management to the server applications. In either case, you may create multiple titles for a server application.

A title is a string of from one to 255 ASCII characters, including nongraphic characters. Uppercase and lowercase letters are significant. For example, the titles TIMESHARING and timesharing are not equivalent.

You can directly specify titles by using the ADD_TITLE and DELETE_TITLE MANNA subcommands. Titles managed by the MANAGE_NETWORK_APPLICATIONS utility remain in effect until you explicitly delete them. The MANNA utility also allows you to delegate title management responsibilities to a server application. A server application manages its titles by means of program interface requests to add and delete titles. You must specify the set of titles the server can manage by indicating one or more title patterns on the ADD_SERVER_MANAGED_TITLE subcommand.

A title pattern defines a set of titles by specifying a substring of characters common to each title in the set. For example, the title pattern:

```
APP*
```

is a title pattern that defines the set of all titles beginning with the characters APP. The asterisk (*) is a substitute (or wild card) character that matches any string of characters. Table 2-2 describes the various substitution characters available to create title patterns.

Table 2-2. Title Pattern Character Descriptions

Character	Description
?	Matches any single character of a title.
*	Matches any string of characters in a title, including the null string.
'c'	Matches the character enclosed in apostrophes. This character may be any character other than the apostrophe.
''	Matches the apostrophe character in a title.
[...]	Matches any single character from a group of characters in a title. The group of characters is specified within the brackets, either as a list ([abc]), a range ([a-z]), or a combination ([a-zA-Z$#1-9]). The expression 'c' may appear within the brackets to represent the character enclosed in apostrophes. Special characters (? * ') do not have special meaning within the brackets. They represent themselves.

The following table lists some example title patterns. For each example pattern, the table gives examples of titles that can be matched and titles that cannot be matched.

Title Pattern	Titles Matched	Titles Not Matched
abc	abc	Abc
a?a	aza	aa
	ala	abab
az*az	azaz	az
	azlaz	azlz
	azXXXaz	azXXXazYYY
a'*'*	a*22	aBC
	a*Z	a'*'Z
	a*	
a"b	a'b	ab
		a"b
a[0123]	a2	ab
		a20
a[a-z*]	aa	aA
	a*	a$
	ab	
a[a-zA-Z]	an	a1
	aJ	

NOTE

Titles managed by a server application are implicitly deleted whenever the server application's network address becomes inactive or a NOS/VE deadstart occurs.

Title Registration in the Network Directory

Titles that are registered in the directory are known throughout the network.

NAM/VE registers server titles in the network directory as follows:

- For user-initiated server applications, NAM/VE registers the server's titles in the directory when the first server job is initiated (using the command or procedure defined by the application itself). The server's titles are deleted when all instances of the server application terminate.
- For NAM-initiated server applications, NAM/VE registers the server's titles in the directory when the server application is activated (using the ACTIVATE_SERVER subcommand).

When requesting a connection to the Timesharing application by using the CDCNET command CREATE_CONNECTION, you specify a title rather than a network address. For example:

```
create_connection nosve002
```

requests a connection to the application that is assigned the title NOSVE002.

Directory Searches

A client application obtains translations of titles registered in the directory by searching the directory. This search involves scanning the directory for titles that match a specified title pattern. A title translation for a title is delivered to the client application if it meets two criteria:

- The title matches the specified title pattern.
- The title was registered by a server application that used the same type of connection as the client application that requested the search. Connection type refers to the protocol used to provide a connection; the CDNA_SESSION protocol is used for host-to-host connections, and the CDNA_VIRTUAL_TERMINAL protocol is used for CDCNET-connected terminals. For more information on the PROTOCOL attribute, see table 3-3.

There are two types of directory searches: nonrecurrent and recurrent. A nonrecurrent search scans the directory only once. A recurrent search scans the directory continuously as long as the client application is active.

Title translations for matching server titles that are subsequently registered in the directory are returned to the client application. The directory returns new server titles to the client application as long as the client remains active. This ensures that active client applications are informed of new servers in the network.

To make a recurrent search possible, you broadcast the title translation to all remote systems when the title is first registered. You do this by using the BROADCAST_REGISTRATION attribute described in table 3-4.

Title Priority

You can specify title priority to indicate a preferred server application among any number of applications that have the same title. Title priority is significant whenever a client application connection request specifies a title that is common to multiple server applications. If different priorities are assigned for the available server applications, the client is connected to the server with the highest priority. If two or more server applications have the same title with the same priority, the translations of the titles are delivered in random order. You use the TITLE_PRIORITY attribute (described in table 3-4) to specify title priority when you define a server application.

NOS/VE System-Defined Network Applications

The system-defined NOS/VE network applications provide connections to CDCNET to support the following services:

- Timesharing
- Desktop Environment (Desktop/VE)
- Permanent File Transfer Facility/Queue File Transfer Facility (PTF/QTF)
- Batch Transfer Facility (BTF)
- Network Transfer Facility (NTF)
- File Transfer Protocol/VE (FTP/VE)
- Simple Mail Transfer Protocol/VE (SMTP/VE)
- Network File System/VE (NFS/VE)

Timesharing

The Timesharing network service provides interactive access to a NOS/VE system. (This service allows interactive terminals to log in to NOS/VE.) On dual-state systems, the Timesharing service is optional, since interactive terminals may also be connected through the dual-state partner system. The Timesharing service is provided by a single network application program. Figure 2-4 illustrates how the Timesharing service provides host-to-terminal connections in the CDCNET network. For more information about this network service, refer to the CDCNET Access Guide online manual (CDCNET_ACCESS).

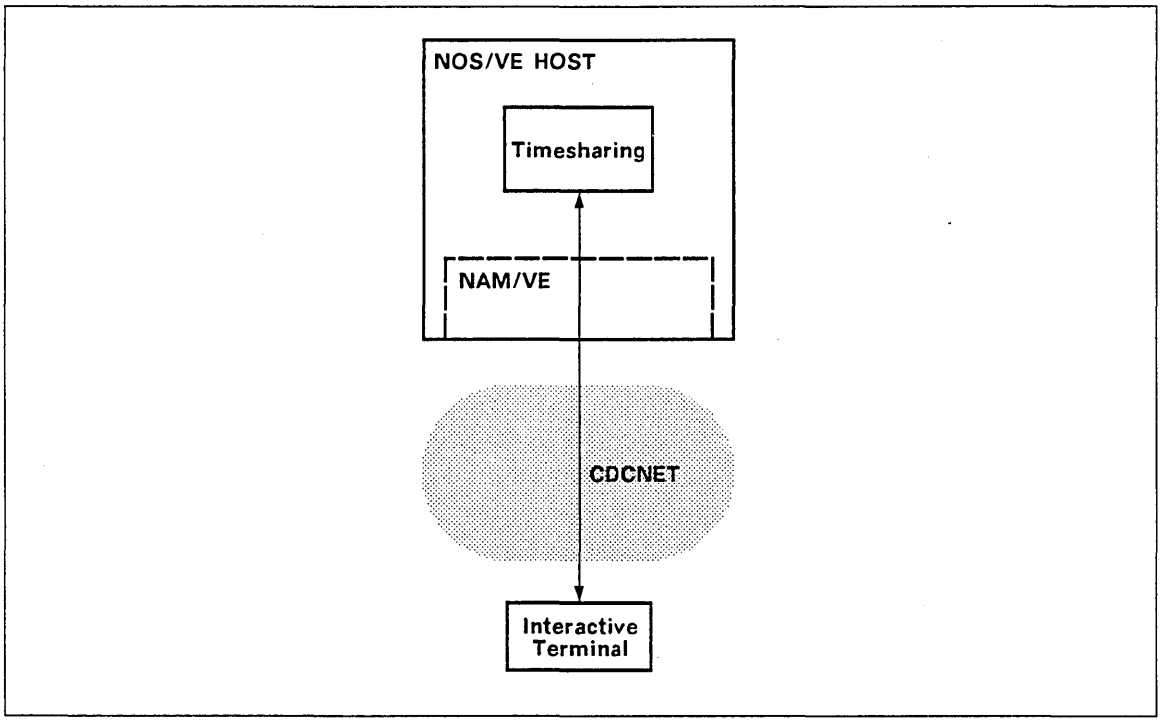


Figure 2-4. Timesharing Host-to-Terminal Connections

Desktop Environment (Desktop/VE)

The Desktop Environment network service provides a secondary connection between a workstation and a NOS/VE system. The primary connection is the Timesharing service. The Desktop/VE service is provided by a single network application program. Each time the Desktop/VE service is started, a new instance of the Desktop/VE application is initiated within the user's job. This secondary connection carries data defined by an internal Desktop/VE protocol. For more information about this network service, refer to the CDCNET Access Guide online manual (CDCNET_ACCESS).

Figure 2-5 shows how the Desktop/VE service provides the host-to-workstation secondary connections in the CDCNET network.

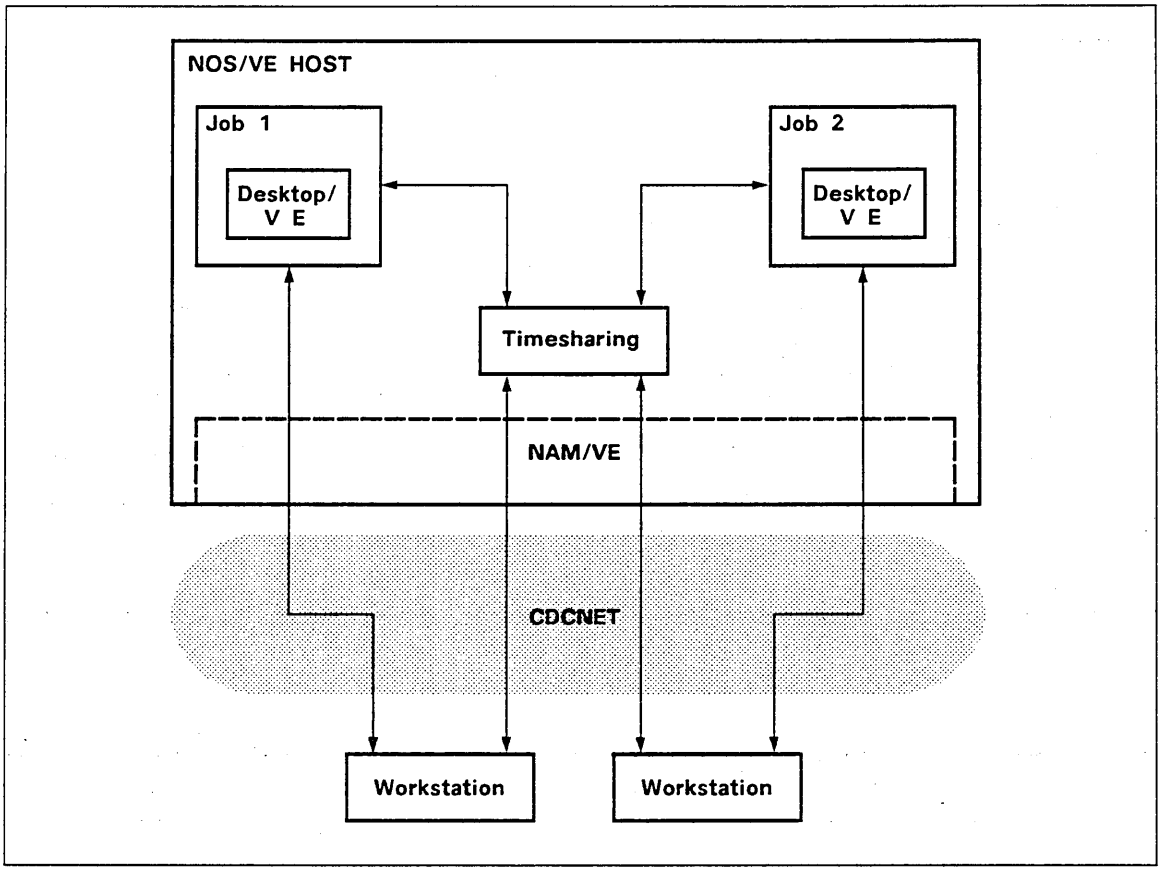


Figure 2-5. Desktop/VE Host-to-Workstation Connections

Permanent File Transfer Facility/Queue File Transfer Facility (PTF/QTF)

The Permanent File Transfer Facility/Queue File Transfer Facility (PTF/QTF) services allow files to be transferred between NOS/VE systems or between NOS/VE and NOS in a CDCNET network. The PTF/QTF services also allow file transfers between a NOS/VE system and an AOS/VS system.

The PTF service and the QTF service each provide two separate applications: a client application and a server application. The client/server relationship is described earlier in this chapter under Network Application Concepts.

The PTF service allows the transfer of temporary and permanent files between NOS/VE systems or between NOS/VE and NOS or AOS/VS systems. The commands that initiate a file transfer are described in the NOS/VE System Usage manual.

The QTF service gives you a means to transfer queued files (both jobs and listings) between NOS/VE systems or between NOS/VE and NOS or AOS/VS systems. The SUBMIT_JOB and PRINT_FILE commands are used to transfer jobs and print files. These commands are described in the NOS/VE System Usage manual.

A site administrator can use the MANAGE_STORE_FORWARD_NETWORK utility to create a store-and-forward network file. This file is used to assist QTF in its store-and-forward function. See chapter 8 of this manual for details.

Figure 2-6 shows PTF and QTF host-to-host connections.

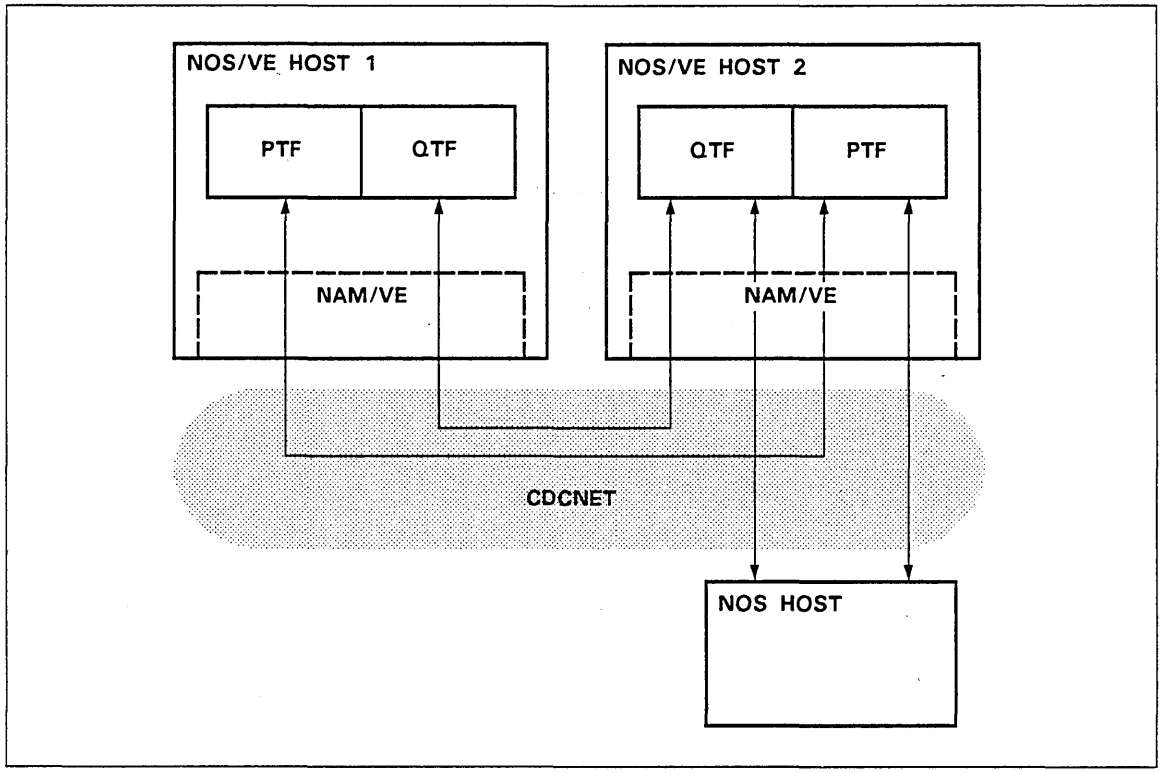


Figure 2-6. PTF and QTF Host-to-Host Connections

Batch Transfer Facility (BTF)

The Batch Transfer Facility (BTF) provides both input and output batch services. The BTF services include several client and server applications. Together, these applications give you batch device support to input jobs and print files in the CDCNET network. (Batch support for NOS/VE is provided only through CDCNET.)

The BTF services consist of the following applications:

Status and Control Facility Server/VE (SCFS/VE)	Server application (called the I/O station control facility) that resides in the NOS/VE host computer. At least one SCFS/VE application must be present in each network. SCFS/VE communicates over connections with SCF/VE, SCF/DI, and OPES applications. It receives the status of batch output files and batch devices, schedules output files to batch devices, and routes I/O station control commands.
Status and Control Facility/VE (SCF/VE)	Client application that resides in every NOS/VE mainframe in the network that supports batch output files. It informs SCFS/VE of available output files and processes file control commands from SCFS/VE.
Status and Control Facility/DI (SCF/DI)	Client application that resides in every DI connected to batch devices. SCF/DI reports the status of batch devices to SCFS/VE and processes batch device control commands from SCFS/VE.
Batch Transfer Facility/VE (BTF/VE)	Client application that resides in every host computer in the network that supports batch output files. It communicates with BTFS/DI to transfer output files from the host to the batch devices.
Batch Transfer Facility Server/DI (BTFS/DI)	Server application that resides in every DI connected to batch output devices. It communicates with BTF/VE to transfer output files from the host to the batch devices.
OPERATE_STATION Utility (OPES)	Client application that resides in the NOS/VE host computer and validates user permissions and accepts commands to control an I/O station. It maintains a connection to SCFS/VE over which commands are sent and command responses are received. A separate instance of the OPERATE_STATION utility application is initiated whenever the OPERATE_STATION command is executed. The OPERATE_STATION utility, its subcommands, and I/O station control for NOS/VE are described in the CDCNET Batch Device User Guide.
Batch Transfer Facility Server/VE (BTFS/VE)	Server application that resides in all hosts that support batch input. It communicates with BTF/DI to transfer input files (jobs) from batch devices to the host.
Batch Transfer Facility/DI (BTF/DI)	Client application that resides in all DIs connected to batch input devices. It communicates with BTFS/VE to transfer input files (jobs) from batch devices to the host.

The relationships between the various BTF applications are shown in figure 2-7.

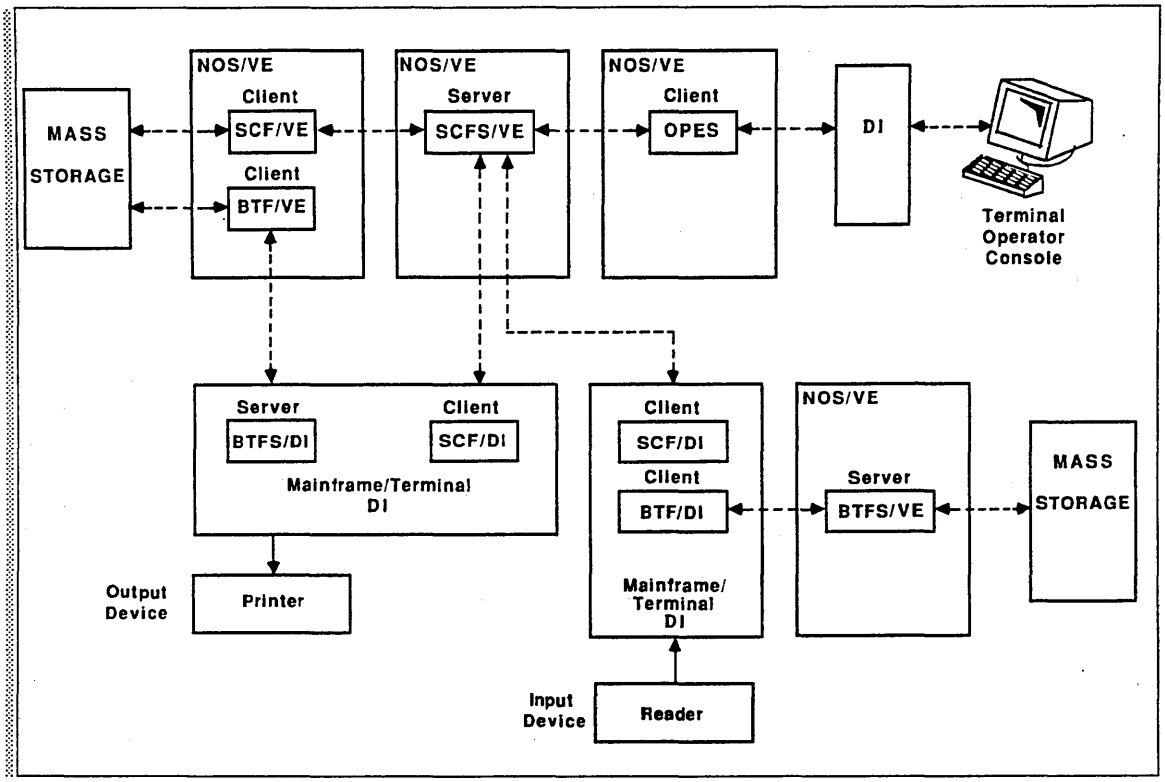


Figure 2-7. BTF Application Services

Network Transfer Facility (NTF)

The Network Transfer Facility (NTF) service allows you to transfer queued files between a NOS/VE host and another host within the same network. The NTF service supports the IBM Network Job Entry (NJE) protocol and the HASP multileaving protocol for communication between hosts.

The NTF service consists of the following applications:

Batch Transfer Facility/VE (BTF/VE)	Client application that resides in every host computer in the network that supports NTF. It communicates with BTFS/DI to transfer jobs and output files from the host to a remote system.
Batch Transfer Facility Server/VE (BTFS/VE)	Server application that resides in every host computer in the network that supports NTF. It communicates with BTF/DI to receive input files and jobs from a remote system.
Status and Control Facility Server/VE (SCFS/VE)	Server application (called the master I/O station control facility) that resides in the NOS/VE host computer. At least one SCFS/VE application must be present in each network. SCFS/VE schedules jobs and output files to batch streams. It also receives control and status commands from both files and batch streams.

Network Transfer Facility/VE (NTF/VE)	Client application that resides in every host computer in the network that supports NTF. NTF processes file control commands and informs SCFS/VE of jobs and output files available for transmission.
Operate NTF (OPENTF)	Client application in the host computer. OPENTF validates user permissions and establishes a connection with SCFS/VE for the operator.
Batch Transfer Facility/DI (BTF/DI)	Client application that transfers jobs and input files to BTFS/VE. BTF/DI processes file transfer commands for jobs and input files received from a remote system.
Batch Transfer Facility Server/DI (BTFS/DI)	Server application that receives output files from BTF/VE. BTFS/DI processes file transfer commands for jobs and output files transmitted to a remote system.
Status and Control Facility/DI (SCF/DI)	Client application that processes batch stream configuration commands from the terminal definition procedure and also processes batch stream control commands received from SCFS/VE. SCF/DI reports the status of NTF remote systems and batch streams to SCFS/VE.

The relationships between the various NTF applications are shown in figure 2-8.

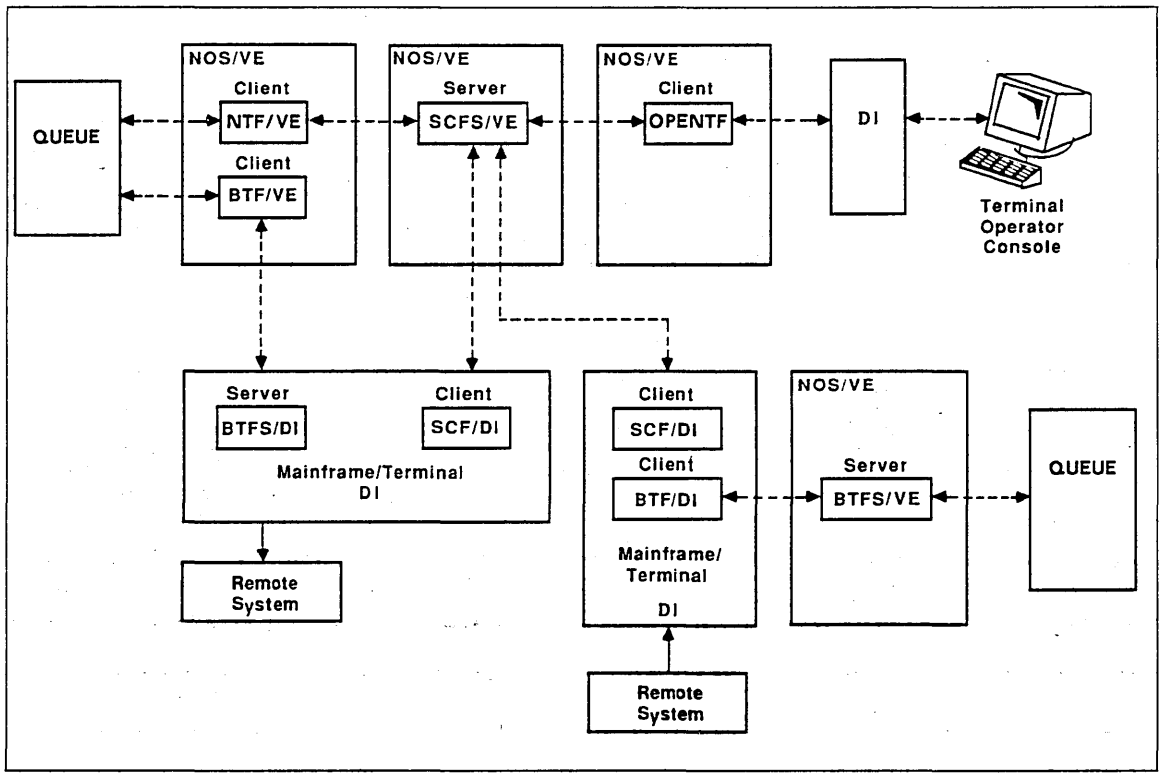


Figure 2-8. NTF Application Services

File Transfer Protocol/VE (FTP/VE)

The File Transfer Protocol/VE (FTP/VE) implements the File Transfer Protocol (FTP) of the set of protocols provided by the Transmission Control Protocol/Internet Protocol (TCP/IP). FTP/VE provides file transfer capabilities to and from connected TCP/IP hosts. Both the FTP client and FTP server protocols are supported. The FTP client application must be running in the local host and its associated FTP server application must be running in the remote host.

NOTE

Both the client application (FTP) and its associated server application (FTPS) are defined as client applications. See appendix C for more information about defining the File Transfer Protocol applications.

For detailed information on FTP/VE, see the TCP/IP Usage manual.

The relationship between the FTP client and server applications is shown in figure 2-9.

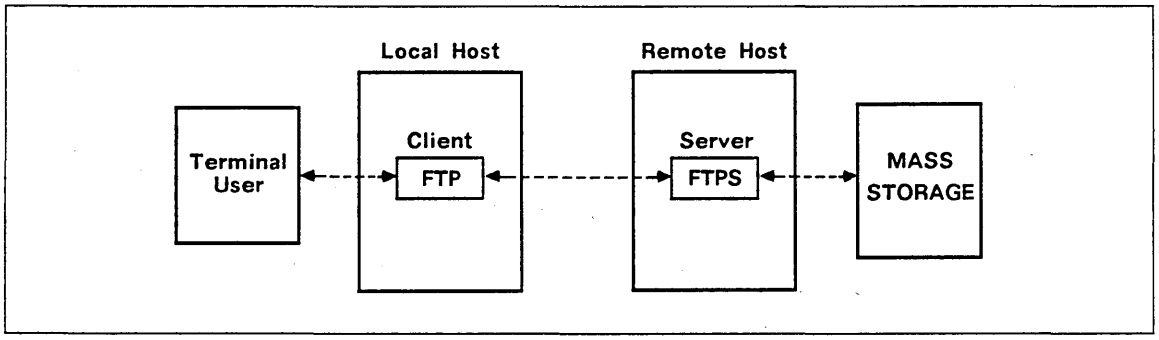


Figure 2-9. FTP Application Services

Simple Mail Transfer Protocol/VE (SMTP/VE)

The Simple Mail Transfer Protocol/VE (SMTP/VE) implements the Simple Mail Transfer Protocol (SMTP) of the set of protocols provided by the Transmission Control Protocol/Internet Protocol (TCP/IP). SMTP/VE provides mail transfer capabilities including mail relay to and from connected TCP/IP hosts. Both the SMTP client and SMTP server protocols are supported. A NOS/VE mail application can communicate with SMTP/VE through the NOS/VE batch mail queue to send and receive mail messages. A command interface is also provided to allow site-written mail applications to directly access SMTP/VE mail services.

NOTE

Both the client application (SMTP) and its associated server application (SMTPS) are defined as client applications. See appendix C for more information about defining the Simple Mail Transfer Protocol applications.

For detailed information about SMTP/VE, see the TCP/IP Usage manual.

The relationship between the SMTP/VE client, server, mail queue, and command interface and the local (user-written) mail application is shown in figure 2-10.

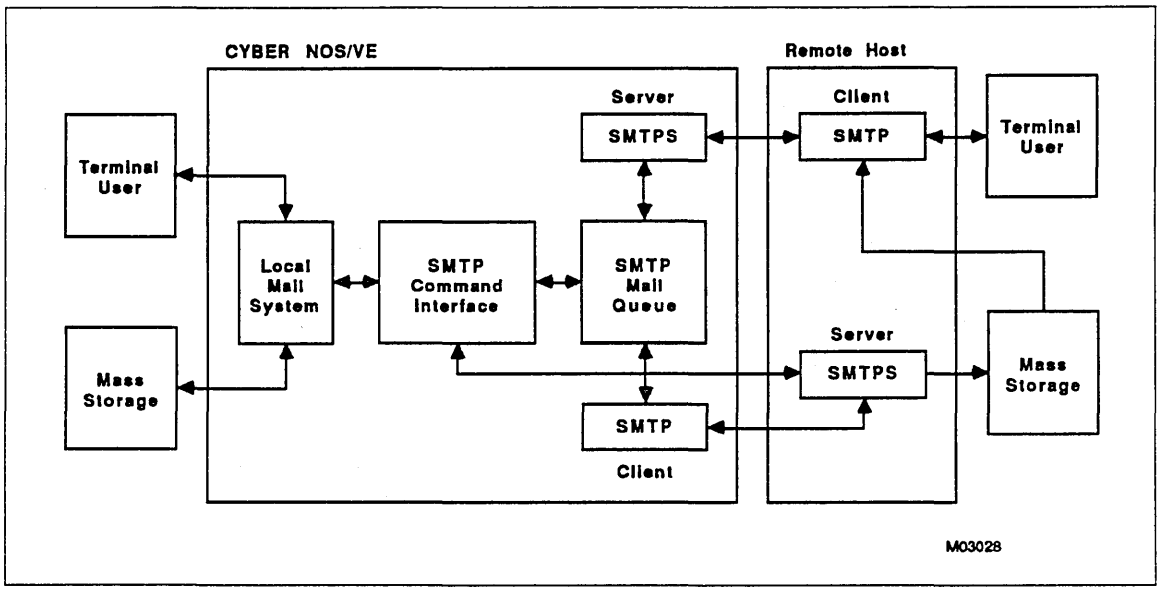


Figure 2-10. SMTP/VE Services

Network File System/VE (NFS/VE)

The Network File System on NOS/VE (NFS/VE) provides file server support for the NFS 3.2 protocol developed by Sun Microsystems, Inc. [1] This support allows any NOS/VE system to act as a file server for client systems with the NFS 3.2 client functions installed.

The NFS/VE server is made up of the following Interprocess Communication (IPC) applications:

- NFS/VE, which contains the bulk of Control Data's implementation of the Network File System.
- Portmapper, which maps calls from client systems into registered server calls.
- PCNFSD, which allows PCs running PC-NFS [2] to be validated to use NFS/VE.

These applications reside on the server system; that is, on the CYBER host running NOS/VE. These applications interface with Sun Microsystems' NFS protocols, Remote Procedure Call (RPC) and External Data Representation (XDR), on the CYBER. These server-based protocols then interface with the RPC and XDR protocols on the client system.

1. Sun Microsystems is a registered trademark and NFS is a trademark of Sun Microsystems, Inc.
2. PC-NFS is a product and trademark of Sun Microsystems, Inc.

NOTE

These Interprocess Communication applications are defined as a single client application. See appendix C for more information about defining the IPC applications.

For detailed information about NFS/VE, see the TCP/IP Usage manual.

Using the MANAGE_NETWORK_ APPLICATIONS Utility 3

Using the MANAGE_NETWORK_ APPLICATIONS Utility 3

Introduction

This chapter shows how you can use the MANAGE_NETWORK_APPLICATIONS (MANNA) utility to manage network applications. The following topics are discussed:

- Relationship between the MANNA utility and network application definitions.
- Using MANNA to define, redefine, or change application definitions.
- Using MANNA to display application status and attributes.
- Updating the application definitions file.
- Attribute descriptions for client and server applications.

A network application definition is a set of attributes that defines the characteristics of a network application. These application definitions identify the network application programs authorized to execute on a NOS/VE system. The MANNA utility creates and maintains the application definitions on the $SYSTEM.NETWORK.APPLICATION.DEFINITIONS file.

The MANNA utility consists of MANNA subcommands and four subutilities that together allow you to define, redefine, and change network application definitions. See figure 3-1 for a hierarchical listing of the MANNA subcommands and subutilities. Complete formats of the MANNA utility command and each subutility subcommand are given in chapter 4.

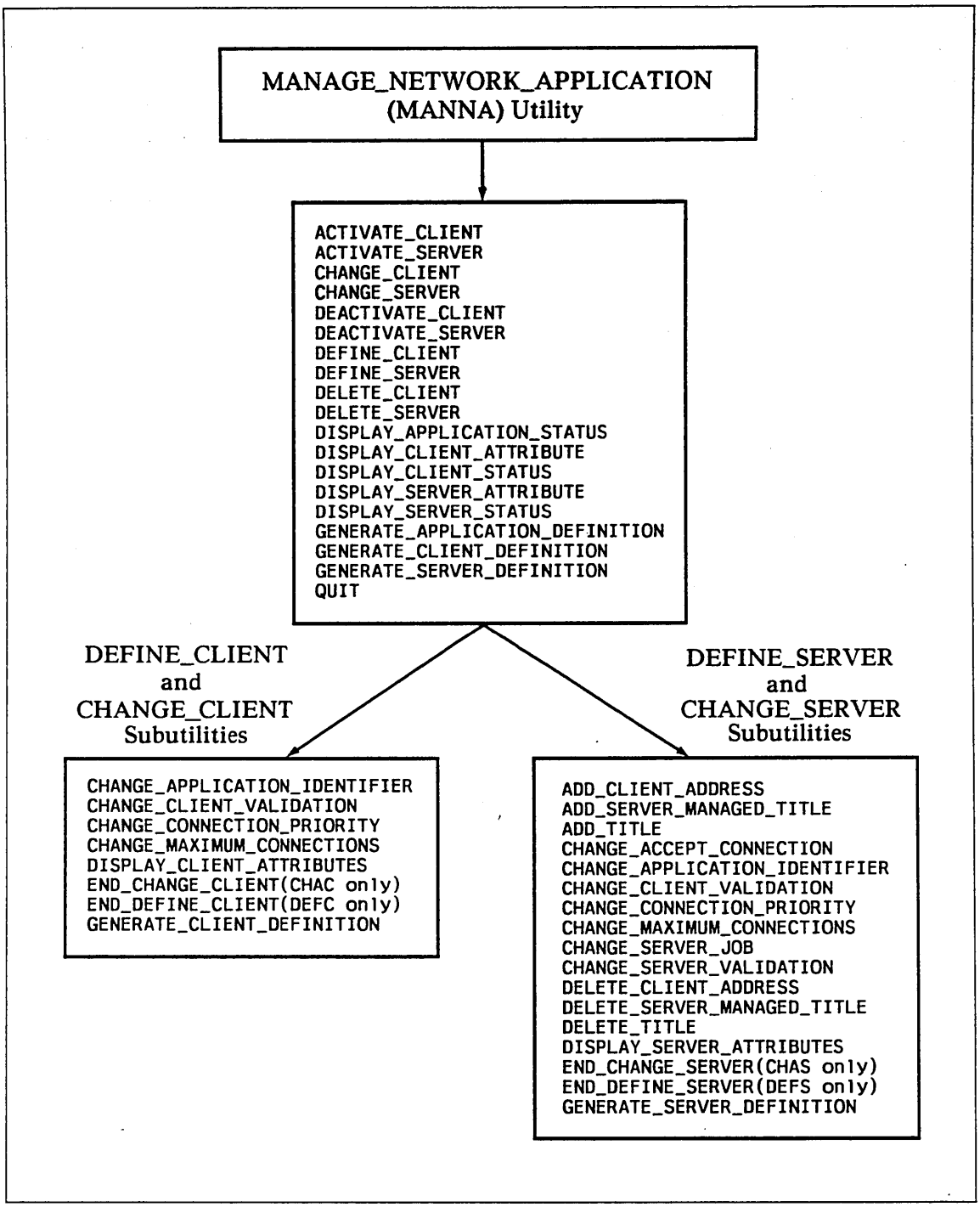


Figure 3-1. Manage Network Application Utility Hierarchy

The DEFINE_CLIENT and DEFINE_SERVER subutilities are used to change the initial values of client and server attributes. (The initial value of an attribute is a default value established by a define subutility.) The CHANGE_CLIENT and CHANGE_SERVER subutilities are used to change the previously defined attribute values.

Therefore, with the MANNA utility, you have the means to define new network applications or modify existing definitions, redefine or modify system-defined default values for application definitions, and define or modify user-defined application definitions.

Two methods are available to change an application's definition. You should use the method that is most appropriate for your purposes:

- Use the GENERATE_CLIENT_DEFINITION, GENERATE_SERVER_DEFINITION, or GENERATE_APPLICATION_DEFINITION MANNA subcommands to generate (on a new file) the subcommands that represent the current definition of the application on a file. After you edit this file, you can modify these subcommands to redefine the application.
- Use the MANNA CHANGE_CLIENT or CHANGE_SERVER subutilities to change only specific attribute values.

Prerequisites for Using MANNA

To use the MANNA utility, your validation must include the NETWORK_APPLICATION_MANAGEMENT capability. You are validated by the system administrator. See the NOS/VE User Validation manual for more information.

Restrictions on Using Subcommands

There are some restrictions on the use of various subcommands within the MANNA utility. The following restrictions apply:

- Subcommands that set application attributes can be entered only within the DEFINE_SERVER, CHANGE_SERVER, DEFINE_CLIENT, or CHANGE_CLIENT subutility.
- Subcommands that activate and deactivate applications and/or that delete application definitions can be entered only outside the subutilities.

Subcommands for displaying application attributes and status are not restricted. You can use them either inside or outside of the subutilities.

Whenever the client or server subutilities are used, the MANNA utility accesses the application definition file in exclusive access mode and enforces SCL restricted-command mode.

Defining an Application

Use the following sequence of steps to define a new application. Before defining a new application, you may wish to review the client and server application attributes in tables 3-1 and 3-2 or the MANNA subcommands in chapter 4 to determine the values you want to use for the attributes.

1. Start the MANNA utility with the MANAGE_NETWORK_APPLICATION command. After you call the utility, you receive this prompt:

   ```
   mna/
   ```

2. Start the subutility for the type of application you want to define:

   - To define a client application, start the client subutility with the DEFINE_CLIENT subcommand. Include the name of the application you are defining and the protocol the application uses. (To determine what protocol you should specify, see the Protocol attribute descriptions in table 3-3.) Once you call the client subutility, you receive this prompt:

     ```
     defc/
     ```

   - To define a server application, start the server subutility with the DEFINE_SERVER subcommand. Include the name of the application you are defining, the protocol the application uses, and how the application is initiated. Once you call the server subutility, you receive this prompt:

     ```
     defs/
     ```

3. If necessary, use the subcommands of the server or client subutility to change the initial values of the attributes as defined by the subutility. The attributes, their initial values, and the subcommands you use to change them are listed in table 3-1 (for client applications) and table 3-2 (for server applications). At any time in this process, you can use the DISPLAY_SERVER_ATTRIBUTE or DISPLAY_CLIENT_ATTRIBUTE subcommand to examine the current attribute values.

4. When you finish changing the values for the application attributes, save the definition by ending the subutility session:

   - To end the client subutility, enter the END_DEFINE_CLIENT subcommand.

   - To end the server subutility, enter the END_DEFINE_SERVER subcommand.

   The MANNA utility checks any definition that is to be saved. If the utility finds errors in the definition, it does not end the subutility session. Instead, it issues messages explaining the problem and gives you the opportunity to correct it.

5. If neccessary, you can discard the definition you created during the session by entering the SAVE_DEFINITION parameter and specifying the value FALSE. The following subcommands discard the client or server definition just created:

   ```
   defc/end_define_client save_definition=false

   defs/end_define_server save_definition=false
   ```

6. After you end the subutility, you again receive the MANNA utility prompt. You should now activate the application with either the ACTIVATE_CLIENT or ACTIVATE_SERVER subcommand. Include the parameter that specifies the name of the application to be activated.

7. Exit the MANNA utility with the QUIT subcommand.

Changing an Application

Use the following sequence of steps to change specific attribute values of an existing application definition. Before changing an application definition, you may wish to review the client and server application attributes in tables 3-1 and 3-2 or the MANNA subcommands in chapter 4 to determine the values you want to use for the attributes.

1. Start the MANNA utility with the MANAGE_NETWORK_APPLICATION command. After you call the utility, you receive this prompt:

   ```
   mna/
   ```

2. Start the subutility for the type of application you want to change:

   - To change a client application, start the client subutility with the CHANGE_CLIENT subcommand and specify the name of the application you are changing. Once you call the client subutility, you receive this prompt:

     ```
     chac/
     ```

   - To change a server application, start the server subutility with the CHANGE_SERVER subcommand and specify the name of the application you are changing. Once you call the server subutility, you receive this prompt:

     ```
     chas/
     ```

3. Use the subcommands of the server or client subutility to change the values of the attributes. The attributes and the subcommands you use to change them are listed in table 3-1 (for client applications) and table 3-2 (for server applications). At any time in this process, you can use the DISPLAY_SERVER_ATTRIBUTE or DISPLAY_CLIENT_ATTRIBUTE subcommand to examine the current attribute values.

4. When you finish changing the values for the application attributes, save the definition by ending the subutility session:

   - To end the client subutility, enter the END_CHANGE_CLIENT subcommand.

   - To end the server subutility, enter the END_CHANGE_SERVER subcommand.

   The MANNA utility checks any definition that is to be saved. If the utility finds errors in the definition, it does not end the subutility session. Instead, the attributes are restored to their values before any changes were entered, and the utility issues messages explaining the problem and gives you the opportunity to correct it.

5. If necessary, you can discard the definition you created during the session by entering the SAVE_DEFINITION parameter and specifying the value FALSE. The following subcommands discard the client or server definition just created:

```
chac/end_change_client save_definition=false

chas/end_change_server save_definition=false
```

6. After you end the subutility, you again receive the MANNA utility prompt. The changes made to the application definition take effect immediately for all new connections to the application. Existing connections are unaffected.

7. Exit the MANNA utility with the QUIT subcommand.

NOTE

You must be careful when changing any of the attributes for the system-defined NOS/VE applications. If you change some of the defined attributes, the application may function improperly or not at all. For example: If you change the ACCEPT_CONNECTION attribute to a value of TRUE for the SCFS/VE application or the PTF application, the application will not function. However, if you change other attributes, the application's behavior may not be altered, and you can then modify the application to meet your needs (for example, to add a title to the PTF server application).

Redefining an Application

Use the following sequence of steps to generate a copy of an existing application definition, edit this copy to redefine the application definition, and replace the original application definition with the edited copy. Before redefining an application, you may wish to review the client and server application attributes in tables 3-1 and 3-2 or the MANNA subcommands in chapter 4 to determine the values you want to use for the attributes.

1. Start the MANNA utility with the MANAGE_NETWORK_APPLICATION command. After you call the utility, you receive the prompt:

```
mna/
```

2. Obtain a copy of the application you wish to redefine using the GENERATE_SERVER_DEFINITION, the GENERATE_CLIENT_DEFINITION, or the GENERATE_APPLICATION_DEFINITION subcommand. Use the OUTPUT parameter to name an output file for the generated definition.

3. Enter the EDIT_FILE command to edit the output file and change the definition.

4. Deactivate the original application with the DEACTIVATE_SERVER or DEACTIVATE_CLIENT subcommand. The TERMINATE_ACTIVE_CONNECTIONS parameter on this subcommand allows you to deactivate the application immediately or let currently active connections end normally.

   - If you specify a value of FALSE, active connections are allowed to continue, but no new users are allowed to access the application. Use either the DISPLAY_CLIENT_STATUS or DISPLAY_SERVER_STATUS subcommand to check for active connections. Then, when all users are disconnected, the MANNA utility deactivates the application.

   - If you specify a value of TRUE, all connections to the application are terminated and the application is deactivated immediately.

5. Delete the original application definition with the DELETE_SERVER or DELETE_CLIENT subcommand.

6. Insert the output file containing the entire edited definition into the command stream using the INCLUDE_FILE command.

7. Reactivate the redefined application with the ACTIVATE_SERVER or ACTIVATE_CLIENT subcommand.

8. Exit the MANNA utility with the QUIT subcommand.

For example, to redefine the server application VE105, you enter:

```
manage_network_application
mna/generate_server_definition server=ve105 output=def
mna/edit_file file=def
   .
   . (make the appropriate changes to the definition)
   .
mna/deactivate_server server=ve105 terminate_active_connections=true
mna/delete_server server=ve105
mna/include_file file=def
mna/activate_server server=ve105
mna/quit
/
```

Displaying Application Status

Several MANNA subcommands allow you to display the status of network applications.

- To display the status of one or more applications use the DISPLAY_APPLICATION_STATUS subcommand. For example:

```
mna/display_application_status application=nve
```

  displays status for the application named NVE on file $OUTPUT. The subcommand:

```
mna/display_application_status application=all
```

  displays status information for all of the current applications in the system. To write the information to a file other than $OUTPUT, specify a file name on the OUTPUT parameter:

```
mna/display_application_status application=all output=$user.save_all
```

- To display status for only server applications or only client applications, you can use either the DISPLAY_SERVER_STATUS or DISPLAY_CLIENT_STATUS subcommand. You can specify either the name of one or more applications to be displayed or the keyword ALL. For example:

```
mna/display_server_status server=ve105
    .
    . (status display for server application ve105)
    .
mna/display_client_status client=all
    .
    . (status display for all client applications)
    .
```

Displaying Application Attribute Values

To display the current attribute values for a server or client application, you can use either the DISPLAY_SERVER_ATTRIBUTE or DISPLAY_CLIENT_ATTRIBUTE subcommand. You can enter one or more application names or the keyword ALL. Also, you can specify whether to display certain attributes or all attributes. For example:

```
mna/display_server_attribute server=ve105 attribute=protocol
   .
   . (attribute display of the protocol for server application VE105)
   .
mna/display_client_attribute client=ve020 attributes=all
   .
   . (attribute display of all attributes for client application VE020)
   .
```

The attribute display subcommands can be entered both inside and outside the subutilities. If you enter a display subcommand outside a subutility and do not specify the applications to be displayed, the attributes for all defined applications are displayed. If you enter a display subcommand inside a subutility and don't specify the applications to be displayed, the attributes for the application currently being defined in the subutility are displayed. For example:

```
mna/display_server_attribute
   .
   . (attribute display of all attributes of all applications)
   .
mna/define_server server=ve100 protocol=cdna_session
defs/display_server_attribute
   .
   . (attribute display of all attributes for application VE100)
   .
```

Updating the Application Definitions File

The MANNA utility performs the following functions whenever you update the application definitions file:

- When you call a define or a change subutility, it attaches the highest cycle of the application definitions file for exclusive access. This prevents concurrent updates.
- The utility checks to see if cycle 1 of the application definitions file already exists. (During the update process, MANNA uses cycle 1 as a temporary definitions file.) If cycle 1 is found, the utility assumes that an update was attempted and failed. Therefore, the utility issues a message to the job log indicating that a previous update attempt has failed, and deletes cycle 1.
- Your changes to the application definitions are made on a working copy of the definitions file within the utility session.
- The modified definitions file is rewritten as cycle 1 of the file.
- The retention period of the previous cycle of the file is changed to two days.
- The utility changes cycle 1 to be the highest cycle of the file, then this cycle of the file is returned.

This updating process ensures that the highest cycle of the application definitions file is valid, and (using the cycle number) it provides a convenient way to identify the latest version of the file on permanent file backup tapes.

The utility does not delete older versions (cycles) of the application definitions file. Each site is responsible for deleting unneeded cycles to conserve file space.

Application Attributes

Client and server applications have separate sets of attributes. Table 3-1 lists client application attributes, their initial values (which are set by the DEFINE_CLIENT subutility), and the subcommands you can use to change their values. Table 3-2 lists the same information for a server application. Tables 3-3 and 3-4 give you an expanded description of each attribute.

Table 3-1. Client Application Attributes Overview

Attribute	Initial Value	Subcommand
Application identifier	VARIABLE	CHANGE_APPLICATION_ IDENTIFIER
Client capability	NONE	CHANGE_CLIENT_VALIDATION
Client ring	13	CHANGE_CLIENT_VALIDATION
Client system privilege	FALSE	CHANGE_CLIENT_VALIDATION
Connection priority	0	CHANGE_CONNECTION_PRIORITY
Maximum connections	65,535	CHANGE_MAXIMUM_CONNECTIONS
Protocol	None[1]	DEFINE_CLIENT

1. You must specify a protocol.

Table 3-2. Server Application Attributes Overview

Attribute	Initial Value	Subcommand
Accept connection	TRUE	CHANGE_ACCEPT_CONNECTION
Application identifier	VARIABLE	CHANGE_APPLICATION_IDENTIFIER
Broadcast registration	FALSE	ADD_TITLE
Client address	No list	ADD_CLIENT_ADDRESS and DELETE_CLIENT_ADDRESS
Client information source	DIALOG	None (this attribute cannot be changed)
Client validation	NONE	CHANGE_CLIENT_VALIDATION
Connection priority	0	CHANGE_CONNECTION_PRIORITY
Maximum connections	65,535	CHANGE_MAXIMUM_CONNECTIONS
NAM-initiated	TRUE	DEFINE_SERVER
Protocol	None[1]	DEFINE_SERVER
Server capability	NONE	CHANGE_SERVER_VALIDATION
Server job	A null file	CHANGE_SERVER_JOB
Server job maximum connections	65,535	CHANGE_SERVER_JOB
Server job validation source	SERVER	CHANGE_SERVER_JOB
Server managed titles	No list	ADD_SERVER_MANAGED_TITLE and DELETE_SERVER_MANAGED_TITLE
Server ring	13	CHANGE_SERVER_VALIDATION
Server system privilege	FALSE	CHANGE_SERVER_VALIDATION
Title	No list	ADD_TITLE and DELETE_TITLE
Title data	No string	ADD_TITLE
Title priority	1	ADD_TITLE

1. You must specify a protocol.

Table 3-3. Client Application Attribute Descriptions

Attribute	Description
Application Identifier	Indicates the identifier to be used in the application's network address. You can choose a specific number in the range from 2000 to 3000 to be used for the identifier, or specify the keyword VARIABLE. VARIABLE causes an identifier to be assigned from a pool of identifiers each time the application is initiated. The initial value is VARIABLE. You can specify this attribute with the CHANGE_APPLICATION_IDENTIFIER subcommand.
Client Capability	Indicates the NOS/VE capability (ability to access special system features or utilities) that the user must have in order to call NAM/VE interfaces that perform client application functions. You may set this attribute's value to name one of the following capabilities: • EXPLICIT_REMOTE_FILE • IMPLICIT_REMOTE_FILE • HPA_VE_INITIATOR • NETWORK_APPLICATION_MANAGEMENT • NETWORK_OPERATION • NTF_OPERATION • STATION_OPERATION • TIMESHARING • Site-defined capability The capability you specify on this attribute must be present in the validation file entry for the user in whose job the client application executes. The family administrator is responsible for assigning this capability to the user in the NOS/VE validation file. For more information about these capabilities, see the NOS/VE User Validation manual listed in appendix B. The initial value is NONE (no capability is required). You can specify this attribute with the CHANGE_CLIENT_VALIDATION subcommand.
Client Ring	Indicates the highest ring from which a client application may call certain NAM/VE interfaces that perform client application functions. The initial value is 13 (the interfaces can be called from any ring up to ring 13). You can specify this attribute with the CHANGE_CLIENT_VALIDATION subcommand.

(Continued)

Table 3-3. Client Application Attribute Descriptions *(Continued)*

Attribute	Description
Client System Privilege	Indicates whether a client application must be system-privileged to call NAM/VE program interfaces that perform client application functions. (System-privileged code is code that is either in a system job or executes at ring 3 or lower.) The initial value is FALSE (the affected NAM/VE interfaces can be called from any user code). You can specify this attribute with the CHANGE_CLIENT_VALIDATION subcommand. **NOTE** This attribute is not operational for this release of NOS/VE.
Connection Priority	Specifies the priority associated with the data sent by the client application over a network connection. Each client application can specify the priority of the data it sends. Client and server applications need not assign the same priority to the connection between them. The initial value is 0, which is the lowest priority. You can specify this attribute with the CHANGE_CONNECTION_PRIORITY subcommand.
Maximum Connections	Indicates the maximum number of concurrent connections that can be established by all instances of a client application. The initial value is 65,535. You can specify this attribute with the CHANGE_MAXIMUM_CONNECTIONS subcommand.
Protocol	Indicates the type of connection that is established by the application. Connection types are identified by the protocol that is used to provide a connection. An application can use only one type of connection. CDNA_SESSION protocol (used for host-to-host connections) and CDNA_VIRTUAL_TERMINAL protocol (used for CDCNET -connected terminals) are currently supported. (For more information on CDCNET protocols, refer to CDCNET System Programmer's Reference, Volume 3.) There is no initial value set by the system. You must specify this attribute with the DEFINE_CLIENT subcommand.

Table 3-4. Server Application Attribute Descriptions

Attribute	Description
Accept Connection	Indicates whether NAM/VE should accept connection requests on behalf of the server application. If NAM/VE accepts connections, the server application need not explicitly accept a connection after acquiring it. NAM/VE must accept a requested connection if a dialog with the client is necessary. Therefore, if validation information is required from the client, and the CLIENT_INFORMATION_SOURCE attribute specifies that the information came from a dialog, the accept connection attribute must be TRUE. (See also the CLIENT_VALIDATION attribute and the SERVER_JOB_VALIDATION_SOURCE attribute.) The initial value is TRUE. You can specify this attribute with the CHANGE_ACCEPT_CONNECTION subcommand.
Application Identifier	Indicates the identifier to be used in the application's network address. You can choose a specific number in the range from 2000 to 3000 to be used for the identifier, or specify the keyword VARIABLE. VARIABLE causes an identifier to be assigned from a pool of identifiers each time the application is initiated. The initial value is VARIABLE. You can specify this attribute with the CHANGE_APPLICATION_IDENTIFIER subcommand.
Broadcast Registration	Indicates whether a title translation should be distributed to a remote system whenever the title is registered in the network directory. Titles should be distributed when a client application uses a recurrent directory search in order to learn of new servers on the network. The initial value is FALSE (titles are not distributed). You can specify this attribute with the ADD_TITLE subcommand.
Client Address	Indicates the network addresses from which a client application may connect to a server. If a client application resides at one of the addresses in this list, NAM/VE assumes that it is authorized to request a connection to the server application. If no addresses are specified, clients at any address are authorized to connect to the server. The initial value is no list of network addresses. You can add client addresses with the ADD_CLIENT_ADDRESS subcommand and delete them with the DELETE_CLIENT_ADDRESS subcommand.

(Continued)

Table 3-4. Server Application Attribute Descriptions *(Continued)*

Attribute	Description
Client Information Source	Indicates how validation information should be obtained from a client application, if necessary. (This information could be required for client validation and/or server job validation.) The value DIALOG indicates that client information is obtained via a dialog using the connection from the client. The initial value is DIALOG. This attribute cannot be changed.
Client Validation	Indicates the NOS/VE capability (ability to access special system features or utilities) that the client must have to access the server. If you want the server application to require validation from the client, you may set this attribute's value to name one of the following capabilities: • EXPLICIT_REMOTE_FILE • IMPLICIT_REMOTE_FILE • HPA_VE_INITIATOR • NETWORK_APPLICATION_MANAGEMENT • NETWORK_OPERATION • NTF_OPERATION • STATION_OPERATION • TIMESHARING • Site-defined capability The capability you specify on this attribute must be present in the client's validation file entry. The family administrator is responsible for assigning this capability to the client in the NOS/VE validation file. For more information about these capabilities, see the NOS/VE User Validation manual listed in appendix B. The initial value is NONE. You can specify this attribute on the CHANGE_CLIENT_VALIDATION subcommand.
Connection Priority	Specifies the priority associated with the data sent by the server application over a network connection. Each server application can specify the priority of the data it sends. Client and server applications need not assign the same priority to the connection between them. The initial value is 0, which is the lowest priority. You can specify this attribute with the CHANGE_CONNECTION_PRIORITY subcommand.

(Continued)

Table 3-4. Server Application Attribute Descriptions *(Continued)*

Attribute	Description
Maximum Connections	Indicates the maximum number of concurrent connections that can be established by all instances of a server application. This value is the total number of connections active for the application. Do not confuse it with the maximum number of concurrent connections for a single instance of an application, which is set by the SERVER_JOB_MAXIMUM_CONNECTIONS attribute. The initial value is 65,535. You can specify this attribute with the CHANGE_MAXIMUM_CONNECTIONS subcommand.
NAM-Initiated	Indicates whether NAM/VE initiates a server application job. The initial value is TRUE. You can specify this attribute with the DEFINE_SERVER subcommand.
Protocol	Indicates the type of connection that is established by the application. Connection types are identified by the protocol that is used to provide a connection. An application can use only one type of connection. CDNA_SESSION protocol (used for host-to-host connections) and CDNA_VIRTUAL_TERMINAL protocol (used for CDCNET -connected terminals) are currently supported. For more information on CDCNET protocols, refer to CDCNET System Programmer's Reference, Volume 3. There is no initial value set by the system. You must specify this attribute with the DEFINE_SERVER subcommand.
Server Capability	Indicates the NOS/VE capability required to call NAM/VE program interfaces that perform server application functions. You may set this attribute's value to one of the following capabilities: • EXPLICIT_REMOTE_FILE • IMPLICIT_REMOTE_FILE • HPA_VE_INITIATOR • NETWORK_APPLICATION_MANAGEMENT • NETWORK_OPERATION • NTF_OPERATION • STATION_OPERATION • TIMESHARING • Site-defined capability

(Continued)

Table 3-4. Server Application Attribute Descriptions *(Continued)*

Attribute	Description
	The capability you specify on this attribute must also be present in a validation file entry for the user in whose job the server application executes. The family administrator is responsible for assigning this capability to the user in the NOS/VE validation file. For more information about these capabilities, see the NOS/VE User Validation manual listed in appendix B. The initial value is NONE (no capability is required). You can specify this attribute with the CHANGE_SERVER_VALIDATION subcommand.
Server Job	Specifies a file that contains the sequence of SCL commands that perform the function of a NAM-initiated server application (that is, the sequence of commands executes an instance of the server application). The sequence of commands can contain any system-defined or user-defined commands. The initial value is a null file (no list of SCL commands). You can specify this attribute with the CHANGE_SERVER_JOB subcommand.
Server Job Maximum Connections	Indicates the maximum number of concurrent connections that a NAM-initiated server application job can support. This value is the number of concurrent connections that a single instance of an application can have. Do not confuse it with the total number of connections active for all instances of an application (this latter number is set by the MAXIMUM_CONNECTIONS attribute). The initial value is 65,535. You can specify this attribute with the CHANGE_SERVER_JOB subcommand.
Server Job Validation Source	Indicates the validation information source for a NAM-initiated server job. If you specify SERVER, the first command in the server job must be a LOGIN command that supplies the necessary information. If you specify CLIENT, validation information is obtained from the client, and the server job cannot contain a LOGIN command. The initial value is SERVER. You can specify this attribute with the CHANGE_SERVER_JOB subcommand.
Server Managed Titles	Specifies a list of title patterns that defines the set of server titles that can be managed by the server application itself via NAM/VE program interfaces. These interfaces allow the server application to add or delete titles to the list of titles registered in the network directory for the server. A title managed via the interfaces must match a title pattern in the list defined by this attribute. The initial value is no list of title patterns. You can add title patterns with the ADD_SERVER_MANAGED_TITLE subcommand or delete title patterns with the DELETE_SERVER_MANAGED_TITLE subcommand.

(Continued)

Table 3-4. Server Application Attribute Descriptions *(Continued)*

Attribute	Description
Server Ring	Indicates the highest ring from which a server application may call certain NAM/VE interfaces that perform server application functions. The initial value is 13 (the interfaces can be called from any ring up to ring 13). You can specify this attribute with the CHANGE_SERVER_VALIDATION subcommand.
Server System Privilege	Indicates whether a server application must be system-privileged to call NAM/VE interfaces that perform server application functions. (System-privileged code is code that is either in a system job or executes at ring 3 or lower.) The initial value is FALSE (the affected NAM/VE interfaces can be called from any user code). You can specify this attribute with the CHANGE_SERVER_VALIDATION subcommand. **NOTE** This attribute is not operational for this release of NOS/VE.
Title	Indicates the title or titles to be registered in the network directory for a server application. The initial value is no list of titles. You can add titles with the ADD_TITLE subcommand and delete titles with the DELETE_TITLE subcommand.
Title Data	Indicates the string of data to be delivered to a client application with the translation of a title. This string can have up to 32 characters. The initial value is no string of data. You can specify this attribute with the ADD_TITLE subcommand.
Title Priority	Indicates the priority of the server application whose title is registered in the network directory relative to other servers that register the same title. The initial value is 1 (maximum priority). You can specify a number in the range from 1 to 255. You can specify this attribute with the ADD_TITLE subcommand.

MANAGE_NETWORK_APPLICATIONS Command and Subcommands 4

MANAGE_NETWORK_APPLICATIONS Command and Subcommands 4

MANNA Utility

The MANAGE_NETWORK_APPLICATIONS (MANNA) utility is an SCL command utility. It follows the rules for utilities given in the NOS/VE System Usage manual.

Except for QUIT, all of the utility subcommands have an optional STATUS parameter. Information on using the STATUS parameter can be found in the NOS/VE System Usage manual.

You must be validated by the system administrator to use the MANNA utility. For more information, refer to the NOS/VE User Validation manual.

Certain MANNA subcommands that define or change applications operate as subutilities of the MANNA utility. These subutililies are: DEFINE_CLIENT, DEFINE_SERVER, CHANGE_CLIENT, and CHANGE_SERVER. These subutilities allow you to change and undo variations to an application definition. To initiate one of these subutilities, use the associated MANNA subcommand: DEFINE_CLIENT (DEFC), DEFINE_SERVER (DEFS), CHANGE_CLIENT (CHAC), or CHANGE_SERVER (CHAS).

MANAGE_NETWORK_APPLICATION Command

Purpose Initiates a MANAGE_NETWORK_APPLICATIONS utility session.

Format

MANAGE_NETWORK_APPLICATION or
MANAGE_NETWORK_APPLICATIONS or
MANNA
STATUS=status variable

Parameters *STATUS*

Returns the completion status of this command.

Remarks

- To use this utility, you must be validated by the family administrator. For more information, refer to the NOS/VE User Validation manual.
- Only one user at a time can modify the application definitions file.
- NAM/VE need not be active to invoke the MANNA utility.
- The prompt for the MANNA utility is:

```
mna/
```

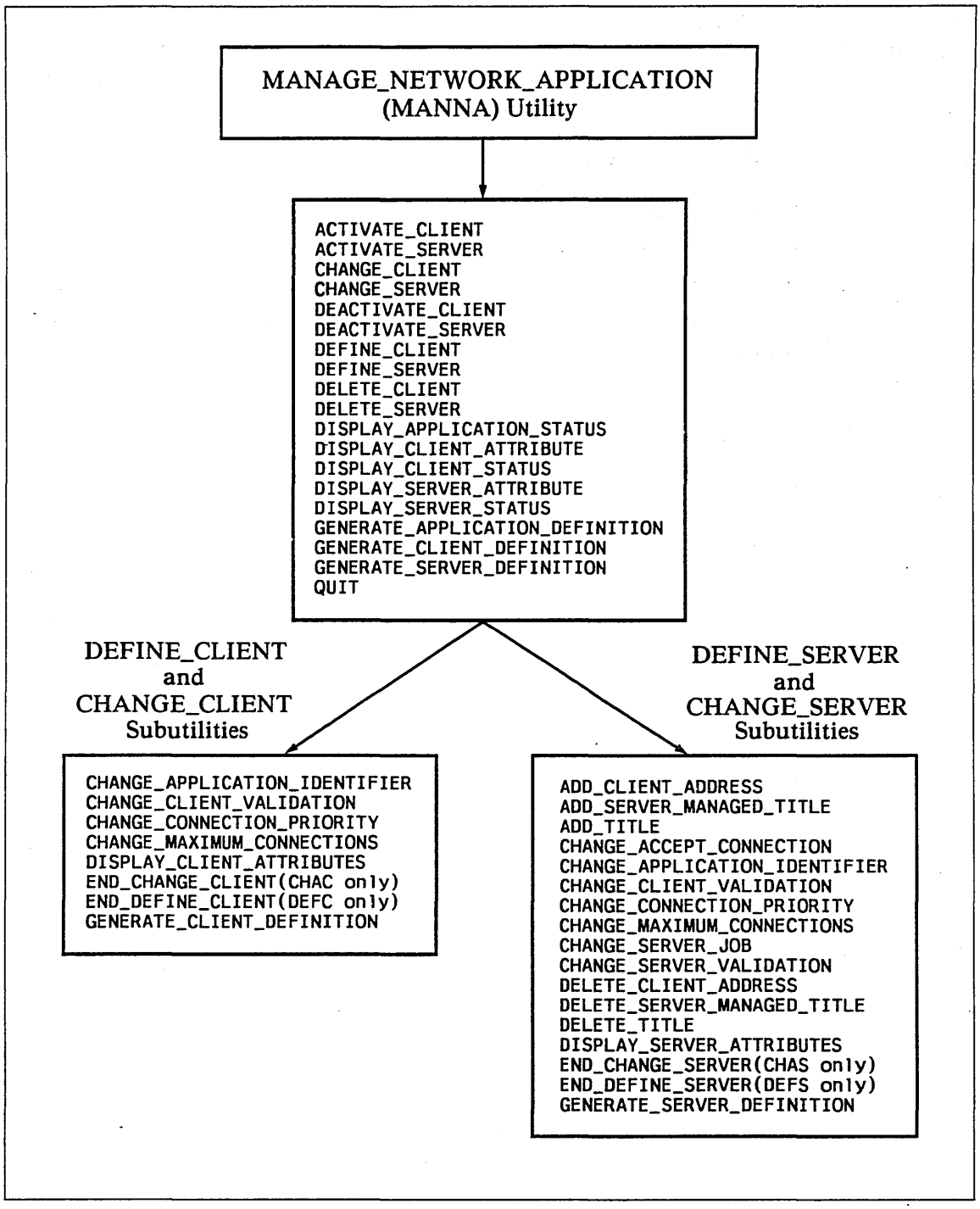


Figure 4-1. MANNA Subcommands

ACTIVATE_CLIENT MANNA Subcommand

Purpose Activates a defined client application.

Format

ACTIVATE_CLIENT or
ACTC
CLIENT=name
STATUS=status variable

Parameters

CLIENT or **C**

Specifies the name of the client application to be activated. This parameter is required.

STATUS

Returns the completion status of this subcommand.

Remarks A client application must be activated before it can make a connection request.

ACTIVATE_SERVER
MANNA Subcommand

Purpose Activates a defined server application.

Format

ACTIVATE_SERVER or
ACTS
SERVER=name
STATUS=status variable

Parameters

SERVER or **S**

Specifies the name of the server application to be activated. This parameter is required.

STATUS

Returns the completion status of this subcommand.

Remarks If the server application is a NAM-initiated server, it must be activated before it can receive connection requests from client applications. If the server application is not a NAM-initiated server, it must be executing in order to receive connection requests from client applications.

ADD_CLIENT_ADDRESS
DEFS and CHAS Subcommand

Purpose Specifies a network address to be added to the list of addresses from which client applications are authorized to connect to the server application.

Format

ADD_CLIENT_ADDRESS or
ADDCA
NETWORK_IDENTIFIER=keyword or *integer*
SYSTEM_IDENTIFIER=keyword or *integer*
APPLICATION_IDENTIFIER=keyword or *integer*
STATUS=status variable

Parameters

NETWORK_IDENTIFIER or *NI*

Specifies the network identifier that must be present in the network address of a client application in order to connect to the server application. The value specified can be a hexadecimal network identifier (integer) in the range from 1 to 0FFFFFFFF. The keyword ALL can be specified to indicate that the network identifier in the client's address is not to be used to limit access to the server application.

The default value is ALL.

SYSTEM_IDENTIFIER or *SI*

Specifies the system identifier that must be present in the network address of a client application in order to connect to the server application. The value specified can be a hexadecimal system identifier (integer) in the range from 1 to 0FFFFFFFFFFFF. The following keyword options are available:

ALL

Indicates that the system identifier in the client's address is not to be used to limit access to the server application.

NOSVE

Indicates that the system identifier of any NOS/VE system is to be accepted.

CDCNET

Indicates that the system identifier of any CDCNET DI is to be accepted.

The default value is ALL.

APPLICATION_IDENTIFIER or *AI*

Specifies the application identifier that must be present in the network address of a client application in order to connect to the server application. The value specified can be an application identifier (integer) in the range from 2000 to 3000. The keyword ALL can be specified to indicate that the application identifier in the client's address is not to be used to limit access to the server application.

The default is ALL.

STATUS

Returns the completion status of this subcommand.

Remarks

- Each ADD_CLIENT_ADDRESS subcommand creates an entry in a list of valid client addresses. A connection request from a client at a network address in the list is processed. A connection request from a client at a network address not in the list is rejected by NAM/VE.
- The DELETE_CLIENT_ADDRESS subcommand removes addresses from the list.
- Chapter 2 contains additional information on validating connection requests based on the client address.

ADD_SERVER_MANAGED_TITLE DEFS and CHAS Subcommand

Purpose Adds one or more title patterns to the list of title patterns that defines the set of server titles the server application can manage.

Format

ADD_SERVER_MANAGED_TITLE or
ADD_SERVER_MANAGED_TITLES or
ADDSMT
TITLE_PATTERN=list of name or **string**
STATUS=status variable

Parameters

TITLE_PATTERN or **TITLE_PATTERNS** or **TP**

Specifies title patterns to be added to the list of title patterns. A title pattern can be a name, a string, or a list of names and strings. For more information on title patterns, see chapter 2.

This parameter is required.

STATUS

Returns the completion status of this subcommand.

Remarks The DELETE_SERVER_MANAGED_TITLE subcommand removes title patterns from the list of title patterns for the server application.

ADD_TITLE
DEFS and CHAS Subcommand

Purpose Adds titles to the list of titles to be registered in the network directory for the server application.

Format

ADD_TITLE or
ADD_TITLES or
ADDT
TITLE=list of name or string
BROADCAST_REGISTRATION=boolean
PRIORITY=integer
DATA=string
STATUS=status variable

Parameters

TITLE or **TITLES** or **T**

Specifies titles to be added to the list of titles for the server application. A title can be a name, a string, or a list of names and strings.

When you specify a name, letters are converted to uppercase before the title is registered in the network directory. For more information on creating titles, see chapter 2.

This parameter is required.

BROADCAST_REGISTRATION or *BR*

Specifies the value of the BROADCAST_REGISTRATION attribute.

TRUE

Indicates that title translations should be distributed throughout the network once they are registered in the directory.

FALSE

Indicates that the title translations are not to be distributed.

The default is FALSE.

PRIORITY or *P*

Specifies the directory priority associated with the title. You can specify an integer from 1 to 255.

The default is 1, which indicates the highest priority.

DATA or *D*

Specifies the application-specific data to be delivered to the client application with a translation of this title. You can specify a string of up to 32 characters.

The default is no string.

STATUS

Returns the completion status of this subcommand.

Remarks The DELETE_TITLE subcommand removes titles from the list of titles for the server application.

CHANGE_ACCEPT_CONNECTION DEFS and CHAS Subcommand

Purpose Specifies whether NAM/VE should accept connection requests on behalf of the server application.

Format **CHANGE_ACCEPT_CONNECTION** or
CHAAC
ACCEPT_CONNECTION=boolean
STATUS=status variable

Parameters **ACCEPT_CONNECTION** or **AC**

Specifies the value of the ACCEPT_CONNECTION attribute.

TRUE

Indicates that NAM/VE is to accept connection requests before assigning the connection to a server application job.

FALSE

Indicates that NAM/VE is not to accept connection requests before assigning the connection to a server application job.

The initial value is TRUE. This parameter is required.

STATUS

Returns the completion status of this subcommand.

Remarks The ACCEPT_CONNECTION attribute must be TRUE when both of the following conditions are TRUE:

- The client application is required to supply validation information (by either the SERVER_JOB_VALIDATION_SOURCE attribute or the CLIENT_VALIDATION attribute).
- The validation information must come from a dialog (specified by the CLIENT_INFORMATION_SOURCE attribute).

CHANGE_APPLICATION_IDENTIFIER DEFC and CHAC Subcommand

Purpose Changes the application identifier used for the client application.

Format

CHANGE_APPLICATION_IDENTIFIER or
CHAAI
APPLICATION_IDENTIFIER=keyword or **integer**
STATUS=status variable

Parameters

APPLICATION_IDENTIFIER or **AI**

Specifies the value of the APPLICATION_IDENTIFIER attribute. The value specified can be an integer in the range from 2000 to 3000. The keyword VARIABLE can be specified to indicate that the application identifier is to be assigned from a pool of values when the client application is initiated.

The initial value is VARIABLE. This parameter is required.

STATUS

Returns the completion status of this subcommand.

Remarks

- If you use the CHANGE_APPLICATION_IDENTIFIER subcommand to change the application identifier of an active client, this change affects only those new connections created after the changed client definition is saved.
- If the keyword VARIABLE is specified, a different value can be assigned each time the application is activated.

CHANGE_APPLICATION_IDENTIFIER DEFS and CHAS Subcommand

Purpose Changes the application identifier used for the server application.

Format

CHANGE_APPLICATION_IDENTIFIER or
CHAAI
APPLICATION_IDENTIFIER=keyword or **integer**
STATUS=status variable

Parameters

APPLICATION_IDENTIFIER or **AI**

Specifies the value of the APPLICATION_IDENTIFIER attribute. The value specified can be an integer in the range from 2000 to 3000. The keyword VARIABLE is specified to indicate that the application identifier is to be assigned from a pool of values when the server application is initiated.

The initial value is VARIABLE. This parameter is required.

STATUS

Returns the completion status of this subcommand.

Remarks

- If you use the CHANGE_APPLICATION_IDENTIFIER subcommand to change the application identifier of an active server, this change affects only those new connections created after the changed server definition is saved.
- If the keyword VARIABLE is specified, a different value can be assigned each time the application is activated.
- The SCFS/VE application name (default name of OSA$STATUS_CONTROL_FAC_SERVER) should be deactivated before its application identifier is changed.

CHANGE_CLIENT
MANNA Subcommand

Purpose Initiates a CHANGE_CLIENT subutility session that changes the definition of an existing client application.

Format

CHANGE_CLIENT or
CHAC
CLIENT=name
STATUS=status variable

Parameters

CLIENT or **C**

Specifies the name of the client application whose definition is to be changed. This parameter is required.

STATUS

Returns the completion status of this subcommand.

Remarks

- The CHANGE_CLIENT subutility prompt is:

  ```
  chac/
  ```

- While you are in a CHANGE_CLIENT subutility session, you can use the following subcommands to change the application's attributes:
  - CHANGE_APPLICATION_IDENTIFIER
  - CHANGE_CLIENT_VALIDATION
  - CHANGE_CONNECTION_PRIORITY
  - CHANGE_MAXIMUM_CONNECTIONS

- When you end the CHANGE_CLIENT subutility with the END_CHANGE_CLIENT subcommand, you have the option of either making the changes permanent or discarding the changes you have made.

CHANGE_CLIENT_VALIDATION DEFC and CHAC Subcommand

Purpose Changes the client validation that NAM/VE performs to verify that the user of the client application is authorized to perform the functions of the client application.

Format

CHANGE_CLIENT_VALIDATION or
CHACV
CAPABILITY=keyword or *name*
RING=integer
SYSTEM_PRIVILEGE=boolean
STATUS=status variable

Parameters *CAPABILITY* or *C*

Specifies the name of the CLIENT_CAPABILITY attribute that must be in the calling user's validation file entry in order to call the affected NAM/VE program interfaces. The keyword NONE can be specified to indicate that no capability is to be validated.

The initial value is NONE. If you omit this parameter, the current value is not changed.

RING or *R*

Specifies the value of the CLIENT_RING attribute. It is the maximum ring number from which the affected NAM/VE program interfaces can be called. You can specify an integer in the range from 1 to 15. The initial value is 13. If you omit this parameter, the current value is not changed.

SYSTEM_PRIVILEGE or *SP*

Specifies the value of the CLIENT_SYSTEM_PRIVILEGE attribute.

TRUE

Indicates that the affected NAM/VE interfaces can be called only by code with a system privilege.

FALSE

Indicates that the affected NAM/VE interfaces can be called by any code.

The initial value is FALSE. If you omit this parameter, the current value is not changed.

NOTE

The SYSTEM_PRIVILEGE parameter is not operational for the current release of NOS/VE.

STATUS

Returns the completion status of this subcommand.

CHANGE_CLIENT_VALIDATION DEFS and CHAS Subcommand

Purpose Specifies whether NAM/VE should perform client validation on behalf of the server application.

Format **CHANGE_CLIENT_VALIDATION** or
CHACV
CAPABILITY=keyword or **name**
STATUS=status variable

Parameters **CAPABILITY** or **C**

Specifies the name of the CLIENT_VALIDATION attribute that must be present in the client's validation file entry. The keyword NONE can be specified to indicate that NAM/VE is not to perform client validation.

The initial value is NONE. This parameter is required.

STATUS

Returns the completion status of this subcommand.

Remarks See chapter 2 for additional information about client validation.

CHANGE_CONNECTION_PRIORITY DEFC and CHAC Subcommand

Purpose Changes the priority associated with the client application's connections.

Format

CHANGE_CONNECTION_PRIORITY or
CHACP
CONNECTION_PRIORITY=integer
STATUS=status variable

Parameters

CONNECTION_PRIORITY or **CP**

Specifies the value of the CONNECTION_PRIORITY attribute. You can specify an integer from 0 through 11. Zero is the lowest priority, and 11 is the highest priority.

This parameter is required.

STATUS

Returns the completion status of this subcommand.

Remarks

- If you use the CHANGE_CONNECTION_PRIORITY subcommand to change the connection priority of an active client, the new connection priority applies only to those connections created after the changed client definition is saved.
- Connection priority has no effect on NAM/VE processing of connection data. However, connection priority does affect CDCNET processing of connection data.
- The NAM/VE priority values from 0 through 11 map into the priority values supported by CDCNET as follows:

NAM/VE Connection Priority	CDCNET
0 - 4	Batch
5 - 8	Interactive
9 - 11	Realtime

- If an application has a requirement for a very low transfer delay, you should specify a value from 9 to 11 in the CONNECTION_PRIORITY parameter of the CHANGE_CONNECTION_PRIORITY subcommand. If an application transfers relatively small amounts of data and has a requirement for a low transfer delay, you should specify a value from 5 to 8 in the CONNECTION_PRIORITY parameter. If an application transfers large amounts of data, you should specify a value from 0 to 4 in the CONNECTION_PRIORITY parameter.

CHANGE_CONNECTION_PRIORITY DEFS and CHAS Subcommand

Purpose Changes the priority associated with the server application's connections.

Format

CHANGE_CONNECTION_PRIORITY or
CHACP
CONNECTION_PRIORITY=integer
STATUS=status variable

Parameters

CONNECTION_PRIORITY or **CP**

Specifies the value of the CONNECTION_PRIORITY attribute. You can specify an integer from 0 through 11. Zero is the lowest priority, and 11 is the highest priority.

This parameter is required.

STATUS

Returns the completion status of this subcommand.

Remarks

- If you use the CHANGE_CONNECTION_PRIORITY subcommand to change the connection priority of an active server, the new connection priority applies only to those connections created after the changed server definition is saved.
- Connection priority has no effect on NAM/VE processing of connection data. However, connection priority does affect CDCNET processing of connection data.
- The NAM/VE priority values from 0 through 11 map into the priority values supported by CDCNET as follows:

NAM/VE Connection Priority	CDCNET
0 - 4	Batch
5 - 8	Interactive
9 - 11	Realtime

- If an application has a requirement for a very low transfer delay, you should specify a value from 9 to 11 in the CONNECTION_PRIORITY parameter of the CHANGE_CONNECTION_PRIORITY subcommand. If an application transfers relatively small amounts of data and has a requirement for a low transfer delay, you should specify a value from 5 to 8 in the CONNECTION_PRIORITY parameter. If an application transfers large amounts of data, you should specify a value from 0 to 4 in the CONNECTION_PRIORITY parameter.

CHANGE_MAXIMUM_CONNECTIONS DEFC and CHAC Subcommand

Purpose Changes the maximum number of concurrent connections that can be established by all instances of the client application.

Format

CHANGE_MAXIMUM_CONNECTIONS or
CHAMC
MAXIMUM_CONNECTIONS=integer
STATUS=status variable

Parameters

MAXIMUM_CONNECTIONS or MC

Specifies the value of the MAXIMUM_CONNECTIONS attribute. The value indicates the total number of concurrent connections allowed to all instances of the client application. You can specify an integer in the range from 1 to 65,535. The initial value is 65,535. This parameter is required.

STATUS

Returns the completion status of this subcommand.

Remarks If you use the CHANGE_MAXIMUM_CONNECTIONS subcommand to change the maximum connection limit for the client to a value lower than the client's active count, the following occurs:

- The utility accepts no new connections until the active connection count is less than the new maximum connection limit.
- Excess connections are not terminated.

CHANGE_MAXIMUM_CONNECTIONS DEFS and CHAS Subcommand

Purpose Changes the maximum number of concurrent connections that can be established by all instances of the server application.

Format

CHANGE_MAXIMUM_CONNECTIONS or
CHAMC
MAXIMUM_CONNECTIONS=integer
STATUS=status variable

Parameters

MAXIMUM_CONNECTIONS or **MC**

Specifies the value of the MAXIMUM_CONNECTIONS attribute. The value indicates the total number of concurrent connections allowed to all instances of the server application. You can specify an integer in the range from 1 to 65,535. The initial value is 65,535. This parameter is required.

STATUS

Returns the completion status of this subcommand.

Remarks

- For a server application that is self-initiated, a request to initiate a new instance of the server application is rejected when enough instances to support the total number of allowed connections are already active.
- If you use the CHANGE_MAXIMUM_CONNECTIONS subcommand to change the maximum connection limit for the server to a value lower than the server's active count, the following occurs:
  - The utility accepts no new connections until the active connection count is less than the new maximum connection limit.
  - Excess connections are not terminated.

CHANGE_SERVER
MANNA Subcommand

Purpose Initiates a CHANGE_SERVER subutility session that changes the definition of an existing server application.

Format

CHANGE_SERVER or
CHAS
SERVER=name
STATUS=status variable

Parameters

SERVER or **S**

Specifies the name of the server application whose definition is to be changed. This parameter is required.

STATUS

Returns the completion status of this subcommand.

Remarks

- The CHANGE_SERVER subutility prompt is:

  ```
  chas/
  ```

- While you are in a CHANGE_SERVER subutility session, you can use the following subcommands to change the application's attributes:
  - ADD_CLIENT_ADDRESS
  - ADD_SERVER_MANAGED_TITLE
  - ADD_TITLE
  - CHANGE_ACCEPT_CONNECTION
  - CHANGE_APPLICATION_IDENTIFIER
  - CHANGE_CLIENT_VALIDATION
  - CHANGE_CONNECTION_PRIORITY
  - CHANGE_MAXIMUM_CONNECTIONS
  - CHANGE_SERVER_JOB
  - CHANGE_SERVER_VALIDATION
  - DELETE_CLIENT_ADDRESS
  - DELETE_SERVER_MANAGED_TITLE
  - DELETE_TITLE

- When you end the CHANGE_SERVER subutility with the END_CHANGE_SERVER subcommand, you have the option of either making the changes permanent or discarding the changes you have made.

CHANGE_SERVER_JOB
DEFS and CHAS Subcommand

Purpose Changes the contents or attributes of the job submitted by NAM/VE when NAM/VE initiates an instance of the server application. This subcommand can be used only for server applications initiated by NAM/VE.

Format

CHANGE_SERVER_JOB or
CHASJ
JOB=file
VALIDATION_SOURCE=keyword
MAXIMUM_CONNECTIONS=integer
INCLUDE_COMMANDS_UNTIL=string
STATUS=status variable

Parameters

JOB or *J*

Specifies the value of the SERVER_JOB attribute (the name of the file that contains the server job). The file you specify contains SCL commands that execute an instance of the server application. The initial value is a null file (no list of SCL commands). If you omit this parameter, the current value is not changed.

VALIDATION_SOURCE or *VS*

Specifies the value of the SERVER_JOB_VALIDATION_SOURCE attribute. The following keywords can be specified:

CLIENT or C

Indicates that NAM/VE is to generate a LOGIN command from information supplied by the client. (Information is obtained from the client as specified by the CLIENT_INFORMATION_SOURCE attribute.)

SERVER or S

Indicates that the first command in the list of SCL commands specified by the SERVER_JOB attribute must be a valid LOGIN command.

The initial value is SERVER. If you omit this parameter, the current value is not changed.

MAXIMUM_CONNECTIONS or *MC*

Specifies the value of the SERVER_JOB_MAXIMUM_CONNECTIONS attribute. This value indicates the number of concurrent connections supported by an instance of the server application. You can specify an integer from 1 to 65,535. The initial value is 65,535. If you omit this parameter, the current value is not changed.

INCLUDE_COMMANDS_UNTIL or *ICU*

Specifies the value of the INCLUDE_COMMANDS_UNTIL attribute. If the file specified by the JOB parameter is $COMMAND, the commands that are to make up the server job are read from the current command file until a line consisting solely of the string specified by the INCLUDE_COMMANDS_UNTIL parameter is encountered.

STATUS

Returns the completion status of this subcommand.

Remarks

- The user name and family name specified on the LOGIN command in the server application job is used to establish validation limits for the server job. Any prologs defined for that user are executed as part of job startup.
- The server job commands are maintained in the catalog $SYSTEM.NETWORK.APPLICATION.JOB. The server name is the name of the file.

CHANGE_SERVER_VALIDATION DEFS and CHAS Subcommand

Purpose Changes the server validation that NAM/VE performs to verify that the application is authorized to execute the functions of the server application.

Format

CHANGE_SERVER_VALIDATION or
CHASV
CAPABILITY=keyword or *name*
RING=integer
SYSTEM_PRIVILEGE=boolean
STATUS=status variable

Parameters *CAPABILITY* or *C*

Specifies the name of the SERVER_CAPABILITY attribute that must be in the calling user's validation file entry in order to call the affected NAM/VE program interfaces. The keyword NONE can be specified to indicate that no capability is to be validated.

The initial value is NONE. If you omit this parameter, the current value is not changed.

RING or *R*

Specifies the value of the SERVER_RING attribute. It is the maximum ring number from which the affected NAM/VE program interfaces can be called. You can specify an integer in the range from 1 to 15. The initial value is 13. If you omit this parameter, the current value is not changed.

SYSTEM_PRIVILEGE or *SP*

Specifies the value of the SERVER_SYSTEM_PRIVILEGE attribute.

TRUE

Indicates that the affected NAM/VE interfaces can be called only by code with a system privilege.

FALSE

Indicates that the affected NAM/VE interfaces can be called by any code.

The initial value is FALSE. If you omit this parameter, the current value is not changed.

NOTE

The SYSTEM_PRIVILEGE parameter is not operational for the current release of NOS/VE.

STATUS

Returns the completion status of this subcommand.

DEACTIVATE_CLIENT
MANNA Subcommand

Purpose Deactivates an active client application.

Format

DEACTIVATE_CLIENT or
DEAC
CLIENT=name
TERMINATE_ACTIVE_CONNECTIONS=boolean
STATUS=status variable

Parameters

CLIENT or **C**

Specifies the name of the client application to be deactivated. This parameter is required.

TERMINATE_ACTIVE_CONNECTIONS or *TAC*

Specifies whether to terminate active connections immediately or allow them to terminate normally.

TRUE

Indicates that active connections are to be terminated immediately.

FALSE

Indicates that active connections are to be allowed to continue until they terminate normally.

The default is FALSE.

STATUS

Returns the completion status of this subcommand.

Remarks Following entry of this subcommand, no new connections can be established by the client application.

DEACTIVATE_SERVER MANNA Subcommand

Purpose Deactivates an active server application.

Format

DEACTIVATE_SERVER or
DEAS
SERVER=name
TERMINATE_ACTIVE_CONNECTIONS=boolean
STATUS=status variable

Parameters

SERVER or **S**

Specifies the name of the server application to be deactivated. This parameter is required.

TERMINATE_ACTIVE_CONNECTIONS or *TAC*

Specifies whether to terminate active connections immediately or allow them to terminate normally.

TRUE

Indicates that active connections are to be terminated immediately.

FALSE

Indicates that active connections are to be allowed to continue until they terminate normally.

The default is FALSE.

STATUS

Returns the completion status of this subcommand.

Remarks Following entry of this subcommand, no new connections can be established to the server application.

DEFINE_CLIENT
MANNA Subcommand

Purpose Initiates a DEFINE_CLIENT subutility session which defines a client application.

Format

DEFINE_CLIENT or
DEFC
CLIENT=name
PROTOCOL=keyword
STATUS=status variable

Parameters

CLIENT or **C**

Specifies the name of the client application to be defined. This parameter is required.

PROTOCOL or **P**

Specifies the protocol used by the application.

CDNA_SESSION or CS

The CS protocol is used by applications that communicate over host-to-host connections.

CDNA_VIRTUAL_TERMINAL or CVT

The CVT protocol is used by applications that communicate with CDCNET-connected terminals.

This parameter is required.

STATUS

Returns the completion status of this subcommand.

Remarks

- When you initiate a DEFINE_CLIENT subutility session, the subutility checks the list of defined applications to make sure that no server or client application with the same name exists. When you end the subutility, the new client application definition is added to the list of defined applications.
- The DEFINE_CLIENT subutility prompt is:

```
defc/
```

- While you are in DEFINE_CLIENT subutility session, you can use the following subcommands to change the application's attributes:
  - CHANGE_APPLICATION_IDENTIFIER
  - CHANGE_CLIENT_VALIDATION
  - CHANGE_CONNECTION_PRIORITY
  - CHANGE_MAXIMUM_CONNECTIONS
- When you end the DEFINE_CLIENT subutility with the END_DEFINE_CLIENT subcommand, you have the option of making the changes permanent or discarding the changes you have made.
- The defined client application is inactive until you activate it with the ACTIVATE_CLIENT subcommand.

DEFINE_SERVER
MANNA Subcommand

Purpose Initiates a DEFINE_SERVER subutility session which defines a server application.

Format

DEFINE_SERVER or
DEFS
SERVER=name
PROTOCOL=keyword
NAM_INITIATED=boolean
STATUS=status variable

Parameters

SERVER or **S**

Specifies the name of the server application to be defined. This parameter is required.

PROTOCOL or **P**

Specifies the protocol used by the application.

CDNA_SESSION or CS

The CS protocol is used by applications that communicate over host-to-host connections.

CDNA_VIRTUAL_TERMINAL or CVT

The CVT protocol is used by applications that communicate with CDCNET-connected terminals.

This parameter is required.

NAM_INITIATED or NI

Specifies the value of the NAM_INITIATED attribute.

TRUE

Indicates that NAM/VE is to initiate an instance of the server application when it is necessary to service a connection request from a client application.

FALSE

Indicates that an application-specific mechanism (user-intitiated) is to initiate an instance of the server application.

The default is TRUE.

STATUS

Returns the completion status of this subcommand.

Remarks

- When you initiate a DEFINE_SERVER subutility session, the subutility checks the list of defined applications to make sure that no server or client application with the same name exists. When you end the subutility, the new server application definition is added to the list of defined applications.

- The DEFINE_SERVER subutility prompt is:

```
defs/
```

- While you are in the DEFINE_SERVER subutility session, you can use the following subcommands to change the application's attributes:
  - ADD_CLIENT_ADDRESS
  - ADD_SERVER_MANAGED_TITLE
  - ADD_TITLE
  - CHANGE_ACCEPT_CONNECTION
  - CHANGE_APPLICATION_IDENTIFIER
  - CHANGE_CLIENT_VALIDATION
  - CHANGE_CONNECTION_PRIORITY
  - CHANGE_MAXIMUM_CONNECTIONS
  - CHANGE_SERVER_JOB
  - CHANGE_SERVER_VALIDATION
  - DELETE_CLIENT_ADDRESS
  - DELETE_SERVER_MANAGED_TITLE
  - DELETE_TITLE

- When you end the DEFINE_SERVER subutility with the END_DEFINE_SERVER submcommand, you have the option of either making the changes permanent or discarding the changes you have made.

- The defined server application is inactive until you activate it with the ACTIVATE_SERVER subcommand.

DELETE_CLIENT
MANNA Subcommand

Purpose Deletes the definition of a client application.

Format

DELETE_CLIENT or
DELC
CLIENT=name
STATUS=status variable

Parameters

CLIENT or **C**

Specifies the name of the client application whose definition is to be deleted. This parameter is required.

STATUS

Returns the completion status of this subcommand.

Remarks Before you can delete a client application definition, the client application must be inactive and have no active connections. To deactivate a client application, you specify TRUE on the TERMINATE_ACTIVE_CONNECTIONS parameter of the DEACTIVATE_CLIENT subcommand.

DELETE_CLIENT_ADDRESS DEFS and CHAS Subcommand

Purpose Specifies a network address to be deleted from the list of addresses from which client applications are authorized to connect to the server application.

Format

DELETE_CLIENT_ADDRESS or
DELCA
NETWORK_IDENTIFIER=keyword or *integer*
SYSTEM_IDENTIFIER=keyword or *integer*
APPLICATION_IDENTIFIER=keyword or *integer*
STATUS=status variable

Parameters

NETWORK_IDENTIFIER or *NI*

Specifies the network identifier of the address to be deleted. The value specified can be a hexadecimal network identifier (integer) in the range from 1 to 0FFFFFFFF. The keyword ALL can be specified to indicate that all network indentifiers are specified in the network address to be deleted.

The default is ALL.

SYSTEM_IDENTIFIER or *SI*

Specifies the system identifier of the address to be deleted. The value specified can be a hexadecimal system identifier (integer) in the range from 1 to 0FFFFFFFFFFFF. The following keyword options are available:

ALL

Indicates that all system identifiers are specified in the network address to be deleted.

NOSVE

Indicates that the system identifier of any NOS/VE system is specified in the network address to be deleted.

CDCNET

Indicates that the system identifier of any CDCNET DI is specified in the network address to be deleted.

The default is ALL.

APPLICATION_IDENTIFIER or *AI*

Specifies the application identifier of the address to be deleted. The value specified can be an integer in the range from 2000 to 3000. The keyword ALL can be specified to indicate that all application identifiers are specified in the network address to be deleted.

The default is ALL.

STATUS

Returns the completion status of this subcommand.

Remarks The network address specified must match exactly an address that was previously placed in the list of authorized addresses by the ADD_CLIENT_ADDRESS subcommand. Refer to chapter 2 for more information on validating connection requests based on a client address.

DELETE_SERVER
MANNA Subcommand

Purpose Deletes the definition of a server application.

Format

DELETE_SERVER or
DELS
SERVER=name
STATUS=status variable

Parameters

SERVER or **S**

Specifies the name of the server application whose definition is to be deleted. This parameter is required.

STATUS

Returns the completion status of this subcommand.

Remarks Before you can delete a server application definition, the server application must be inactive and have no active connections or attached server jobs. To deactivate a server application, you specify TRUE on the TERMINATE_ACTIVE_CONNECTIONS parameter of the DEACTIVATE_SERVER subcommand. It may be necessary to use the TERMINATE_JOB SCL command to detach any active server application jobs. See the NOS/VE System Usage manual for more information about the TERMINATE_JOB command.

DELETE_SERVER_MANAGED_TITLE DEFS and CHAS Subcommand

Purpose Deletes one or more title patterns from the list of title patterns that defines the set of server titles the server application can manage.

Format

DELETE_SERVER_MANAGED_TITLE or
DELETE_SERVER_MANAGED_TITLES or
DELSMT
TITLE_PATTERN=list of name or **string**
STATUS=status variable

Parameters

TITLE_PATTERN or **TITLE_PATTERNS** or **TP**

Specifies title patterns to be deleted from the list of title patterns. A title pattern can be a name, a string, or a list of names and strings. For more information on title patterns, see chapter 2.

This parameter is required.

STATUS

Returns the completion status of this subcommand.

Remarks The ADD_SERVER_MANAGED_TITLE subcommand adds title patterns to the list of title patterns for the server application.

DELETE_TITLE
DEFS and CHAS Subcommand

Purpose Deletes titles from the list of titles to be registered in the network directory for the server application.

Format

DELETE_TITLE or
DELETE_TITLES or
DELT
TITLE=list of name or **string**
STATUS=status variable

Parameters

TITLE or **TITLES** or **T**

Specifies one or more titles to be deleted from the list of titles for the server application. A title can be a name, a string, or a list of names and strings. For more information on titles, see chapter 2.

This parameter is required.

STATUS

Returns the completion status of this subcommand.

Remarks The ADD_TITLE subcommand adds titles to the list of titles for the server application.

DISPLAY_APPLICATION_STATUS MANNA Subcommand

Purpose Displays status information for one or more defined server and client applications.

Format

DISPLAY_APPLICATION_STATUS or
DISAS
APPLICATION=keyword or *list of name*
OUTPUT=file
STATUS=status variable

Parameters

APPLICATION or *APPLICATIONS* or *A*

Specifies the names of one or more applications for which status is to be displayed. The keyword ALL can be specified to display the status of all applications.

The default value is ALL.

OUTPUT or *O*

Specifies the file on which the displayed information is to be written. The default is $OUTPUT.

STATUS

Returns the completion status of this subcommand.

Remarks

- The status information displayed includes:
  - an indication of whether the application is active or inactive,
  - the number of active connections for the application, and
  - the names of jobs in which the application is active.
- NAM/VE must be active in order to display status information.

Examples The following DISPLAY_APPLICATION_STATUS subcommand generates the output display shown below:

```
mna/display_application_status osa$timesharing


Server:                          OSA$TIMESHARING
  Status:                        active
  Active_server_connections:     3
  Server_added_titles:           None
  Active_server_jobs:            0
```

DISPLAY_CLIENT_ATTRIBUTE
MANNA, DEFC, and CHAC Subcommand

Purpose Displays attribute values for defined client applications.

Format

DISPLAY_CLIENT_ATTRIBUTE or
DISPLAY_CLIENT_ATTRIBUTES or
DISCA
CLIENT=keyword or *list of name*
DISPLAY_OPTION=list of keyword
OUTPUT=file
STATUS=status variable

Parameters

CLIENT or *C*

Specifies the names of one or more client applications for which attributes are to be displayed. The keyword ALL can be specified to display the attributes of all client applications.

The default value is ALL if used outside of the DEFINE_CLIENT or CHANGE_CLIENT subutility. When used inside of the DEFINE_CLIENT or CHANGE_CLIENT subutility, the client application currently being defined or changed is the default.

DISPLAY_OPTION or *DISPLAY_OPTIONS* or *DO*

Specifies the attribute values to be displayed. For a description of client application attributes, see table 3-3.

Options are one or more of the following keywords:

ALL
APPLICATION_IDENTIFIER or AI
CLIENT_CAPABILITY or CC
CLIENT_RING or CR
CLIENT_STATUS or CS
CLIENT_SYSTEM_PRIVILEGE or CSP
CONNECTION_PRIORITY or CP
MAXIMUM_CONNECTIONS or MC
PROTOCOL or P

The attribute value ALL displays all attribute values. The default is ALL.

OUTPUT or *O*

Specifies the file on which the displayed information is to be written. The default is $OUTPUT.

STATUS

Returns the completion status of this subcommand.

Examples The following DISPLAY_CLIENT_ATTRIBUTE subcommand generates the output display shown below:

```
mna/display_client_attributes client=osa$file_transfer_client


Client:                          OSA$FILE_TRANSFER_CLIENT
  Status:                        active
  Protocol:                      cdna_session
  Application_Identifier:        2001
  Maximum_connections:           40
  Connection_priority:           0
  Client_capability:             None
  Client_ring:                   13
  Client_system_privilege:       TRUE
```

DISPLAY_CLIENT_STATUS
MANNA Subcommand

Purpose Displays status information for defined client applications.

Format

DISPLAY_CLIENT_STATUS or
DISCS
CLIENT=keyword or *list of name*
OUTPUT=file
STATUS=status variable

Parameters

CLIENT or *C*

Specifies the names of one or more client applications for which status is to be displayed. The keyword ALL can be specified to display the status of all client applications.

The default value is ALL.

OUTPUT or *O*

Specifies the file on which the displayed information is to be written. The default is $OUTPUT.

STATUS

Returns the completion status of this subcommand.

Remarks

- The status information displayed includes:
  - an indication of whether the client application is active or inactive and
  - the number of active connections for the client application.
- NAM/VE must be active in order to display status information.

Examples The following DISPLAY_CLIENT_STATUS subcommand generates the output display shown below:

```
mna/display_client_status client=osa$file_transfer_client


Client:                          OSA$FILE_TRANSFER_CLIENT
  Status:                        active
  Active_client_connections:     0
```

DISPLAY_SERVER_ATTRIBUTE MANNA, DEFS, and CHAS Subcommand

Purpose Displays attribute values for defined server applications.

Format

DISPLAY_SERVER_ATTRIBUTE or
DISPLAY_SERVER_ATTRIBUTES or
DISSA
SERVER=keyword or *list of name*
DISPLAY_OPTION=list of keyword
OUTPUT=file
STATUS=status variable

Parameters

SERVER or *S*

Specifies the names of one or more server applications for which attributes are to be displayed. The keyword ALL can be specified to display the attributes of all server applications.

The default value is ALL if used outside of the DEFINE_SERVER or CHANGE_SERVER subutility. The server application currently being defined or changed is the default if you enter the subcommand within the DEFINE_SERVER or CHANGE_SERVER subutility.

DISPLAY_OPTION or *DISPLAY_OPTIONS* or *DO*

Specifies the attribute values to be displayed. For a description of server application attributes, see table 3-4. Options are one or more of the following keywords:

ACCEPT_CONNECTION or AC
ALL
APPLICATION_IDENTIFIER or AI
CLIENT_ADDRESS or CLIENT_ADDRESSES or CA
CLIENT_INFO_SOURCE or CIS
CLIENT_VALIDATION or CV
CONNECTION_PRIORITY or CP
MAXIMUM_CONNECTIONS or MC
NAM_INITIATED or NI
PROTOCOL or P
SERVER_CAPABILITY or SC
SERVER_JOB or SJ
SERVER_JOB_MAXIMUM_CONNECTIONS or SJMC
SERVER_JOB_VALIDATION_SOURCE or SJVS
SERVER_MANAGED_TITLE or SERVER_MANAGED_TITLES or SMT
SERVER_RING or SR
SERVER_SYSTEM_PRIVILEGE or SSP
SERVER_STATUS or SS
TITLE or TITLES or T
TITLE_ATTRIBUTE or TITLE_ATTRIBUTES or TA

The attribute value ALL displays all attribute values. The attribute value TITLE_ATTRIBUTES displays the TITLE_PRIORITY, BROADCAST_REGISTRATION, and TITLE_DATA attributes for each title. The default is ALL.

OUTPUT or *O*

Specifies the file on which the displayed information is to be written. The default is $OUTPUT.

STATUS

Returns the completion status of this subcommand.

Examples

The following DISPLAY_SERVER_ATTRIBUTE subcommand generates the output display shown below:

```
mna/display_server_attributes server=osa$file_transfer_server


Server:                          OSA$FILE_TRANSFER_SERVER
  Status:                        active
  Protocol:                      cdna_session
  Application_Identifier:        Variable
  Maximum_connections:           40
  Connection_priority:           0
  Server_capability:             None
  Server_ring:                   13
  Server_system_privilege:       TRUE
  Title:                         PTFS$NVEDEV
    Broadcast_registration:      FALSE
    Title_priority:              1
    Title_data:                  None
  Title:                         PTFS$TESTVE830631
    Broadcast_registration:      FALSE
    Title_priority:              1
    Title_data:                  None
  Title:                         PTFS$VN1
    Broadcast_registration:      FALSE
    Title_priority:              1
    Title_data:                  None
  Server_managed_titles:         None
  Accept_connection:             FALSE
  Client_validation:             None
  Client_information_source:     dialog
  Client_addresses:              all addresses
  NAM_Initiated:                 FALSE
```

DISPLAY_SERVER_STATUS
MANNA Subcommand

Purpose Displays status information for defined server applications.

Format

DISPLAY_SERVER_STATUS or
DISSS
SERVER=keyword or *list of name*
OUTPUT=file
STATUS=status variable

Parameters

SERVER or *S*

Specifies the names of one or more server applications for which status is to be displayed. The keyword ALL can be specified to display the status of all server applications.

The default value is ALL.

OUTPUT or *O*

Specifies the file on which the displayed information is to be written. The default is $OUTPUT.

STATUS

Returns the completion status of this subcommand.

Remarks

- The status information displayed by this subcommand includes:
  - an indication of whether the server application is active or inactive,
  - the names of jobs in which the application is active (server application jobs),
  - a list of currently active server managed titles, and
  - the number of active connections for each server job.
- NAM/VE must be active in order to display status information.

Examples The following DISPLAY_SERVER_STATUS subcommand generates the output display shown below:

```
mna/display_server_status server=osa$file_transfer_server


Server:                              OSA$FILE_TRANSFER_SERVER
  Status:                            active
  Active_server_connections:         0
  Server_added_titles:               None
  Active_server_jobs:                1
    Job name              Connections
  $0830_0631_AAA_0000           0
```

END_CHANGE_CLIENT
CHAC Subcommand

Purpose Ends a CHANGE_CLIENT subutility session.

Format

END_CHANGE_CLIENT or
ENDCC or
QUIT or
QUI
SAVE_DEFINITION=boolean
STATUS=status variable

Parameters

SAVE_DEFINITION or *SD*

Specifies whether to save or discard the changes you have made to the client definition.

TRUE

Indicates that the client definition is to be saved.

FALSE

Indicates that the client definition is to be discarded.

The default is TRUE.

STATUS

Returns the completion status of this subcommand.

Remarks

- When the client application definition is to be saved, the definition is checked for errors. The CHANGE_CLIENT subutility session does not end if errors are found. The subutility issues a message documenting the problem. (You can enter subcommands to correct the problem.) A definition in error is never saved.
- The client definition is recorded in the $SYSTEM.NETWORK.APPLICATION.DEFINITIONS file when this subcommand executes successfully. The definition remains in effect until explicitly deleted.

END_CHANGE_SERVER
CHAS Subcommand

Purpose Ends a CHANGE_SERVER subutility session.

Format

END_CHANGE_SERVER or
ENDCS or
QUIT or
QUI
SAVE_DEFINITION=boolean
STATUS=status variable

Parameters *SAVE_DEFINITION* or *SD*

Specifies whether to save or discard the changes you have made to the server definition.

TRUE

Indicates that the server definition is to be saved.

FALSE

Indicates that the server definition is to be discarded.

The default is TRUE.

STATUS

Returns the completion status of this subcommand.

Remarks

- When the server application definition is to be saved, the definition is checked for errors. The CHANGE_SERVER subutility session does not end if errors are found. The subutility issues a message documenting the problem. (You can enter subcommands to correct the problem.) A definition in error is never saved.
- The server definition is recorded in the $SYSTEM.NETWORK.APPLICATION.DEFINITIONS file when this subcommand executes successfully. The definition remains in effect until explicitly deleted.

END_DEFINE_CLIENT
DEFC Subcommand

Purpose Ends a DEFINE_CLIENT subutility session.

Format

END_DEFINE_CLIENT or
ENDDC or
QUIT or
QUI
SAVE_DEFINITION=boolean
STATUS=status variable

Parameters

SAVE_DEFINITION or *SD*

Specifies whether to save or discard the changes you have made to the client definition.

TRUE

Indicates that the client definition is to be saved.

FALSE

Indicates that the client definition is to be discarded.

The default is TRUE.

STATUS

Returns the completion status of this subcommand.

Remarks

- When the client application definition is to be saved, the definition is checked for errors. The DEFINE_CLIENT subutility session does not end if errors are found. The subutility issues a message documenting the problem. (You can enter subcommands to correct the problem.) A definition in error is never saved.
- The client definition is recorded in the $SYSTEM.NETWORK.APPLICATION.DEFINITIONS file when this subcommand executes successfully. The definition remains in effect until explicitly deleted.

END_DEFINE_SERVER
DEFS Subcommand

Purpose Ends a DEFINE_SERVER subutility session.

Format

END_DEFINE_SERVER or
ENDDS or
QUIT or
QUI
SAVE_DEFINITION=boolean
STATUS=status variable

Parameters

SAVE_DEFINITION or *SD*

Specifies whether to save or discard the changes you have made to the server definition.

TRUE

Indicates that the server definition is to be saved.

FALSE

Indicates that the server definition is to be discarded.

The default is TRUE.

STATUS

Returns the completion status of this subcommand.

Remarks

- When the server application definition is to be saved, the definition is checked for errors. The DEFINE_SERVER subutility session does not end if errors are found. The subutility issues a message documenting the problem. (You can enter subcommands to correct the problem.) A definition in error is never saved.
- The server definition is recorded in the $SYSTEM.NETWORK.APPLICATION.DEFINITIONS file when this subcommand executes successfully. The definition remains in effect until explicitly deleted.

GENERATE_APPLICATION_DEFINITION MANNA Subcommand

Purpose Generates a set of MANNA subcommands that defines one or more existing client and server applications. You can reenter this set of subcommands in the MANNA utility to redefine the application.

Format **GENERATE_APPLICATION_DEFINITION** or **GENAD**

APPLICATION=keyword or *list of name*
OUTPUT=file
STATUS=status variable

Parameters *APPLICATION* or *A*

Specifies the names of one or more applications for which application definitions are to be generated. The keyword ALL can be specified to generate the definitions of all applications.

The default value is ALL.

OUTPUT or *O*

Specifies the file on which the generated definition subcommands are to be written. The default is $OUTPUT.

STATUS

Returns the completion status of this subcommand.

Remarks You must specify an output file in order to edit the definition.

GENERATE_CLIENT_DEFINITION
MANNA, DEFC, and CHAC Subcommand

Purpose Generates a set of MANNA subcommands that defines one or more existing client applications. You can reenter this set of subcommands in the MANNA utility to redefine the client application.

Format **GENERATE_CLIENT_DEFINITION** or
GENCD
CLIENT=keyword or *list of name*
OUTPUT=file
STATUS=status variable

Parameters *CLIENT* or *C*

Specifies the names of one or more client applications for which client application definitions are to be generated. The keyword ALL can be specified to generate the definitions of all client applications.

The default value is ALL if used outside of the DEFINE_CLIENT or CHANGE_CLIENT subutility. When used inside of the DEFINE_CLIENT or CHANGE_CLIENT subutility, the client definition currently being defined or changed is the default.

OUTPUT or *O*

Specifies the file on which the generated definition subcommands are to be written. The default is $OUTPUT.

STATUS

Returns the completion status of this subcommand.

Remarks You must specify an output file in order to edit the definition.

GENERATE_SERVER_DEFINITION
MANNA, DEFS, and CHAS Subcommand

Purpose Generates a set of MANNA subcommands that defines one or more existing server applications. You can reenter this set of subcommands in the MANNA utility to redefine the server application.

Format **GENERATE_SERVER_DEFINITION** or
GENSD
SERVER=keyword or *list of name*
OUTPUT=file
STATUS=status variable

Parameters *SERVER* or *S*

Specifies the names of one or more server applications for which server application definitions are to be generated. The keyword ALL can be specified to generate the definitions of all server applications.

The default value is ALL if used outside of the DEFINE_SERVER or CHANGE_SERVER subutility. When used inside of the DEFINE_SERVER or CHANGE_SERVER subutility, the server application currently being defined or changed is the default.

OUTPUT or *O*

Specifies the file on which the generated definition subcommands are to be written. The default is $OUTPUT.

STATUS

Returns the completion status of this subcommand.

Remarks You must specify an output file in order to edit the definition.

QUIT
MANNA Subcommand

Purpose Ends a MANNA utility session.

Format **QUIT** or
QUI

Parameters None.

Application Installation Commands 5

Application Installation Commands 5

This chapter describes the NOS/VE commands used to manage the system-defined network applications. The one exception is the DEFINE_NETWORK command that defines the Timesharing application. The DEFINE_NETWORK command is described in the NOS/VE Software Release Bulletin.

Generally, there are four commands/subcommands for each network application; one each to define, delete, activate, and deactivate the application. You can enter the define and delete subcommands from the MANAGE_APPLICATION_DEFINITIONS utility. These subcommands are described in alphabetical order following the MANAGE_APPLICATION_DEFINITIONS command. The activate and deactivate commands can be found later in this chapter in the Activate and Deactivate Commands section.

To bring up any of the system-defined applications, the application must first be defined, then activated, using the appropriate commands described in this chapter. The commands that activate each application are typically placed in the network activation epilog file ($SYSTEM.PROLOGS_AND_EPILOGS.NETWORK_ACTIVATION_EPILOG). The commands that define the applications are not placed in the network activation epilog file because the definitions of the applications are retained across deadstarts. The network activation epilog file is described in the NOS/VE System Performance and Maintenance manual, Volume 2.

MANAGE_APPLICATION_DEFINITIONS Utility

The MANAGE_APPLICATION_DEFINITIONS (MANAD) utility is an SCL command utility. It follows the rules for utilities given in the NOS/VE System Usage manual.

Except for QUIT, all of the utility subcommands have an optional STATUS parameter. Information on using the STATUS parameter can be found in the NOS/VE System Usage manual.

MANAGE_APPLICATION_DEFINITIONS

Purpose Initiates a MANAGE_APPLICATION_DEFINITIONS utility session.

Format

MANAGE_APPLICATION_DEFINITIONS or
MANAD
STATUS=status variable

Parameters *STATUS*

Returns the completion status of this command.

Remarks

- To use this utility, you must be validated by the system administrator. For more information, refer to the NOS/VE User Validation manual.
- This utility must be entered at the NOS/VE system console.
- The prompt for the MANAD utility is:

```
mad/
```

DEFINE_BTF

Purpose Defines the Batch Transfer Facility/VE (BTF/VE) network client application.

Format

DEFINE_BTF
STATUS=status variable

Parameters

STATUS

Returns the completion status of this subcommand.

Remarks

- The name of the client application created is OSA$BATCH_TRANSFER_CLIENT.
- The system task for BTF is activated by the SCF system task as an asynchronous task whenever output is transferred to an active batch device, or by the NTF system task as an asynchronous task whenever a file is transferred to an active batch stream.

DEFINE_BTFS

Purpose Defines the Batch Transfer Facility Server/VE (BTFS/VE) network application.

Format

DEFINE_BTFS
FAMILY_NAME=list of names
STATUS=status variable

Parameters

FAMILY_NAME or FAMILY_NAMES or FN

Defines all families on this mainframe that can be accessed by batch input. Up to 26 family names can be specified. This is a required parameter.

STATUS

Returns the completion status of this subcommand.

Remarks The name of the server application created is OSA$BATCH_TRANSFER_SERVER.

DEFINE_DESKTOP_ENVIRONMENT

Purpose Defines the Desktop/VE network client application.

Format

DEFINE_DESKTOP_ENVIRONMENT
STATUS=status variable

Parameters

STATUS

Returns the completion status of this subcommand.

Remarks The name of the client application created is DESKTOP_VE.

DEFINE_FTP

Purpose Defines the File Transfer Protocol (FTP) network client application.

Format

DEFINE_FTP
STATUS=status variable

Parameters *STATUS*

Returns the completion status of this subcommand.

Remarks The name of the client application created is OSA$FTP_CLIENT.

DEFINE_FTPS

Purpose Defines the File Transfer Protocol Server (FTPS) network application.

Format

DEFINE_FTPS
STATUS=status variable

Parameters

STATUS

Returns the completion status of this subcommand.

Remarks

- The name of the network application created is OSA$FTP_SERVER.
- Although this application operates as a server application, it is defined as a client application. See appendix C for more information about defining the File Transfer Protocol applications.

DEFINE_IPC_APPLICATIONS

Purpose Defines the Interprocess Communication (IPC) network client application.

Format

DEFINE_IPC_APPLICATIONS
STATUS=status variable

Parameters *STATUS*

Returns the completion status of this subcommand.

Remarks

- The name of the client application created is OSA$IPC_APPLICATIONS.
- The IPC client application is made up of the following server applications:
  - NFS/VE, which contains the bulk of Control Data's implementation of NFS.
  - Portmapper, which maps calls from client systems into registered server calls.
  - PCNFSD, which allows PCs running PC-NFS to be validated to use NFS/VE.

DEFINE_NTF

Purpose Defines the Network Transfer Facility (NTF) network client application.

Format

DEFINE_NTF
STATUS=status variable

Parameters *STATUS*

Returns the completion status of this subcommand.

Remarks The name of the client application created is OSA$NETWORK_TRANSFER_FAC_CLIENT.

DEFINE_OPENTF

Purpose Defines the Operate NTF (OPENTF) network client application.

Format

DEFINE_OPENTF
STATUS=status variable

Parameters

STATUS

Returns the completion status of this subcommand.

Remarks

- The name of the client application created is OSA$NTF_OPERATOR.
- The Operate NTF (OPENTF) application is initiated by a user that is validated for this privilege.

DEFINE_OPES

Purpose Defines the OPERATE_STATION utility (OPES) network client application.

Format

DEFINE_OPES
STATUS=status variable

Parameters

STATUS

Returns the completion status of this subcommand.

Remarks

- The name of the client application created is OSA$STATION_OPERATOR.
- The OPERATE_STATION utility is initiated by a user that is validated for this privilege.

DEFINE_PTF

Purpose Defines the Permanent File Transfer Facility (PTF) client and server applications.

Format

DEFINE_PTF
FAMILY_NAME=list of names
STATUS=status variable

Parameters

FAMILY_NAME or FAMILY_NAMES or FN

Defines all families on this mainframe that can be accessed by PTF. Up to 26 family names can be specified. This is a required parameter.

STATUS

Returns the completion status of this subcommand.

Remarks The name of the server application created is OSA$FILE_TRANSFER_SERVER. The name of the client application created is OSA$FILE_TRANSFER_CLIENT.

DEFINE_QTF

Purpose Defines the NAM/VE Queue File Transfer Facility (QTF) client application.

Format

DEFINE_QTF
STATUS=status variable

Parameters *STATUS*

Returns the completion status of this subcommand.

Remarks The name of the NAM/VE QTF client application created is OSA$QUEUE_TRANSFER_CLIENT.

DEFINE_QTFS

Purpose Defines the NAM/VE Queue File Transfer Facility Server (QTFS) application.

Format

DEFINE_QTFS
FAMILY_NAME=list of names
STATUS=status variable

Parameters

FAMILY_NAME or **FAMILY_NAMES** or **FN**

Defines all families that can be accessed by QTF. Up to 26 family names can be specified. This parameter is required.

STATUS

Returns the completion status of this subcommand.

Remarks The name of the NAM/VE QTF server application created is OSA$QUEUE_TRANSFER_SERVER.

DEFINE_SCF

Purpose Defines the Status and Control Facility/VE (SCF/VE) network client application.

Format

DEFINE_SCF
STATUS=status variable

Parameters

STATUS

Returns the completion status of this subcommand.

Remarks The name of the client application created is OSA$STATUS_CONTROL_FAC_CLIENT.

DEFINE_SCFS

Purpose Defines the Status and Control Facility Server/VE (SCFS/VE) network application.

Format

DEFINE_SCFS
APPLICATION_NAME=name
STATUS=status variable

Parameters

APPLICATION_NAME or *AN*

Specifies the SCF server application name. The default name is OSA$STATUS_CONTROL_FAC_SERVER.

STATUS

Returns the completion status of this subcommand.

DEFINE_SMTP

Purpose Defines the Simple Mail Transfer Protocol (SMTP) network client application.

Format **DEFINE_SMTP**
STATUS=status variable

Parameters *STATUS*

Returns the completion status of this subcommand.

Remarks The name of the client application created is OSA$SMTP_CLIENT.

DEFINE_SMTPS

Purpose Defines the Simple Mail Transfer Protocol (SMTP) network server application.

Format

DEFINE_SMTPS
STATUS=status variable

Parameters

STATUS

Returns the completion status of this subcommand.

Remarks

- The name of the network application created is OSA$SMTP_SERVER.
- Although this application operates as a server application, it is defined as a client application. See appendix C for more information about defining the Simple Mail Transfer Protocol applications.

DELETE_BTF

Purpose Deletes the definition of the Batch Transfer Facility/VE (BTF/VE) network client application.

Format

DELETE_BTF
TERMINATE_ACTIVE_CONNECTIONS=boolean
STATUS=status variable

Parameters

TERMINATE_ACTIVE_CONNECTIONS or *TAC*

Specifies whether to terminate active connections immediately or allow them to terminate normally.

TRUE

Indicates that active connections are to be terminated immediately.

FALSE

Indicates that active connections are to be allowed to continue until they terminate normally.

The default value is FALSE.

STATUS

Returns the completion status of this subcommand.

DELETE_BTFS

Purpose Deletes the definition of the Batch Transfer Facility Server/VE (BTFS/VE) network application.

NOTE

Before deleting the BTFS application definition, you must deactivate the associated system task using the DEACTIVATE_BTFS command at the system console.

Format

DELETE_BTFS
TERMINATE_ACTIVE_CONNECTIONS=boolean
STATUS=status variable

Parameters

TERMINATE_ACTIVE_CONNECTIONS or *TAC*

Specifies whether to terminate active connections immediately or allow them to terminate normally.

TRUE

Indicates that active connections are to be terminated immediately.

FALSE

Indicates that active connections are to be allowed to continue until they terminate normally.

The default value is FALSE.

STATUS

Returns the completion status of this subcommand.

Remarks The name of the server application deleted is OSA$BATCH_TRANSFER_SERVER.

DELETE_DESKTOP_ENVIRONMENT

Purpose Deletes the definition of the Desktop/VE network client application.

Format

DELETE_DESKTOP_ENVIRONMENT
TERMINATE_ACTIVE_CONNECTIONS=boolean
STATUS=status variable

Parameters

TERMINATE_ACTIVE_CONNECTIONS or *TAC*

Specifies whether to terminate active connections immediately or allow them to terminate normally.

TRUE

Indicates that active connections are to be terminated immediately.

FALSE

Indicates that active connections are to be allowed to continue until they terminate normally.

The default value is FALSE.

STATUS

Returns the completion status of this subcommand.

Remarks The name of the network client application deleted is DESKTOP_VE.

DELETE_FTP

Purpose Deletes the definition of the File Transfer Protocol (FTP) network client application.

Format

DELETE_FTP
TERMINATE_ACTIVE_CONNECTIONS=boolean
STATUS=status variable

Parameters

TERMINATE_ACTIVE_CONNECTIONS or *TAC*

Specifies whether to terminate active connections immediately or allow them to terminate normally.

TRUE

Indicates that active connections are to be terminated immediately.

FALSE

Indicates that active connections are to be allowed to continue until they terminate normally.

The default value is FALSE.

STATUS

Returns the completion status of this subcommand.

Remarks The name of the client application deleted is OSA$FTP_CLIENT.

DELETE_FTPS

Purpose Deletes the definition of the File Transfer Protocol Server (FTPS) application.

Format

DELETE_FTPS
TERMINATE_ACTIVE_CONNECTIONS=boolean
STATUS=status variable

Parameters

TERMINATE_ACTIVE_CONNECTIONS or *TAC*

Specifies whether to terminate active connections immediately or allow them to terminate normally.

TRUE

Indicates that active connections are to be terminated immediately.

FALSE

Indicates that active connections are to be allowed to continue until they terminate normally.

The default value is FALSE.

STATUS

Returns the completion status of this subcommand.

Remarks

- The name of the application deleted is OSA$FTP_SERVER.
- Although this application operates as a server application, it is defined as a client application. See appendix C for more information about defining the File Transfer Protocol applications.

DELETE_IPC_APPLICATIONS

Purpose Deletes the Interprocess Communication (IPC) network client application.

Format

DELETE_IPC_APPLICATIONS
STATUS=status variable

Parameters

STATUS

Returns the completion status of this subcommand.

Remarks The name of the client application deleted is OSA$IPC_APPLICATIONS.

DELETE_NTF

Purpose Deletes the definition of the Network Transfer Facility (NTF) network client application.

NOTE

Before deleting the NTF application definition, you must deactivate the associated system task using the DEACTIVATE_NTF command at the system console.

Format

DELETE_NTF
TERMINATE_ACTIVE_CONNECTIONS=boolean
STATUS=status variable

Parameters

TERMINATE_ACTIVE_CONNECTIONS or *TAC*

Specifies whether to terminate active connections immediately or allow them to terminate normally.

TRUE

Indicates that active connections are to be terminated immediately.

FALSE

Indicates that active connections are to be allowed to continue until they terminate normally.

The default value is FALSE.

STATUS

Returns the completion status of this subcommand.

Remarks The name of the client application deleted is OSA$NETWORK_TRANSFER_FAC_CLIENT.

DELETE_OPENTF

Purpose Deletes the definition of the Operate NTF (OPENTF) network client application.

Format

DELETE_OPENTF
TERMINATE_ACTIVE_CONNECTIONS=boolean
STATUS=status variable

Parameters

TERMINATE_ACTIVE_CONNECTIONS or *TAC*

Specifies whether to terminate active connections immediately or allow them to terminate normally.

TRUE

Indicates that active connections are to be terminated immediately.

FALSE

Indicates that active connections are to be allowed to continue until they terminate normally.

The default value is FALSE.

STATUS

Returns the completion status of this subcommand.

Remarks The name of the client application deleted is OSA$NTF_OPERATOR.

DELETE_OPES

Purpose Deletes the definition of the OPERATE_STATION utility (OPES) network client application.

Format

DELETE_OPES
TERMINATE_ACTIVE_CONNECTIONS=boolean
STATUS=status variable

Parameters

TERMINATE_ACTIVE_CONNECTIONS or *TAC*

Specifies whether to terminate active connections immediately or allow them to terminate normally.

TRUE

Indicates that active connections are to be terminated immediately.

FALSE

Indicates that active connections are to be allowed to continue until they terminate normally.

The default value is FALSE.

STATUS

Returns the completion status of this subcommand.

Remarks The name of the client application deleted is OSA$STATION_OPERATOR.

DELETE_PTF

Purpose Deletes the definition of the Permanent File Transfer Facility (PTF) network client and server applications.

Format

DELETE_PTF
TERMINATE_ACTIVE_CONNECTIONS=boolean
STATUS=status variable

Parameters

TERMINATE_ACTIVE_CONNECTIONS or *TAC*

Specifies whether to terminate active connections immediately or allow them to terminate normally.

TRUE

Indicates that active connections are to be terminated immediately.

FALSE

Indicates that active connections are to be allowed to continue until they terminate normally.

The default value is FALSE.

STATUS

Returns the completion status of this subcommand.

Remarks The name of the client application deleted is OSA$FILE_TRANSFER_CLIENT. The name of the server application deleted is OSA$FILE_TRANSFER_SERVER.

DELETE_QTF

Purpose Deletes the definition of the NAM/VE Queue File Transfer Facility (QTF) client application.

Format

DELETE_QTF
TERMINATE_ACTIVE_CONNECTIONS=boolean
STATUS=status variable

Parameters

TERMINATE_ACTIVE_CONNECTIONS or *TAC*

Specifies whether to terminate active connections immediately or allow them to terminate normally.

TRUE

Indicates that active connections are to be terminated immediately.

FALSE

Indicates that active connections are to be allowed to continue until they terminate normally.

The default value is FALSE.

STATUS

Returns the completion status of this subcommand.

Remarks The name of the NAM/VE QTF client application deleted is OSA$QUEUE_TRANSFER_CLIENT.

DELETE_QTFS

Purpose Deletes the definition of the NAM/VE Queue File Transfer Facility Server (QTFS) application.

Format

DELETE_QTFS
TERMINATE_ACTIVE_CONNECTIONS=boolean
STATUS=status variable

Parameters

TERMINATE_ACTIVE_CONNECTIONS or *TAC*

Specifies whether to terminate active connections immediately or allow them to terminate normally.

TRUE

Indicates that active connections are to be terminated immediately.

FALSE

Indicates that active connections are to be allowed to continue until they terminate normally.

The default value is FALSE.

STATUS

Returns the completion status of this subcommand.

Remarks The name of the NAM/VE QTF server application deleted is OSA$QUEUE_TRANSFER_SERVER.

DELETE_SCF

Purpose Deletes the definition of the Status and Control Facility/VE (SCF/VE) network client application.

NOTE

If the DELETE_SCF subcommand executes before the DEACTIVATE_SCF command executes, the system task terminates but is still defined.

Format

DELETE_SCF
TERMINATE_ACTIVE_CONNECTIONS=boolean
STATUS=status variable

Parameters

TERMINATE_ACTIVE_CONNECTIONS or *TAC*

Specifies whether to terminate active connections immediately or allow them to terminate normally.

TRUE

Indicates that active connections are to be terminated immediately.

FALSE

Indicates that active connections are to be allowed to continue until they terminate normally.

The default value is FALSE.

STATUS

Returns the completion status of this subcommand.

Remarks The name of the client application deleted is OSA$STATUS_CONTROL_FAC_CLIENT.

DELETE_SCFS

Purpose Deletes the definition of the Status and Control Facility Server/VE (SCFS/VE) network application.

NOTE

If the DELETE_SCFS subcommand executes before the DEACTIVATE_SCFS command executes, the system task terminates but is still defined.

Format

DELETE_SCFS
APPLICATION_NAME=name
TERMINATE_ACTIVE_CONNECTIONS=boolean
STATUS=status variable

Parameters

APPLICATION_NAME or *AN*

Specifies the SCF server application that is to be deleted. The default application name is OSA$STATUS_CONTROL_FAC_SERVER.

TERMINATE_ACTIVE_CONNECTIONS or *TAC*

Specifies whether to terminate active connections immediately or allow them to terminate normally.

TRUE

Indicates that active connections are to be terminated immediately.

FALSE

Indicates that active connections are to be allowed to continue until they terminate normally.

The default value is FALSE.

STATUS

Returns the completion status of this subcommand.

Remarks The name of the client application deleted is OSA$STATUS_CONTROL_FAC_SERVER.

DELETE_SMTP

Purpose Deletes the definition of the Simple Mail Transfer Protocol (SMTP) network client application.

Format

DELETE_SMTP
TERMINATE_ACTIVE_CONNECTIONS=boolean
STATUS=status variable

Parameters

TERMINATE_ACTIVE_CONNECTIONS or *TAC*

Specifies whether to terminate active connections immediately or allow them to terminate normally.

TRUE

Indicates that active connections are to be terminated immediately.

FALSE

Indicates that active connections are to be allowed to continue until they terminate normally.

The default value is FALSE.

STATUS

Returns the completion status of this subcommand.

Remarks The name of the application deleted is OSA$SMTP_CLIENT.

DELETE_SMTPS

Purpose Deletes the definition of the Simple Mail Transfer Protocol (SMTP) server application.

Format

DELETE_SMTPS
TERMINATE_ACTIVE_CONNECTIONS=boolean
STATUS=status variable

Parameters

TERMINATE_ACTIVE_CONNECTIONS or *TAC*

Specifies whether to terminate active connections immediately or allow them to terminate normally.

TRUE

Indicates that active connections are to be terminated immediately.

FALSE

Indicates that active connections are to be allowed to continue until they terminate normally.

The default value is FALSE.

STATUS

Returns the completion status of this subcommand.

Remarks

- The name of the application deleted is OSA$SMTP_SERVER.
- Although this application operates as a server application, it is defined as a client application. See appendix C for more information about defining the Simple Mail Transfer Protocol applications.

QUIT

Purpose Ends a MANAGE_APPLICATION_DEFINITIONS utility session.

Format **QUIT** or
QUI

Parameters None.

Activate and Deactivate Commands

The commands that activate each application are typically placed in the network activation epilog file $SYSTEM.PROLOGS_AND_EPILOGS.NETWORK_ACTIVATION_EPILOG. If it is necessary to deactivate an application, this can be done by using the appropriate deactivate command. The application can then be reactivated using the corresponding activate command. The activate and deactivate commands must be entered at the NOS/VE system console.

ACTIVATE_BTFS

Purpose Defines and activates a system task for the Batch Transfer Facility Server/VE (BTFS/VE) application. If the BTFS system task definition already exists when this command is entered, that task definition is removed before a new task is defined and activated.

Format

ACTIVATE_BTFS
LOAD_MAP=file
STATUS=status variable

Parameters *LOAD_MAP* or *LM*

Specifies a file to which the load map should be written. The default file name is $NULL.

STATUS

Returns the completion status of this command.

Remarks

- The name of the system task created by this command is BTF_SERVER.
- This command should be entered only once after deadstart. If it is necessary to enter a DEACTIVATE_BTFS command, then this command may be entered to reactivate the BTF server application.

ACTIVATE_INETD

Purpose Activates a system task for the Internet Daemon (INETD) server application.

Format

ACTIVATE_INETD
TRACE_MODE=boolean
STATUS=status variable

Parameters

TRACE_MODE or *TM*

Specifies whether to log tracing information.

TRUE

The INETD server application creates a file and logs tracing information to it. The file name is :$SYSTEM.$SYSTEM.TCP_IP.INETD_TRACE_FILE_OUTPUT.$NEXT.

FALSE

Tracing information is not logged.

The default value is FALSE.

STATUS

Returns the completion status of this command.

Remarks

- The name of the system task created by this command is INTERNET_DAEMON.
- This command should be entered only once after deadstart. If it is necessary to enter a DEACTIVATE_INETD command, then this command may be entered to reactivate the INETD server application.

ACTIVATE_NETWORK

Purpose Activates NAM/VE and executes the network activation prolog and epilog files ($SYSTEM.PROLOGS_AND_EPILOGS.NETWORK_ACTIVATION_PROLOG and $SYSTEM.PROLOGS_AND_EPILOGS.NETWORK_ACTIVATION_EPILOG) to activate the network tasks.

Format

ACTIVATE_NETWORK or
ACTN
STATUS=status variable

Parameters *STATUS*

Returns the completion status of this command:

Remarks

- If the NETWORK_ACTIVATION system attribute is set to 1 (the default value), the ACTIVATE_NETWORK command is executed automatically during deadstart. The NETWORK_ACTIVATION attribute is described in the NOS/VE System Performance and Maintenance manual, Volume 2.
- If the NETWORK_ACTIVATION parameter for the ACTIVATE_PRODUCTION_ENVIRONMENT command is TRUE (the default value), the ACTIVATE_NETWORK command is executed by the ACTIVATE_PRODUCTION_ENVIRONMENT command. The ACTIVATE_PRODUCTION_ENVIRONMENT command is described in the NOS/VE Operations manual.
- Once NAM/VE has been activated, the ACTIVATE_NETWORK command only executes the network activation epilog file.

ACTIVATE_NFS

Purpose Activates a system job for the Network File System on NOS/VE (NFS/VE) server application.

Format

ACTIVATE_NFS
DEBUG=boolean
STATUS=status variable

Parameters

DEBUG or *D*

Specifies whether to turn on debugging code.

TRUE

Debugging messages are written to the $SYSTEM.NFS.NFS_TASK_LOG and $SYSTEM.NFS.NFS_LOG files.

FALSE

Debugging code remains off.

The default value is FALSE.

STATUS

Returns the completion status of this command.

Remarks

- The name of the system job created by this command is NFS_DAEMON.
- This command should be entered only once after deadstart. If it is necessary to enter a DEACTIVATE_NFS command, then this command may be entered to reactivate the NFS/VE server application.
- Although this application operates as a server application, it is considered an IPC application which must be defined as a client application. See appendix C for more information about defining the IPC applications.
- The Portmapper server application must be activated with the ACTIVATE_PORTMAP command before the NFS/VE server application can be activated.

ACTIVATE_NTF

Purpose Defines and activates a system task for the Network Transfer Facility (NTF) client application. If the NTF system task definition already exists when this command is entered, that task definition is removed before a new task is defined and activated.

Format

ACTIVATE_NTF
AUTOMATIC_STATION_NAME=name
ACTIVATE_NTF_MAIL=boolean
LOAD_MAP=file
STATUS=status variable

Parameters

AUTOMATIC_STATION_NAME or **ASN**

Specifies the control facility name that should be used when you specify AUTOMATIC as the STATION parameter of the PRINT_FILE or SUBMIT_JOB command with a destination usage of NTF. This parameter is required.

ACTIVATE_NTF_MAIL or *ANM*

Specifies whether the NTF mail system task should be initiated at the same time as the NTF system task.

TRUE

If you specify TRUE and Mail/VE can process BITNET mail, the ACTIVATE_NTF command automatically issues the following command:

```
ACTIVATE_NTF_MAIL CONTROL_FACILITY_NAME=name
```

where name is the same control facility name that is specified for the AUTOMATIC_STATION_NAME parameter.

FALSE

If you specify FALSE or Mail/VE cannot process BITNET mail, the NTF mail system task is not initiated.

The default value is TRUE.

LOAD_MAP or *LM*

Specifies a file to receive load map information from the system task. If the LOAD_MAP parameter is omitted, no load map information is written.

STATUS

Returns the completion status of this command.

Remarks

- The name of the system task created by this command is NTF_CLIENT.
- This command should be entered only once after deadstart. If it is necessary to deactivate NTF using the DEACTIVATE_NTF command, then this command may be entered to reactivate the NTF client application.
- The ACTIVATE_NTF command defines and activates the system task for NTF. The NTF system task activates the BTF system task as an asynchronous task whenever a file is transferred to an active batch stream.
- The ACTIVATE_NTF command must be issued on each NOS/VE machine that is to send files to remote systems through NTF.
- The ACTIVATE_NTF command should be placed in the network activation epilog file ($SYSTEM.PROLOGS_AND_EPILOGS.NETWORK_ACTIVATION_EPILOG) for automatic activation across deadstarts. The network activation epilog file is described in the NOS/VE System Performance and Maintenance manual, Volume 2.

Examples

In this example, both the NTF system task and the NTF mail system task are initiated. NTF expects remote systems to be controlled by control facility NTF_CONTROL_FAC.

```
/activate_ntf asn=ntf_control_fac anm=true
```

ACTIVATE_NTF_MAIL

Purpose Defines and activates the Network Transfer Facility (NTF) mail system task that enables BITNET class M mail to be sent and received by Mail/VE Version 1.

Format

ACTIVATE_NTF_MAIL
CONTROL_FACILITY_NAME=name
LOAD_MAP=file
STATUS=status variable

Parameters

CONTROL_FACILITY_NAME or **CFN**

Specifies the name of the control facility that controls the NTF remote systems that can receive letters from Mail/VE Version 1. This paramter is required.

LOAD_MAP or *LM*

Specifies a file to receive load map information from the system task. If the LOAD_MAP parameter is omitted, no load map information is written.

STATUS

Returns the completion status of this command.

Remarks

- The name of the system task created by this command is NTF_MAIL_CLIENT.
- The ACTIVATE_NTF_MAIL command is issued automatically when the ACTIVATE_NTF command is specified with the ACTIVATE_NTF_MAIL parameter set to TRUE. For this special case, the control facility name for the ACTIVATE_NTF_MAIL command is the same as the control facility name specified in the AUTOMATIC_STATION_NAME parameter for the ACTIVATE_NTF command.

Examples In this example, the NTF mail system task is not activated by the ACTIVATE_NTF command. Also, the NTF mail system task is activated with a control facility name that is different than the automatic station name for NTF.

```
/activate_ntf asn=ntf_control_fac anm=false
/activate_ntf_mail cfn=ntf_bitnet
```

ACTIVATE_PCNFSD

Purpose Activates a system job for the PC Network File System (PCNFSD) server application.

Format

ACTIVATE_PCNFSD
FAMILY=name
DEBUG=boolean
STATUS=status variable

Parameters

FAMILY or **F**

Specifies the name of the family that PCs running PC-NFS can access. This parameter is required.

DEBUG or *D*

Specifies whether to turn on debugging code.

TRUE

Debugging messages are written to the $SYSTEM.NFS.PCNFSD_TASK_LOG and $SYSTEM.NFS.PCNFSD_LOG files.

FALSE

Debugging code remains off.

The default value is FALSE.

STATUS

Returns the completion status of this command.

Remarks

- The name of the system job created by this command is PCNFS_DAEMON.
- This command should be entered only once after deadstart. If it is necessary to enter a DEACTIVATE_PCNFSD command, then this command may be entered to reactivate the PCNFSD server application.
- Although this application operates as a server application, it is considered an IPC application which must be defined as a client application. See appendix C for more information about defining the IPC applications.
- The Portmapper server application must be activated with the ACTIVATE_PORTMAP command before the PCNFSD server application can be activated.
- The Portmapper, NFS/VE, and PCNFSD server applications must be activated before a PC can use PC-NFS to access the CYBER file system.
- PCNFSD is limited to one family per CYBER mainframe.

ACTIVATE_PORTMAP

Purpose Activates a system job for the Portmapper server application.

Format

ACTIVATE_PORTMAP
DEBUG=boolean
STATUS=status variable

Parameters

DEBUG or *D*

Specifies whether to turn on debugging code.

TRUE

Debugging messages are written to the $SYSTEM.NFS.PORTMAPPER_TASK_LOG and $SYSTEM.NFS.PORTMAPPER_LOG files.

FALSE

Debugging code remains off.

The default value is FALSE.

STATUS

Returns the completion status of this command.

Remarks

- The name of the system job created by this command is PORTMAP_DAEMON.
- This command should be entered only once after deadstart. If it is necessary to enter a DEACTIVATE_PORTMAP command, then this command may be entered to reactivate the PORTMAPPER server application.
- Although this application operates as a server application, it is considered an IPC application which must be defined as a client application. See appendix C for more information about defining the IPC applications.
- The Portmapper server application must be activated with the ACTIVATE_PORTMAP command before the NFS/VE or PCNFSD server applications can be activated.

ACTIVATE_PTF

Purpose Defines and activates a system task for the Permanent File Transfer Facility (PTF) server application. If the PTF system task definition already exists when this command is entered, that task definition is removed before a new task is defined and activated.

Format

ACTIVATE_PTF
LOAD_MAP=file
STATUS=status variable

Parameters *LOAD_MAP* or *LM*

Specifies a file to which the load map should be written. The default file name is $NULL.

STATUS

Returns the completion status of this command.

Remarks

- The name of the system task created by this command is OSA$FILE_TRANSFER_SERVER.
- This command should be entered only once after deadstart. If it is necessary to deactivate PTF using the DEACTIVATE_PTF command, then this command may be entered to reactivate the PTF server application.

ACTIVATE_QTF

Purpose Defines and activates a system task for the Queue File Transfer Facility (QTF) client application. If the QTF system task definition already exists when this command is entered, that task definition is removed before a new task is defined and activated.

Format

ACTIVATE_QTF
HOST_PHYSICAL_IDENTIFIER=string
MAXIMUM_SUBTASKS=keyword or integer
LOAD_MAP=file
STATUS=status variable

Parameters

HOST_PHYSICAL_IDENTIFIER or **HPI**

Specifies the physical identifier of the local host mainframe. The value specified can be a string of up to 31 characters. This parameter is required.

MAXIMUM_SUBTASKS or *MS*

Specifies the maxium number of QTF subtasks that may be active at one time. The value specified can be an integer from 1 to 20 or the keyword UNLIMITED. The default value is UNLIMITED.

LOAD_MAP or *LM*

Specifies a file to which the load map should be written. The default file name is $NULL.

STATUS

Returns the completion status of this command.

Remarks

- The name of the system task created by this command is QUEUE_TRANSFER_CLIENT.
- This command should be entered only once after deadstart. If it is necessary to enter a DEACTIVATE_QTF command, then this command may be entered to reactivate the QTF client application.
- The HOST_PHYSICAL_IDENTIFIER parameter is used when QTF uses NAM/VE to transfer a file. If LCN is used, the physical identifier of the host is obtained by the application. The host physical identifier specified when activating QTF should be the same as the one specified when activating QTFS.

ACTIVATE_QTFS

Purpose Defines and activates a system task for the Queue File Transfer Facility Server (QTFS) application. If the QTFS system task definition already exists when this command is entered, that task definition is removed before a new task is defined and activated.

Format

ACTIVATE_QTFS
HOST_PHYSICAL_IDENTIFIER=string
LOAD MAP=file
STATUS=status variable

Parameters

HOST_PHYSICAL_IDENTIFIER or **HPI**

Specifies the physical identifier of the local host mainframe. The value specified can be a string of up to 31 characters. This parameter is required.

LOAD_MAP or *LM*

Specifies a file to which the load map should be written. The default file name is $NULL.

STATUS

Returns the completion status of this command.

Remarks

- The name of the system task created by this command is QUEUE_TRANSFER_SERVER.
- This command should be entered only once after deadstart. If it is necessary to enter a DEACTIVATE_QTFS command, then this command may be entered to reactivate the QTF server application.
- The HOST_PHYSICAL_IDENTIFIER parameter is used when QTFS uses NAM/VE to transfer a file. If LCN is used, the physical identifier of the host is obtained by the application. The host physical identifier specified when activating QTFS should be the same as the one specified when activating QTF.

ACTIVATE_SCF

Purpose Defines and activates a system task for the Status and Control Facility/VE (SCF/VE) client application. If the SCF system task definition already exists when this command is entered, that task definition is removed before a new task is defined and activated.

Format

ACTIVATE_SCF
LOAD_MAP=file
STATUS=status variable

Parameters *LOAD_MAP* or *LM*

Specifies a file to which the load map should be written. The default file name is $NULL.

STATUS

Returns the completion status of this command.

Remarks

- The name of the system task created by this command is SCF_CLIENT.
- This command should be entered only once after deadstart. If it is necessary to enter a DEACTIVATE_SCF command, then this command may be entered to reactivate the SCF client application.
- The ACTIVATE_SCF command defines and activates the system task for SCF. The SCF system task activates the BTF system task as an asynchronous task whenever output is transferred to an active batch device.

ACTIVATE_SCFS

Purpose Defines and activates a system task for the Status and Control Facility Server/VE (SCFS/VE) application. If the SCFS system task definition already exists when this command is entered, that task definition is removed before a new task is defined and activated.

Format

ACTIVATE_SCFS
APPLICATION_NAME=name
CONTROL_FACILITY_NAME=name
SYSTEM_TASK_NAME=name
LOAD_MAP=file
LOGGING=boolean
NTF_SYSTEM_LIST=file
STATUS=status variable

Parameters

APPLICATION_NAME or *AN*

Specifies the name of the SCFS application (as defined previously on the DEFINE_SCFS command). The default name is OSA$STATUS_CONTROL_FAC_SERVER. See Remarks for further restrictions on the use of this parameter.

CONTROL_FACILITY_NAME or *CFN*

Specifies a name for the control facility. The default name is STATION_CONTROLLER_1. See Remarks for further restrictions on the use of this parameter.

SYSTEM_TASK_NAME or *STN*

Specifies the name of the system task. The default name is SCF_SERVER. See Remarks for further restrictions on the use of this parameter.

LOAD_MAP or *LM*

Specifies a file to which the load map should be written. The default file name is $NULL.

LOGGING or *L*

Specifies whether or not to log SCF server events.

TRUE

SCF server events are logged.

FALSE

SCF server events are not logged.

The default value is FALSE.

NTF_SYSTEM_LIST or *NSL*

Specifies the file in which the remote system names used by NTF are stored. This is an optional parameter that must be specified if NTF is to be run in conjunction with this SCFS application. If it is not specified, this SCFS application will not control NTF remote systems.

STATUS

Returns the completion status of this command.

Remarks

- The default name of the system task is SCF_SERVER.
- This command should be entered only once after deadstart. If it is necessary to enter a DEACTIVATE_SCFS command, then this command may be entered to reactivate the SCF server application.
- The parameters APPLICATION_NAME, CONTROL_FACILITY_NAME, and SYSTEM_TASK_NAME can be used only in one of the following combinations:
  - Specify none of the parameters and allow the defaults to be used.
  - Specify all three parameters by entering values other than the default values.
  - Specify only the parameter APPLICATION_NAME by entering a value other than the default value. In this case, the name specified for APPLICATION_NAME will also be used for the CONTROL_FACILITY_NAME and SYSTEM_TASK_NAME parameters.
- The file $SYSTEM.CDCNET.SITE_CONTROLLED.PROCEDURES.TERMINAL is a file of command procedures. Procedures on this file may contain commands for DEFINE_I_O_STATION and DEFINE_USER_I_O_STATION. Each of these commands has a CONTROL_FACILITY parameter. In order for the I/O stations to communicate with the NOS/VE SCF server, the value specified for this parameter on both commands must match the value of the CONTROL_FACILITY_NAME parameter of the ACTIVATE_SCFS command.
- The SCFS logging capability (controlled by the logging parameter) is normally used only for debugging purposes. It is recommended that the SCFS log be enabled only when advised to do so by a Control Data customer engineer or analyst. If the SCF logging capability is enabled, a binary log file is created in catalog $SYSTEM.BATCH_DEVICE_SUPPORT.SCFS_LOG. The name of the binary log is the same as the control facility name.
- Multiple copies of the SCFS system task can be resident in a NOS/VE system or in the catenet. If additional copies are needed, they can be defined with unique system task names (if in the same NOS/VE system) at any time.
- If SCFS is defined more than once in a catenet, unique names must be used for the control facility names. Unique names must also be used for application names and system task names defined within the same NOS/VE system. This means that the default name for the SCFS server application can be used only once in a given network.
- Be sure that all I_O_STATION_NAMES (IOSN parameter on the DEFIOS TDP commands) in the catenet differ from the control facility names. Like the name for the control facility (or the Timesharing application title), each IOSN is a title to be registered in the catenet.
- If SCFS is to be run in association with the Network Transfer Facility (NTF), the NTF_SYSTEM_LIST parameter must specify the file in which the remote system names used by NTF are stored.

ACTIVATE_SMTP

Purpose Defines and activates a system task for the Simple Mail Transfer Protocol (SMTP) client application. If the SMTP system task already exists when this command is entered, that task definition is removed before a new task is defined and activated.

Format

ACTIVATE_SMTP
DEBUG_MODE=boolean
LOAD_MAP=file
MAXIMUM_QUEUED_TIME=keyword or *integer*
MAXIMUM_SUBTASKS=integer
PROCESS_INTERVAL=integer
RETRANSMISSION_INTERVAL=integer
STATUS=status variable

Parameters

DEGUG_MODE or *DM*

Specifies whether to turn on debugging code.

TRUE

Debugging messages are written to the system job log.

FALSE

Debugging code remains off.

The default value is FALSE.

LOAD_MAP or *LM*

Specifies a file to which the load map should be written. The default file is $NULL.

MAXIMUM_QUEUED_TIME or *MQT*

Specifies the maximum time interval in minutes that a message can stay in the SMTP mail queue to wait for retransmission. The message is deleted when the specified time has elapsed. Specify an integer in the range from 0 to 65,535 or the keyword INFINITE. If 0 is specified, a message is not queued for a later retry; it is deleted and the system operator is notified. If INFINITE is specified, the message stays in the queue until it is either delivered or deleted from the queue by the system operator by way of a queue file management command. The default value is 60.

MAXIMUM_SUBTASKS or *MS*

Specifies the maximum number of concurrent mail delivery tasks that can be active at one time. Specify an integer in the range from 1 to 20. This parameter allows a site to keep the SMTP client from using excessive system resources for mail deliveries. The default value is 10.

PROCESS_INTERVAL or *PI*

Specifies the time interval in minutes to poll the SMTP mail queue for newly arrived messages to process. Specify an integer in the range from 0 to 60. If 0 is specified, a message is processed as soon as it is placed in the SMTP mail queue. A maximum value is imposed in order to prevent excessive build up of SMTP mail messages in the mail queue. The default value is 5.

RETRANSMISSION_INTERVAL or *RI*

Specifies the time interval in minutes to initiate retransmission of previously undeliverable mail messages in the SMTP mail queue. Specify an integer in the range from 5 to 60. The default value is 15.

STATUS

Returns the completion status of the command.

Remarks

- The name of the system task created by this command is SMTP_CLIENT.
- This command should be entered only once after deadstart. If it is necessary to enter a DEACTIVATE_SMTP command, then this command may be entered to reactivate the SMTP client application.

DEACTIVATE_BTFS

Purpose Deactivates and removes the system task for the Batch Transfer Facility Server/VE (BTFS/VE) application.

Format

DEACTIVATE_BTFS
STATUS=status variable

Parameters

STATUS

Returns the completion status of this command.

Remarks

- The name of the system task deactivated and removed by this command is BTF_SERVER.
- After the BTF server is deactivated by this command, the ACTIVATE_BTFS command can be used to reactivate the BTF server application.

DEACTIVATE_INETD

Purpose Deactivates the system task for the Internet Daemon (INETD) server application.

Format

DEACTIVATE_INETD
STATUS=status variable

Parameters

STATUS

Returns the completion status of this command.

Remarks

- The name of the system task deactivated by this command is INTERNET_DAEMON.
- After the INETD server is deactivated by this command, the ACTIVATE_INETD command can be used to reactivate the INETD server application.

DEACTIVATE_NFS

Purpose Deactivates and removes the system job for the Network File System on NOS/VE (NFS/VE) server application.

Format

DEACTIVATE_NFS
STATUS=status variable

Parameters

STATUS

Returns the completion status of this command.

Remarks

- The name of the system job deactivated and removed by this command is NFS_DAEMON.
- After the NFS/VE server is deactivated by this command, the ACTIVATE_NFS command can be used to reactivate the NFS/VE server application.

DEACTIVATE_NTF

Purpose Deactivates and removes the system task for the Network Transfer Facility (NTF) client application.

Format

DEACTIVATE_NTF
STATUS=status variable

Parameters *STATUS*

Returns the completion status of this command.

Remarks

- The name of the system task deactivated and removed by this command is NTF_CLIENT.
- After the NTF client is deactivated by this command, the ACTIVATE_NTF command can be used to reactivate the NTF client application.

DEACTIVATE_NTF_MAIL

Purpose Deactivates and removes the Network Transfer Facility (NTF) mail system task that enables BITNET class M mail to be sent and received by Mail/VE Version 1.

Format

DEACTIVATE_NTF_MAIL
STATUS=status variable

Parameters

STATUS

Returns the completion status of this command.

Remarks

- The name of the system task deactivated and removed by this command is NTF_MAIL_CLIENT.
- After the NTF_MAIL_CLIENT system task is deactivated by this command, the ACTIVATE_NTF_MAIL command can be used to reactivate the NTF_MAIL_CLIENT system task.

DEACTIVATE_PCNFSD

Purpose Deactivates and removes the system job for the PC Network File System (PCNFSD) server application.

Format

DEACTIVATE_NFS
STATUS=status variable

Parameters *STATUS*

Returns the completion status of this command.

Remarks

- The name of the system job deactivated and removed by this command is PCNFS_DAEMON.
- After the PCNFSD server is deactivated by this command, the ACTIVATE_PCNFSD command can be used to reactivate the PCNFSD server application.

DEACTIVATE_PORTMAP

Purpose Deactivates and removes the system job for the Portmapper server application.

Format

DEACTIVATE_PORTMAP
STATUS=status variable

Parameters

STATUS

Returns the completion status of this command.

Remarks

- The name of the system job deactivated and removed by this command is PORTMAP_DAEMON.
- After the Portmapper server is deactivated by this command, the ACTIVATE_PORTMAP command can be used to reactivate the Portmapper server application.

DEACTIVATE_PTF

Purpose Deactivates and removes the system task for the Permanent File Transfer Facility (PTF) server application.

Format

DEACTIVATE_PTF
STATUS=status variable

Parameters

STATUS

Returns the completion status of this command.

Remarks

- The name of the system task deactivated and removed by this command is OSA$FILE_TRANSFER_SERVER.
- After the PTF server is deactivated by this command, the ACTIVATE_PTF command can be used to reactivate the PTF server application.

DEACTIVATE_QTF

Purpose Deactivates and removes the system task for the Queue File Transfer Facility (QTF) client application.

Format

DEACTIVATE_QTF
STATUS=status variable

Parameters

STATUS

Returns the completion status of this command.

Remarks

- The name of the system task deactivated and removed by this command is QUEUE_TRANSFER_CLIENT.
- After the QTF client is deactivated by this command, the ACTIVATE_QTF command can be used to reactivate the QTF client application.

DEACTIVATE_QTFS

Purpose Deactivates and removes the system task for the Queue File Transfer Facility Server (QTFS) application.

Format

DEACTIVATE_QTFS
STATUS=status variable

Parameters

STATUS

Returns the completion status of this command.

Remarks

- The name of the system task deactivated and removed by this command is QUEUE_TRANSFER_SERVER.
- After the QTF server is deactivated by this command, the ACTIVATE_QTFS command can be used to reactivate the QTF server application.

DEACTIVATE_SCF

Purpose Deactivates and removes the system task for the Status and Control Facility/VE (SCF/VE) client application.

Format

DEACTIVATE_SCF
STATUS=status variable

Parameters *STATUS*

Returns the completion status of this command.

Remarks

- The name of the system task deactivated and removed by this command is SCF_CLIENT.
- After the SCF client is deactivated by this command, the ACTIVATE_SCF command can be used to reactivate the SCF client application.

DEACTIVATE_SCFS

Purpose Deactivates and removes the system task for the Status and Control Facility Server/VE (SCFS/VE) application.

Format

DEACTIVATE_SCFS
SYSTEM_TASK_NAME=name
STATUS=status variable

Parameters

SYSTEM_TASK_NAME or *STN*

Specifies the name of the system task. The name specified for this parameter must match the name specified for the SYSTEM_TASK_NAME parameter of the ACTIVATE_SCFS command. The default name is SCF_SERVER.

STATUS

Returns the completion status of this command.

Remarks After the SCF server is deactivated by this command, the ACTIVATE_SCFS command can be used to reactivate the SCF server application.

DEACTIVATE_SMTP

Purpose Deactivates and removes the system task for the Simple Mail Transfer Protocol (SMTP) client application.

Format

DEACTIVE_SMTP
STATUS=status variable

Parameters

STATUS

Returns the completion status of this command.

Remarks

- The name of the system task deactivated and removed by this command is SMTP_CLIENT.
- After the SMTP client is deactivated by this command, the ACTIVATE_SMTP command can be used to reactivate the SMTP client application.

CDCNET Management Entities 6

CDCNET Management Entities 6

This chapter defines and describes CDCNET Management Entities and how to activate and deactivate them. Descriptions of the commands for activating and deactivating Management Entities are included. This chapter also includes brief descriptions of those CDCNET and NOS/VE utilities you can use to manage a network.

Management Entities (MEs)

CDCNET provides software applications called Management Entities (MEs). MEs provide support for network administration and allow network resources to be shared between a NOS/VE host and remote CDCNET systems.

The MEs supported by NOS/VE are Initialization, File Access, Log, Clock, Command, and Alarm. The Initialization, File Access, Log, and Clock MEs provide services to the other systems in the network. Each of these entities executes as a system task in the system job and is activated and deactivated by NOS/VE operator commands. (These commands are described later in this chapter under Management Entity Commands.) Only one instance of each ME may be active at a time.

Initialization ME

The Initialization ME provides a method for initializing a device interface (DI) on a directly connected network. The Initialization ME initializes mainframe DIs that are directly connected to a channel on a NOS/VE host computer. A CYBER 930 computer can initialize terminal device interfaces (TDIs) and network device interfaces (NDIs) on the directly connected ethernet.

File Access ME

The File Access ME provides access to network management permanent files for those systems in the network that do not support a file storage device. The network management files reside in the $SYSTEM.CDCNET catalog on NOS/VE. These files provide capabilities to dump, load, and configure DIs. They provide a DI with access to terminal, user, operator, and device-load procedures.

Log ME

The Log ME records all log messages received from DIs in the network on the permanent file $SYSTEM.CDCNET.LOG. Log messages can be generated, for example, to note significant events, record statistics, and report status.

When a log file is terminated for any reason, the LOG_ME task submits a job to perform the desired processing on the log file. You can obtain the job file from the file $SYSTEM.CDCNET.VERSION_INDEPENDENT.PROCESS_LOG_JOB. Typical functions performed by this job are: executing the log file compression routine for all unprocessed log files, archiving or purging the log files, and executing the NOS/VE Network Performance Analyzer utility.

Clock ME

The Clock ME synchronizes the clocks in DIs in the network. Clock synchronization is initiated by the DI being synchronized. The Clock ME provides the current clock value to any system requesting it and verifies that the clock is set within acceptable tolerances.

Command and Alarm MEs

The Command and Alarm MEs are executed in a NOS/VE user job as a command utility called Network Operator Utility. For more information about the Network Operator Utility, see the CDCNET Network Operations manual.

The Command ME allows a network operator to send commands to remote DIs for execution and to receive the responses for the commands executed. The operator can send commands, for example, to inquire about a DI's status, request statistics, activate or deactivate logging, and execute diagnostics.

The Alarm ME allows a network operator to receive alarm messages from remote DIs. An alarm is a message generated by a DI when an abnormal event occurs.

Management Entity Activation and Deactivation

There are two ways to activate an ME:

- Enter the ME activation commands at the system console. ME commands cannot be executed from an interactive terminal.
- Include the ME activation commands in the network activation epilog file ($SYSTEM.PROLOGS_AND_EPILOGS.NETWORK_ACTIVATION_EPILOG) so the services are activated automatically during NOS/VE deadstart. Refer to the NOS/VE System Performance and Maintenance manual, Volume 2, for more information about the network activation epilog file.

The only way to deactivate an ME is to enter an ME deactivation command from the system console.

Once an ME is activated, you can change parameter values for that ME only by deactivating and then reactivating the ME with the new parameter values.

Management Entity Commands

This section describes the ME commands you use to activate and deactivate the network management services.

All ME commands have an optional STATUS parameter. Information on using the STATUS parameter is in the NOS/VE System Usage manual.

ACTIVATE_NETWORK_CLOCK

Purpose Activates the Clock ME for the network. The network Clock ME allows DI clocks throughout the network to be synchronized to a common time. The time used for synchronization is obtained from the clock in the NOS/VE host that activates the ME.

NOTE

The network requires only one network Clock ME. If network Clock MEs are activated on more than one DI or NOS/VE host in the network, DI clocks may be synchronized only to the extent that the local clocks of the various network Clock MEs are synchronized.

Format

ACTIVATE_NETWORK_CLOCK or
ACTNC
MAXIMUM_CONNECTIONS=integer
STATUS=status variable

Parameters *MAXIMUM_CONNECTIONS* or *MC*

Specifies the maximum number of concurrent connections to the network Clock ME. You can specify an integer from 1 to 1000. The default is 1000.

STATUS

Returns the completion status of this command.

Remarks You can change network Clock ME parameters only by deactivating the ME, then reactivating it with the new parameter values. This action, however, interrupts active connections to the network Clock ME.

ACTIVATE_NETWORK_FILE_ACCESS

Purpose Activates the File Access ME for DIs that do not support a file storage device.

Format

ACTIVATE_NETWORK_FILE_ACCESS or
ACTNFA
FILE_TYPE=list of keyword
MAXIMUM_CONNECTIONS=integer
MAXIMUM_DUMPS=integer
MAXIMUM_DUMP_SIZE=integer
STATUS=status variable

Parameters *FILE_TYPE* or *FT*

Specifies the list of file types supported by the network File Access ME. The network File Access ME registers a title in the network directory for each supported file type. See the CDCNET System Programmer's Reference Set, Volume 2, for a detailed description of these file types.

You can specify one or more of the following keywords:

ALL

Supports all file types.

BOOT

Supports network boot files.

CONFIGURATION

Supports network DI configuration files.

DUMP

Supports network DI dump files.

EXCEPTION

Supports network exception files.

LIBRARY

Supports network library files.

LOAD_PROCEDURE

Supports network device-load procedure files.

OPERATOR_PROCEDURE

Supports network operator procedure files.

TERMINAL_PROCEDURE

Supports network terminal procedure files.

USER_PROCEDURE

Supports network user procedure files.

The default is ALL.

MAXIMUM_CONNECTIONS or *MC*

Specifies the maximum number of concurrent connections to the network File Access ME. You can specify an integer from 1 to 1000. The default is 1000.

MAXIMUM_DUMPS or *MD*

Specifies the maximum number of dump files allowed for a given DI. When the dump file limit for a DI is reached, the network File Access ME rejects attempts to create additional dump files for that DI. You can specify an integer from 0 to 1000. The default is 10.

MAXIMUM_DUMP_SIZE or *MDS*

Specifies the maximum size of a dump file in bytes. When a dump file reaches this limit, the network File Access ME closes the file and terminates the dump. You can specify an integer from 0 to 4,398,046,511,103 ($2^{42} - 1$). The default is 4.5 million bytes.

STATUS

Returns the completion status of this command.

Remarks

- If a connection is established to access an unsupported file type, the ME issues a log message to the system job log.
- If the ME detects that a supported file type is unavailable (either at activation or as a result of an attempt to reference the file type), the ME deletes the file type title from the network directory and issues a log message to the system job log.
- If the ME detects that none of the supported file types are available (either at activation or as a result of an attempt to reference the file types), it issues a warning message and deactivates itself. When the file types are available, the ME must be explicitly reactivated with an ACTIVATE_NETWORK_FILE_ACCESS command.
- You can change the network File Access ME parameters only by deactivating the ME, then reactivating it with the new parameter values. This action, however, interrupts active connections to the network File Access ME.

ACTIVATE_NETWORK_INITIALIZER

Purpose Activates the Initialization ME for directly connected DIs that cannot self-initialize.

Format

ACTIVATE_NETWORK_INITIALIZER or
ACTNI
PRIORITY=integer
MAXIMUM_CONNECTIONS=integer
MAXIMUM_DUMPS=integer
MAXIMUM_DUMP_SIZE=integer
STATUS=status variable

Parameters

PRIORITY or *P*

Specifies the priority of the activated network Initialization ME relative to other network Initialization MEs in the network. You can specify an integer from 0 to 3. The default is 3, which is the highest priority.

MAXIMUM_CONNECTIONS or *MC*

Specifies the maximum number of concurrent connections to the network Initialization ME. You can specify an integer from 1 to 1000. The default is 1000.

MAXIMUM_DUMPS or *MD*

Specifies the maximum number of dump files allowed for a given DI. When the dump file limit for a DI is reached, the ME does not create additional dump files when initializing that DI. You can specify an integer from 0 to 1000. The default is 10.

MAXIMUM_DUMP_SIZE or *MDS*

Specifies the maximum size of a dump file in bytes. When a dump file reaches this limit, the ME closes the file and terminates the dump. You can specify an integer from 0 to 4,398,046,511,103 ($2^{42} - 1$). The default is 4.5 million bytes.

STATUS

Returns the completion status of this command.

Remarks

- Once a DI is initialized, it attempts to access its configuration file. Access to this file must be provided by a network File Access ME that is located on the same network that initialized the DI.
- You can change the network Initialization ME parameters only by deactivating the ME, then reactivating it with the new parameter values. This action, however, interrupts active connections to the ME.
- Each NOS/VE system that is configured to support a Mainframe Device Interface (MDI) or Mainframe Terminal Interface (MTI) must have an active network Initialization ME to reload the MDI or MTI in the event of a failure. The exception is an MDI that is configured to load across another network. An MTI can be loaded only by its host system.

ACTIVATE_NETWORK_LOG

Purpose Activates the Log ME for DIs that cannot store and analyze log information.

Format

ACTIVATE_NETWORK_LOG or
ACTNL
GROUPS=list of (name,integer)
MAXIMUM_CONNECTIONS=integer
MAXIMUM_LOG_CYCLES=integer
MAXIMUM_LOG_SIZE=keyword or *integer*
INTERVAL=keyword or *integer*
STATUS=status variable

Parameters

GROUPS or *GROUP* or *G*

Specifies a list of pairs of values that indicates the name and priority of each log group for which log entries are collected. The value specified for priority can be an integer in the range from 1 to 255, with 1 being the highest priority. The priority for the specified log group is relative to other network Log MEs in the network. See the CDCNET Configuration and Site Administration Guide for valid names of log groups. The default is (CATENET,1).

MAXIMUM_CONNECTIONS or *MC*

Specifies the maximum number of concurrent connections to the network Log ME. You can specify an integer from 1 to 1000. The default is 1000.

MAXIMUM_LOG_CYCLES or *MLC*

Specifies the maximum number of log file cycles allowed. You can specify an integer from 2 to 999. The default is 999.

NOTE

When log file cycle 999 is reached, logging is disabled in the host until all old log file cycles are purged (up to and including cycle 999). Usually, this is done by the log file processing job. Once the old cycles are purged, logging resumes and writing to log file cycle 1 begins.

MAXIMUM_LOG_SIZE or *MLS*

Specifies the maximum size of a log file cycle in bytes. You can specify an integer in the range from 0 to 4,398,046,511,103 ($2^{42} - 1$). When a log file cycle reaches this limit, the log file is closed and processed, and a new log file cycle is created. If the keyword NONE is specified, the system-specified maximum file size is used (the maximum file size is specified by a system configuration parameter). The default is NONE.

INTERVAL or *I*

Specifies the time interval in minutes between log file cycles. You can specifiy an integer in the range from 1 to 1440. When the specified time has elapsed, the active log file cycle is closed and processed, and a new log file cycle is started. If the keyword NONE is specified, the ME will not process log files.

STATUS

Returns the completion status of this command.

Remarks

- You can change the network Log ME parameters only by deactivating the ME, then reactivating it with new parameter values. This action, however, interrupts active connections to the ME.
- When the limit on the number of log file cycles is reached, the network log service issues a log message and retries at one-minute intervals. You must process and delete at least one of the existing log file cycles either manually or with the automated log processing job.
- The network Log ME places log data on a new cycle of the $SYSTEM.CDCNET.LOG file. If the log catalog does not exist, it is created. When either the log file reaches its maximum size or the log file processing interval has elapsed, the current file is closed and a new log file cycle is created.
- When a log file is closed because the file has reached its maximum size or the log file processing interval has elapsed, the network Log ME submits a job to process the log file. The job file is obtained from file $SYSTEM.CDCNET.VERSION_INDEPENDENT.PROCESS_LOG_JOB. This file is included with the CDCNET system release materials and contains a sample log file processing job.

  Typically, the job includes these functions (among others): execute the log file compression routine for all unprocessed log files, archive or purge the log files, and execute the network performance analyzer.

DEACTIVATE_NETWORK_CLOCK

Purpose Deactivates the Clock ME for the network.

Format **DEACTIVATE_NETWORK_CLOCK** or
DEANC
STATUS=status variable

Parameters *STATUS*

Returns the completion status of this command.

Remarks When the network Clock ME is deactivated, all active connections are terminated.

DEACTIVATE_NETWORK_FILE_ACCESS

Purpose Deactivates the File Access ME for the network.

Format **DEACTIVATE_NETWORK_FILE_ACCESS** or
DEANFA
STATUS=status variable

Parameters *STATUS*

Returns the completion status of this command.

Remarks When the network File Access ME is deactivated, all active connections are terminated.

DEACTIVATE_NETWORK_INITIALIZER

Purpose Deactivates the Initialization ME for directly connected DIs.

Format **DEACTIVATE_NETWORK_INITIALIZER** or
DEANI
STATUS=status variable

Parameters *STATUS*

Returns the completion status of this command.

Remarks

- Each NOS/VE system that is configured to support a Mainframe Device Interface (MDI) or Mainframe Terminal Interface (MTI) must have an active network Initialization ME to reload the MDI or MTI in the event of a failure. The exception is an MDI that is configured to load across another network. An MTI can be loaded only by its host system. Therefore, under normal operating conditions, do not deactivate the network Initialization ME.
- When the network Initialization ME is deactivated, all active connections are terminated.

DEACTIVATE_NETWORK_LOG

Purpose Deactivates the Log ME for the network.

Format **DEACTIVATE_NETWORK_LOG** or
DEANL
STATUS=status variable

Parameters *STATUS*

Returns the completion status of this command.

Remarks When the network Log ME is deactivated, the current cycle of the $SYSTEM.CDCNET.LOG file is closed and returned, and all active connections are terminated.

Utilities for Network Management

The following CDCNET and NOS/VE utilities can be used to manage a network that includes NOS/VE hosts. They give you convenient access to CDCNET management activities involving configuration, report generation, and operator support.

Manage CDCNET Configuration Utility

The Manage CDCNET Configurations (MANCC) utility allows you to create and update the CDCNET configuration files. The MANCC utility operates as a full-screen editor containing function key definitions and help screens. For more information about MANCC, see the CDCNET Configuration and Site Administration Guide listed in appendix B.

CDCNET Network Configuration Utility

The CDCNET Network Configuration Utility (NETCU) is a NOS/VE based utility that enables you to enter configuration information for your CDCNET network. Based on the information you enter, NETCU creates a configuration database for your network and generates CDCNET configuration procedures. For more information about NETCU, refer to the NETCU online manual, the NETCU Summary Card, and the CDCNET Configuration and Site Administration Guide listed in appendix B.

NOTE

In a future release of NOS/VE, this utility may replace the Manage CDCNET Configuration Utility (MANCC).

Network Performance Analyzer Utility

The Network Performance Analyzer (NPA) NOS/VE utility produces a wide variety of statistical reports that you can use for network planning and for communication-link performance evaluating. NPA generates its reports from host-maintained CDCNET log files. For more information about NPA, see the CDCNET Network Performance Analyzer manual listed in appendix B.

Network Operator Utility

The Network Operator Utility (NETOU) provides operator support for CDCNET systems. It also implements the Alarm and Command MEs. For more information about NETOU, see the CDCNET Network Operations manual listed in appendix B.

Tailoring NAM/VE Attributes 7

Tailoring NAM/VE Attributes 7

The following commands, CHANGE_NAM_ATTRIBUTES and DISPLAY_NAM_ATTRIBUTES, change and display various NAM/VE attributes. These capabilities allow you to tune the operation of NAM/VE to your particular environment and, as a result, improve performance and use of resources. It is recommended that the CHANGE_NAM_ATTRIBUTES command be placed in the network activation prolog file ($SYSTEM.PROLOGS_AND_EPILOGS.NETWORK_ACTIVATION_PROLOG). The network activation prolog file is described in the NOS/VE System Performance and Maintenance manual, Volume 2.

CHANGE_NAM_ATTRIBUTES

Purpose Modifies specified NAM attributes to improve NAM/VE performance and use of network resources.

Format

CHANGE_NAM_ATTRIBUTES or
CHANA

ENABLE_CHECKSUMMING=boolean
ENABLE_STATISTICS=boolean
MAXIMUM_CONNECTIONS=integer
MAXIMUM_ICA_DUMPS=integer
MAXIMUM_LOGIN_ATTEMPS=integer
MCI_RESET_DOWN_THRESHOLD=keyword or integer
ICA_RESET_DOWN_THRESHOLD=keyword or integer
PREFERRED_PROTOCOL_STACK=keyword
ADDITIONAL_LOGIN_PROMPTS=list of keyword
STATUS=status variable

Parameters

ENABLE_CHECKSUMMING or *EC*

Specifies whether NAM/VE enables checksumming for all locally generated messages.

TRUE

Checksumming is enabled for locally generated messages.

FALSE

Checksumming is not enabled for locally generated messages.

The default is TRUE.

ENABLE_STATISTICS or ES

Specifies whether NAM/VE statistics are collected.

TRUE

NAM/VE statistics are collected.

FALSE

NAM/VE statistics are not collected.

The default is FALSE.

MAXIMUM_CONNECTIONS or *MC*

Specifies the maximum number of concurrent connections that NAM/VE will support. This maximum is a global maximum that includes all connections for all applications. In addition, it includes connections required for network management and administrative functions. When the total number of connections reaches this limit, no further connections are accepted. This attribute should be changed prior to activating NAM/VE.

The value specified can be from 0 to 4096. The default is 256.

NOTE

When you change the MAXIMUM_CONNECTIONS attribute, consideration should be given to changing related system and job class attributes, since each NAM/VE connection requires a certain set of system resources (for example, maximum jobs, maximum tasks).

MAXIMUM_ICA_DUMPS or *MAXID*

Specifies the maximum number of ICA dump files that may be created for each ICA. When the number of dump files for a given ICA reaches this limit, no further dumps are taken.

The value specified can be from 0 to 1000. The default is 3.

MAXIMUM_LOGIN_ATTEMPTS or *MLA*

Specifies the maximum number of login attempts that NAM/VE will allow. When the number of unsuccessful login attempts reaches this limit, NAM/VE terminates the connection between the host and the network, issues an error message to both the user and the system operator, and emits a statistic.

The value specified can be from 1 to 10. The default is 3.

MCI_RESET_DOWN_THRESHOLD or *MRDT*

Specifies the threshold value for MCI reset frequency. When this threshold value is reached, NAM/VE changes the element state of an MCI to DOWN. The threshold is expressed as the number of resets that occur in a 10-minute interval.

The value specified can be from 2 to 100. The keyword NONE indicates that no threshold is to be applied. The default is 10.

ICA_RESET_DOWN_THRESHOLD or *IRDT*

Specifies the threshold value for ICA reset frequency. When this threshold value is reached, NAM/VE changes the element state of an ICA to DOWN. The threshold is expressed as the number of resets that occur in a 10-minute interval.

The value specified can be from 2 to 100. The keyword NONE indicates that no threshold is to be applied. The default is 10.

PREFERRED_PROTOCOL_STACK or *PPS*

Specifies which protocol stack to use for communications with a CDNA bridge system. In a bridge system, the OSI and XNS protocol stacks coexist and users of CDNA session and generic transport can communicate over either stack. A change in the value of this attribute only affects subsequent requests for communication. All communications initiated before this change proceed to completion using the protocol stack specified at the time.

You can specify one of the following keywords:

OSI

Selects the Open System Interconnection (OSI) protocol stack.

XNS

Selects the Xerox Network Systems (XNS) protocol stack.

The default is XNS.

ADDITIONAL_LOGIN_PROMPTS or *ALP*

Specifies which prompts are issued during the login dialog in addition to the user and password prompts. Changing these attributes may affect login dialogs that are already in progress when the change is made.

You can specify one or more of the following keywords:

ALL

Specifies prompting for family, account, and project.

FAMILY or F

Specifies prompting for family name.

ACCOUNT or A

Specifies prompting for account number.

PROJECT or P

Specifies prompting for project number.

NONE

Specifies no prompting for family, account, and project.

The default is FAMILY.

STATUS

Returns the completion status of this command.

Remarks

- The CHANGE_NAM_ATTRIBUTES command must be entered at the NOS/VE system console.
- When statistics collection is enabled after having been disabled, the statistic values are reinitialized to zero.
- The statistics collected for NAM/VE are described in the NOS/VE System Performance and Maintenance manual, Volume 1.
- If more than a user name is entered during login, the system assumes that all login validation information has been entered and no additional prompts are issued. If only a user name is entered after the user prompt, the password prompt is issued followed by any prompts that are specified on the ADDITIONAL_LOGIN_PROMPTS parameter.

DISPLAY_NAM_ATTRIBUTES

Purpose Displays current NAM attribute values at the operator console.

Format

DISPLAY_NAM_ATTRIBUTES or
DISNA
DISPLAY_OPTIONS=list of keyword
OUTPUT=file
STATUS=status variable

Parameters

DISPLAY_OPTIONS or *DO*

Specifies the NAM/VE attribute or attributes for which the current values are to be displayed at the system console. All of these attributes except CURRENT_CONNECTIONS are listed in the description of the CHANGE_NAM_ATTRIBUTES command. The keyword value CURRENT_CONNECTIONS requests a display of the current number of network connections. If this value is greater than or equal to the value of the MAXIMUM_CONNECTIONS attribute, no new connections are allowed.

If you specify the keyword ALL, all attributes are displayed. The default is ALL.

OUTPUT or *O*

Specifies the file to which the generated output is to be written. The default is $OUTPUT.

STATUS

Returns the completion status of this command.

Remarks The DISPLAY_NAM_ATTRIBUTES command must be entered at the NOS/VE system console.

Examples The following is a sample DISPLAY_NAM_ATTRIBUTES display.

```
/display_nam_attributes
  Enable_Checksumming        : FALSE
  Enable_Statistics          : FALSE
  Maximum_Connections        : 256
  Current_Connections        : 55
  Maximum_ICA_Dumps          : 3
  Maximum_Login_Attempts     : 3
  ICA_Reset_Down_Threshold   : 10
  MCI_Reset_Down_Threshold   : 10
  Preferred_Protocol_Stack   : XNS
  Prompt_For_Login_Account   : FALSE
  Prompt_For_Login_Family    : TRUE
  Prompt_For_Login_Project   : FALSE
```

Using the MANAGE_STORE_FORWARD_NETWORK Utility 8

Using the MANAGE_STORE_FORWARD_NETWORK Utility 8

The MANAGE_STORE_FORWARD_NETWORK (MANSFN) utility is an SCL command utility. It follows the rules for utilities given in the NOS/VE System Usage manual.

Except for QUIT, all of the utility subcommands and directives have an optional STATUS parameter. Information about using the STATUS parameter is found in the NOS/VE System Usage manual.

You must be validated by the system administrator to use the MANSFN utility. Your validation must include the NETWORK_OPERATION capability or the NETWORK_APPLICATION_MANAGEMENT capability. For more information about validation, see the NOS/VE User Validation manual.

This chapter explains how to use the MANSFN utility to do the following.

- Create and install a MANSFN store-and-forward network file.
- Generate and change an existing MANSFN store-and-forward network file.
- Display information defined for a MANSFN store-and-forward network file.

Introduction to the MANSFN Utility

The MANAGE_STORE_FORWARD_NETWORK utility allows a site administrator to create store-and-forward configuration information. The Queue File Transfer Facility (QTF) then uses this configuration information to transfer queued files between arbitrarily named NOS/VE systems even if a non-VE system is somewhere in the destination path.

There are two reasons to use the MANSFN utility to create store-and-forward configuration information. First, queued files may need to be forwarded through non-VE systems to reach their destinations. The problem is that some systems (NOS is an example) use a queued file transfer protocol that can only transfer 3-character source and destination names. Thus, NOS/VE system names like SYSTEM_A and SYSTEM_B are truncated to SYS and are no longer unique within the network. Since it is impractical to rename all of the families and systems in several networks with 3-character names, the MANSFN utility can be used to create a file of configuration information that defines unique names as required throughout the network. This is done by creating a file of MANSFN directives that defines the required name substitutions. This file is called a *directives file.* The directives file is then installed using the MANSFN subcommand INSTALL_STORE_FORWARD_NETWORK. The installed directives file is then called a *store-and-forward network file.* This network file's system name is $SYSTEM.PTF_QTF.STORE_FORWARD_NETWORK. The MANSFN subcommands and directives are described later in this chapter.

The second reason to use the MANSFN utility to create configuration information is the need to use more than one queued file transfer application to reach the desired destination system. For example, assume the destination system is reachable only by way of the Network Transfer Facility (NTF) and the originating system is reachable

only by way of QTF. Further assume a bridge system has connections to both the originating and destination systems. In this case, the file transfer can be made by switching applications from QTF to NTF at the bridge system. This is done by using the MANSFN directive DEFINE_APPLICATION_NAME_SWITCH to define which application to use and when to use it. Again, this directive is part of a directives file that becomes the store-and-forward network file when the directives file is installed.

Although a new version of the store-and-forward network file may be *installed* at any time, it does not become *active* until the next time the QTF client and server applications are activated. When a QTF client application begins execution, it attaches the highest cycle of the store-and-forward network file and extracts initiator related store-and-forward network information from the file. This information is then passed to each initiator task as that task starts. Similarly, when a QTF server application begins execution, it attaches the highest cycle of the store-and-forward network file and extracts server related substitution network information from the file.

The MANSFN utility only creates the store-and-forward network file. It is the responsibility of the queued file transfer utilities to search this network file for the applicable directives.

Creating a MANSFN Store-and-Forward Network File

Use the following steps to create and install a new MANSFN store-and-forward network file:

1. Use the EDIT_FILE utility to create a file of MANSFN utility directives. The directives are described in the MANSFN Directives section in this chapter. For this example, assume the name of the directives file is SUB_DIRECTIVES.

2. Start the MANSFN utility by entering:

```
/manage_store_forward_network
```

3. Verify the directives file created in step 1 by using the VERIFY_STORE_FORWARD_NETWORK subcommand. If necessary, correct and reverify the file. For this example, enter:

```
msf/verify_store_forward_network input=sub_directives
```

4. Install the verified directives file by using the INSTALL_STORE_FORWARD_NETWORK subcommand. For this example, enter:

```
msf/install_store_forward_network input=sub_directives
```

5. Exit the MANSFN utility by using the QUIT subcommand.

See the NOS/VE File Editor manual for more information about the EDIT_FILE utility.

Updating a MANSFN Store-and-Forward Network File

Use the following steps to generate a file of existing MANSFN utility directives, modify the directives file, and install the file to create a new version of the store-and-forward network file:

1. Start the MANSFN utility by entering:

   ```
   /manage_store_forward_network
   ```

2. Display the information defined for the existing directives file by using the DISPLAY_STORE_FORWARD_NETWORK subcommand.

   ```
   msf/display_store_forward_network
   ```

   Examine the information and determine what changes you want to make.

3. Generate a file of the existing directives by using the GENERATE_STORE_FORWARD_NETWORK subcommand. For this example, assume the name of the output file is OLD_DIRECTIVES.

   ```
   msf/generate_store_forward_network output=old_directives
   ```

4. Modify the directives file (OLD_DIRECTIVES) by using the EDIT_FILE utility.

5. Verify the modified directives file by using the VERIFY_STORE_FORWARD_NETWORK subcommand. For this example, enter:

   ```
   msf/verify_store_forward_network input=old_directives
   ```

6. Install the verified directives file by using the INSTALL_STORE_FORWARD_NETWORK subcommand. For this example, enter:

   ```
   msf/install_store_forward_network input=old_directives
   ```

7. Exit the MANSFN utility by using the QUIT subcommand.

Accessing the MANSFN Store-and-Forward Network File

All applications that use the MANSFN store-and-forward network file ($SYSTEM.PTF_QTF.STORE_FORWARD_NETWORK) must attach the file in shared read mode so that other applications can also access the file. This is done by attaching the file using the ATTACH_FILE command and specifying SHARE_MODE=READ. For more information about this command, see the NOS/VE Commands and Functions manual.

MANSFN Command and Subcommands

This section describes the MANSFN utility command and its subcommands. These subcommands are used to display, generate, verify, and install the directives file. The directives are described in the MANSFN Directives section later in this chapter. See the Examples section later in this chapter for a detailed example of using the MANSFN utility.

MANAGE_STORE_FORWARD_NETWORK Command

Purpose Initiates execution of the MANAGE_STORE_FORWARD_NETWORK (MANSFN) utility.

Format **MANAGE_STORE_FORWARD_NETWORK** or
MANSFN
STATUS=status variable

Parameters *STATUS*

Returns the completion status of this command.

Remarks

- Only one site administrator at a time should use this utility to modify the store-and-forward network file.
- The prompt for the MANSFN utility is:

```
msf/
```

DISPLAY_STORE_FORWARD_NETWORK Subcommand

Purpose Displays information about MANSFN directives that corresponds with the contents of the highest cycle of the store-and-forward network file.

Format

DISPLAY_STORE_FORWARD_NETWORK or
DISSFN
NAME=name or *string*
APPLICATION_QUALIFIER=keyword
DISPLAY_OPTION=list of keyword
OUTPUT=file
STATUS=status variable

Parameters

NAME or *N*

Specifies a search name or string, such as a system name in a network. Only those items of information that contain this name or string are to be displayed. The default is to display information for all names or strings.

APPLICATION_QUALIFIER or *AQ*

Specifies an application keyword that defines the selection criteria for the display. Only those items of information that contain the specified application keyword (or for which an application qualifier is not applicable) are to be displayed. You can specify one of the following keywords:

NTFI

Reserved for future use.

NTFS

Reserved for future use.

QTFI

Display only those items of information concerning the client (initiator) component of the QTF application.

QTFS

Display only those items of information concerning the server component of the QTF application.

The default is to display information for all application qualifiers.

DISPLAY_OPTION or *DISPLAY_OPTIONS* or *DO*

Specifies the type of information to be displayed. You can specify one or more of the following keywords:

APPLICATION_NAME_SWITCH or ANS

Displays information associated with the DEFINE_APPLICATION_NAME_SWITCH directive.

DESTINATION_GROUP or DESTINATION_GROUPS or DG

Displays information associated with the DEFINE_DESTINATION_GROUP directive.

DESTINATION_NAME_SWITCH or DNS

Displays information associated with the DEFINE_DESTINATION_NAME_SWITCH directive.

SOURCE_NAME_SWITCH or SNS

Displays information associated with the DEFINE_SOURCE_NAME_SWITCH directive.

ALL

Displays information associated with all of the MANSFN directives.

The default is ALL.

OUTPUT or *O*

Specifies the file where the requested display information is to be written. The default is $OUTPUT.

STATUS

Returns the completion status of this subcommand.

Examples The following example displays information obtained by entering the keyword DESTINATION_NAME_SWITCH of the DISPLAY_OPTION parameter.

```
/manage_store_forward_network
msf/display_store_forward_network do=destination_name_switch


  DEFINE_DESTINATION_NAME_SWITCH  --  INFORMATION

    DESTINATION NAME                  : FAM_G
      NEXT_HOP_NAME                   : MMG
      APPLICATION_QUALIFIER           : QTFI

    DESTINATION NAME                  : FAM_H
      NEXT_HOP_NAME                   : MMH
      APPLICATION_QUALIFIER           : QTFI

    DESTINATION NAME                  : MMB
      NEXT_HOP_NAME                   : FAM_B
      APPLICATION_QUALIFIER           : QTFS

    DESTINATION NAME                  : MMC
      NEXT_HOP_NAME                   : FAM_C
      APPLICATION_QUALIFIER           : QTFS
```

GENERATE_STORE_FORWARD_NETWORK Subcommand

Purpose Generates a file of MANSFN directives that corresponds with the contents of the highest cycle of the store-and-forward network file.

Format

GENERATE_STORE_FORWARD_NETWORK or
GENSFN
OUTPUT=file
STATUS=status variable

Parameters

OUTPUT or **O**

Specifies the file where the generated MANSFN directives are to be written. This parameter is required.

STATUS

Returns the completion status of this subcommand.

INSTALL_STORE_FORWARD_NETWORK Subcommand

Purpose Installs the specified file of MANSFN directives and creates a new version of the store-and-forward network file $SYSTEM.PTF_QTF.STORE_FORWARD_NETWORK.

Format

INSTALL_STORE_FORWARD_NETWORK or
INSSFN
INPUT=file
ERROR=file
STATUS=status variable

Parameters

INPUT or **I**

Specifies the file containing the MANSFN directives that you want to install. This parameter is required.

ERROR or *E*

Specifies the file where the MANSFN directives are to be written along with any errors detected by this subcommand. The default is $ERRORS.

STATUS

Returns the completion status of this subcommand.

Remarks

- If the highest cycle of the store-and-forward network file is busy, a new cycle will be created; if it is not busy, the highest cycle of the file will be overwritten.
- The new version of the store-and-forward network file does not become active until the next time the QTF client and server applications are activated.
- Use the VERIFY_STORE_FORWARD_NETWORK subcommand to verify the MANSFN directives file before attempting to install the file.
- If errors are detected in the MANSFN directives file, the store-and-forward network file will not be created.
- Refer to appendix D for descriptions of error messages that may appear in the error file.

QUIT Subcommand

Purpose Ends a MANSFN utility session.

Format **QUIT** or
QUI

Parameters None.

VERIFY_STORE_FORWARD_NETWORK Subcommand

Purpose Verifies that the MANSFN directives in the input file form a syntactically correct and consistent set of store-and-forward network directives.

Format

VERIFY_STORE_FORWARD_NETWORK or
VERSFN
INPUT=file
OUTPUT=file
STATUS=status variable

Parameters

INPUT or **I**

Specifies the file containing the MANSFN directives that you want to verify. This parameter is required.

OUTPUT or *O*

Specifies the file where the MANSFN directives are to be written along with any error messages concerning the directives. The default is $OUTPUT.

STATUS

Returns the completion status of this subcommand.

Remarks

- Use this subcommand to verify the MANSFN directives file before attempting to install the file.
- Refer to appendix D for descriptions of error messages that may appear in the output file.

MANSFN Directives

This section describes the MANSFN utility directives. These directives are used to define mainframe name substitutions that are to be made for source and destination systems when forwarding queued files. These directives are also used to define destination groups and to define which queued file transfer applications to use when forwarding queued files. See the Examples section later in this chapter for a detailed example of using these directives.

DEFINE_APPLICATION_NAME_SWITCH Directive

Purpose Defines the application name substitution to be made when a specified application receives a queued file for a specified destination group.

Format

DEFINE_APPLICATION_NAME_SWITCH or
DEFANS
NEXT_HOP_APPLICATION=name
APPLICATION_QUALIFIER=list of keyword
DESTINATION_GROUP_QUALIFIER=name
STATUS=status variable

Parameters

NEXT_HOP_APPLICATION or **NHA**

Specifies the name of an application that will be responsible for forwarding a queued file on the next hop to its destination. This parameter is required.

APPLICATION_QUALIFIER or **AQ**

Specifies which file transfer server applications can make the name substitution. You can specify the following keywords:

NTFS

Reserved for future use.

QTFS

The server component of the QTF application makes the substitution.

This parameter is required.

DESTINATION_GROUP_QUALIFIER or *DGQ*

Specifies the name of a destination group. If a value is specified for this parameter, the application name substitution defined by this directive applies only to files destined for the specified group. If this parameter is not specified, the name substitution is made regardless of the file's destination.

STATUS

Returns the completion status of this directive.

DEFINE_DESTINATION_GROUP Directive

Purpose Defines a group of destination systems and assigns a name to the group. This group of destinations can then be referenced by a single name in other directives.

Format

DEFINE_DESTINATION_GROUP or
DEFDG
GROUP_NAME_=name
DESTINATION_NAMES=list of name or **string**
STATUS=status variable

Parameters

GROUP_NAME or **GN**

Specifies a name by which a group of destination systems can be referenced. This parameter is required.

DESTINATION_NAMES or **DN**

Specifies one or more names of destination systems to be included in the destination group. This parameter is required.

STATUS

Returns the completion status of this directive.

Remarks Group names must be unique within a host.

DEFINE_DESTINATION_NAME_SWITCH Directive

Purpose Defines the destination name substitution to be made when a specified application transfers a queued file to a destination system.

Format

DEFINE_DESTINATION_NAME_SWITCH or
DEFDNS
NAME=name or **string**
NEXT_HOP_NAME=name or **string**
APPLICATION_QUALIFIER=list of keyword
STATUS=status variable

Parameters

NAME or **N**

Specifies the name of a destination system that requires a name substitution. This parameter is required.

NEXT_HOP_NAME or **NHN**

Specifies the name to be used in place of the original destination name when transferring a queued file to a destination system. This parameter is required.

APPLICATION_QUALIFIER or **AQ**

Specifies one or more applications that can make the name substitution. You can specify the following keywords:

NTFI

Reserved for future use.

NTFS

Reserved for future use.

QTFI

The client (initiator) component of the QTF application makes the substitution before a file is transferred.

QTFS

The server component of the QTF application makes the substitution after a file is transferred.

This parameter is required.

STATUS

Returns the completion status of this directive.

DEFINE_SOURCE_NAME_SWITCH Directive

Purpose Defines the source name substitution to be made when a specified application transfers a queued file to a specified destination group.

Format

DEFINE_SOURCE_NAME_SWITCH or
DEFSNS
NAME=name or string
NEXT_HOP_NAME=name or string
APPLICATION_QUALIFIER=list of keyword
DESTINATION_GROUP_QUALIFIER=name
STATUS=status variable

Parameters

NAME or **N**

Specifies the name of the source system that requires a name substitution. This parameter is required.

NEXT_HOP_NAME or **NHN**

Specifies the name to be used in place of the original source name when transferring a queued file from the source system. This parameter is required.

APPLICATION_QUALIFIER or **AQ**

Specifies one or more applications that can make the name substitution. You can specify the following keywords:

NTFI

Reserved for future use.

NTFS

Reserved for future use.

QTFI

The client (initiator) component of the QTF application makes the substitution before a file is transferred.

QTFS

The server component of the QTF application makes the substitution after a file is transferred.

This parameter is required.

DESTINATION_GROUP_QUALIFIER or *DGQ*

Specifies the name of a destination group. If a value is specified for this parameter, the source name substitution defined by this directive applies only to files destined for the specified group. If this parameter is not specified, the name substitution is made regardless of the file's destination.

STATUS

Returns the completion status of this directive.

Examples

The following examples show how to use the MANSFN utility.

Example 1

This example uses the combination CDCNET and LCN network configuration shown in figure 8-1. In this example, the NOS system M07 uses a queued file transfer protocol that can only transfer 3-character source and destination names. Thus, NOS/VE system names like FAM_C and FAM_H get truncated to FAM when they are forwarded through the NOS system M07. To avoid this problem, use the MANSFN utility to define name substitutions for the adjacent NOS/VE systems.

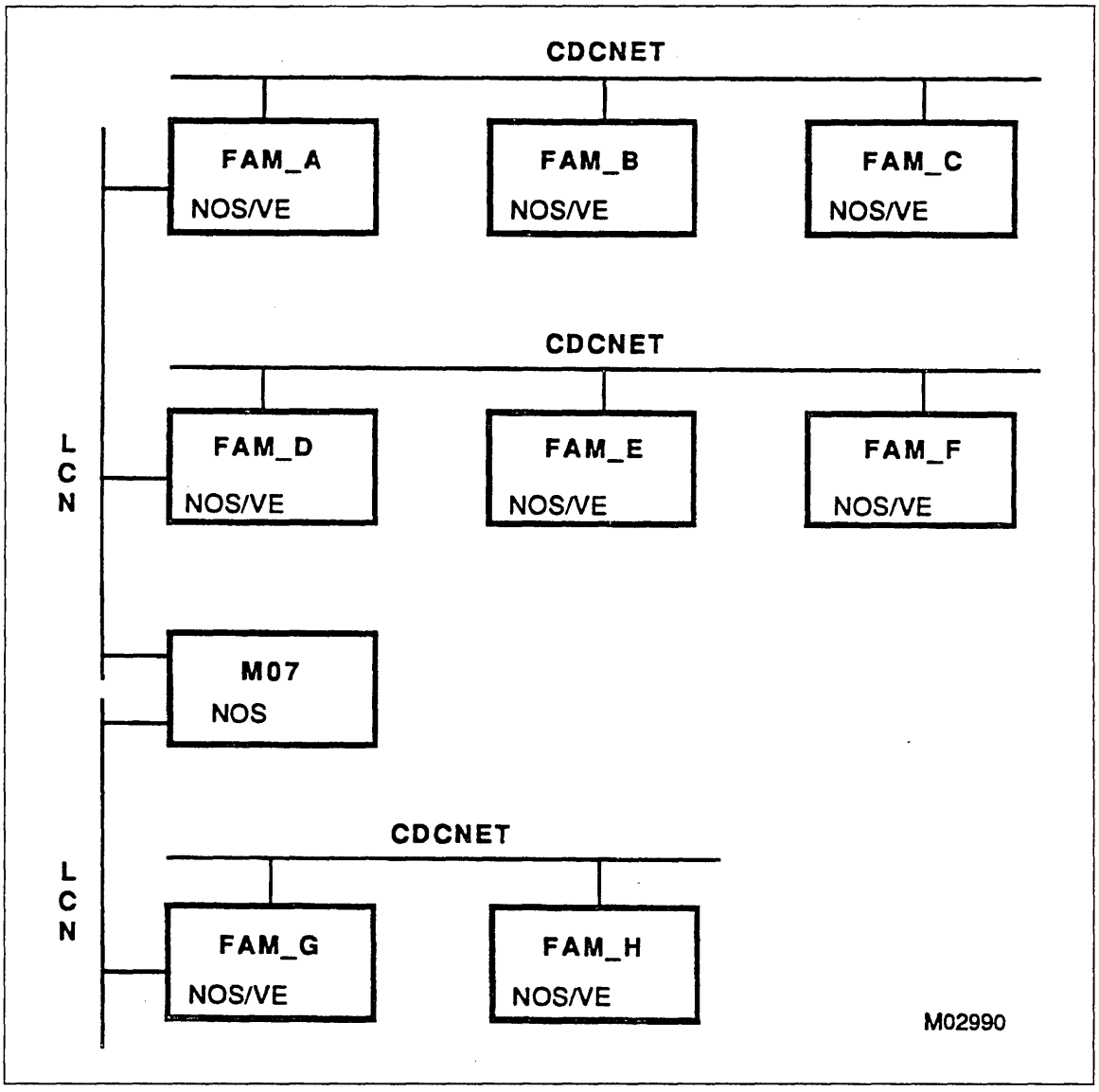


Figure 8-1. CDCNET/LCN Network Configuration

Define the following QTF server titles using the MANNA utility. See chapter 2, Understanding Network Applications, for more information about titles and title management.

FAM_A Titles	FAM_D Titles	FAM_G Titles
FAM_A	FAM_D	FAM_G
FAM_D	FAM_A	FAM_A
FAM_E	FAM_B	FAM_B
FAM_F	FAM_C	FAM_C
FAM_G	FAM_G	FAM_D
FAM_H	FAM_H	FAM_E
M07	M07	FAM_F
		M07

Define the following LIDs/PIDs using the MANRN utility. See the LCN Configuration and Network Management manual for information about the MANRN utility.

FAM_A

```
define_local_host pid='MMA' lid=('MMA','FAM_A')
define_remote_host pid='MMD' lid=('MMD','FAM_D','FAM_E','FAM_F')
define_remote_host pid='M07' lid=('M07','MMG','MMH')
```

FAM_D

```
define_local_host pid='MMD' lid=('MMD','FAM_D')
define_remote_host pid='MMA' lid=('MMA','FAM_A','FAM_B','FAM_C')
define_remote_host pid='M07' lid=('M07','MMG','MMH')
```

FAM_G

```
define_local_host pid='MMG' lid=('MMG','FAM_G')
define_remote_host pid='M07' lid=('M07','MMA','MMB','MMC', ..
  'MMD','MME','MMF')
```

Define the destination groups and the source and destination LID name substitutions using the following MANSFN utility directives.

1. Use the DEFINE_DESTINATION_GROUP directive to define destination groups.
2. Use the DEFINE_SOURCE_NAME_SWITCH directive to define name substitutions to be made by the initiator (client) for outgoing source LIDs.
3. Use the DEFINE_DESTINATION_NAME_SWITCH directive to define name substitutions to be made by the initiator for outgoing destination LIDs.
4. Use the DEFINE_DESTINATION_NAME_SWITCH directive to define name substitutions to be made by the server for incoming destination LIDs.
5. Use the DEFINE_SOURCE_NAME_SWITCH directive to define name substitutions to be made by the server for incoming source LIDs.

FAM_A Substitutions

```
define_destination_group gn=GROUP_1 dn=(FAM_B,FAM_C)
define_destination_group gn=GROUP_3 dn=(FAM_G,FAM_H)

define_source_name_switch n=FAM_A nhn=MMA aq=QTFI dgq=GROUP_3
define_source_name_switch n=FAM_B nhn=MMB aq=QTFI dgq=GROUP_3
define_source_name_switch n=FAM_C nhn=MMC aq=QTFI dgq=GROUP_3

define_destination_name_switch n=FAM_G nhn=MMG aq=QTFI
define_destination_name_switch n=FAM_H nhn=MMH aq=QTFI

define_destination_name_switch n=MMB nhn=FAM_B aq=QTFS
define_destination_name_switch n=MMC nhn=FAM_C aq=QTFS

define_source_name_switch n=MMG nhn=FAM_G aq=QTFS dgq=GROUP_1
define_source_name_switch n=MMH nhn=FAM_H aq=QTFS dgq=GROUP_1
```

FAM_D Substitutions

```
define_destination_group gn=GROUP_2 dn=(FAM_E,FAM_F)
define_destination_group gn=GROUP_3 dn=(FAM_G,FAM_H)

define_source_name_switch n=FAM_D nhn=MMD aq=QTFI dgq=GROUP_3
define_source_name_switch n=FAM_E nhn=MME aq=QTFI dgq=GROUP_3
define_source_name_switch n=FAM_F nhn=MMF aq=QTFI dgq=GROUP_3

define_destination_name_switch n=FAM_G nhn=MMG aq=QTFI
define_destination_name_switch n=FAM_H nhn=MMH aq=QTFI

define_destination_name_switch n=MME nhn=FAM_E aq=QTFS
define_destination_name_switch n=MMF nhn=FAM_F aq=QTFS

define_source_name_switch n=MMG nhn=FAM_G aq=QTFS dgq=GROUP_2
define_source_name_switch n=MMH nhn=FAM_H aq=QTFS dgq=GROUP_2
```

FAM_G Substitutions

```
define_destination_group gn=GROUP_12 dn=(FAM_A,FAM_B,FAM_C,FAM_D,FAM_E,FAM_F)

define_source_name_switch n=FAM_G nhn=MMG aq=QTFI dgq=GROUP_12
define_source_name_switch n=FAM_H nhn=MMH aq=QTFI dgq=GROUP_12

define_destination_name_switch n=FAM_A nhn=MMA aq=QTFI
define_destination_name_switch n=FAM_B nhn=MMB aq=QTFI
define_destination_name_switch n=FAM_C nhn=MMC aq=QTFI
define_destination_name_switch n=FAM_D nhn=MMD aq=QTFI
define_destination_name_switch n=FAM_E nhn=MME aq=QTFI
define_destination_name_switch n=FAM_F nhn=MMF aq=QTFI

define_destination_name_switch n=MMH nhn=FAM_H aq=QTFS

define_source_name_switch n=MMA nhn=FAM_A aq=QTFS
define_source_name_switch n=MMB nhn=FAM_B aq=QTFS
```

```
define_source_name_switch n=MMC nhn=FAM_C aq=QTFS
define_source_name_switch n=MMD nhn=FAM_D aq=QTFS
define_source_name_switch n=MME nhn=FAM_E aq=QTFS
define_source_name_switch n=MMF nhn=FAM_F aq=QTFS
```

For this example, assume you want to submit a job at system FAM_C, execute the job at system FAM_H, and return the output file to the originating system FAM_C.

The QTF initiator at FAM_C does not make any substitutions because none are defined for FAM_C. The job is sent to FAM_A for forwarding.

The QTF server at FAM_A does not make any substitutions because none of the defined substitutions apply. The QTF initiator at FAM_A substitutes source name MMC for FAM_C and substitutes destination name MMH for FAM_H. The job is sent to M07 and then forwarded to FAM_G.

The QTF server at FAM_G substitutes source name FAM_C for MMC and substitutes destination name FAM_H for MMH. The job is sent to its destination FAM_H, where it executes. The job's output is sent back to the originating system FAM_C.

For the return trip, the output file's origin is FAM_H and its destination is FAM_C. Since no substitutions are defined for the QTF initiator at FAM_H, it sends the output file to FAM_G for forwarding.

The QTF initiator at FAM_G substitutes destination name MMC for FAM_C and substitutes source name MMH for FAM_H. The output file is sent to M07 and then forwarded to FAM_A.

The QTF server at FAM_A substitutes source name FAM_H for MMH and substitutes destination name FAM_C for MMC. The QTF initiator at FAM_A does not make any substitutions because none of the defined substitutions apply. The output file is sent to its destination FAM_C, where it is put into the output queue for printing.

Example 2

This example shows how to use the MANSFN utility to switch applications when it is necessary to use more than one file transfer application to reach the desired destination. Assume the destination system can be reached only by NTF and the source system can be reached only by QTF. Further assume that you want to send a file to one of the destination systems within the destination group called GRAY_GROUP. Use the following MANSFN utility directives to define the destination group and the application name switch.

```
define_destination_group gn=GRAY_GROUP dn=(GRAY_A,GRAY_B,GRAY_C)
define_application_name_switch nha=NTF aq=QTFS dgq=GRAY_GROUP
```

When the QTF server receives a file with a destination of GRAY_A, GRAY_B, or GRAY_C, the server switches the application name to NTF. Thus, the NTF application is responsible for forwarding the file on its next hop to its destination.

Glossary A

Glossary A

A

Application Title

The logical name or names by which a network server application is known to the client.

Asynchronous TIP

The terminal interface program (TIP) that configures terminal devices and establishes terminal attributes for a generic asynchronous terminal connected to a DI. The asynchronous TIP resides in a DI that is configured to support asynchronous terminals.

B

Batch Transfer Facility (BTF)

A group of network applications that allow input to be routed from I/O stations or batch streams to a host computer and output to be routed to designated I/O stations or batch streams.

BTF

See Batch Transfer Facility.

C

Catenet

See Concatenated Network

CDCNET

Control Data Distributed Communications Network. CDCNET is the collection of hardware and software products offered by Control Data to interconnect computer resources (that is, host computer systems, terminals, workstations, unit record devices, communications media, and CDCNET device interfaces) into distributed communications networks. These hardware and software products are compatible with the Control Data Network Architecture (CDNA).

CDNA

Control Data Network Architecture. CDNA is the network architecture designed by Control Data to support the recommendations of the International Standards Organization's Open System Interconnection (OSI) model.

Client Application

Any network application that is authorized to initiate a connection to a server application. See also Server Application.

Communication Line

A terminal line that establishes a complete communication circuit between a terminal or workstation and a DI.

Concatenated Network

A communications network composed of more than one type of communications medium. It is often established when it is necessary to interconnect a local area network with other resources (for example, another local area network or geographically remote computer-related resources). It is also called a catenet.

Connection Request Validation

The validation performed by the server application (or by NAM/VE) when the client requests a connection to the server.

Cycle

A numbered version of a file that can be registered with the same file name and in the same catalog as the other versions of the file. Each permanent file can have from 1 to 999 cycles.

D

Device Interface (DI)

The communications processor used by Control Data as its CDCNET hardware product.

DI

See Device Interface.

F

File Transfer Protocol (FTP)

The Control Data application-to-application protocol that enables applications programs executing on one host system to exchange information with applications programs that execute on other host systems.

FTP

See File Transfer Protocol.

H

Host System

A mainframe computer and its operating system connected to a communications network that provides services such as database access, execution of user applications, and program compilation.

I

Instance

One execution of an application.

M

Management Entity (ME)

CDCNET software that performs network management functions. There are various MEs to perform specific network tasks.

ME

See Management Entity.

Multiconnection Application

A network application for which one instance of execution can support more than one connection.

N

NAM/VE

See Network Access Method/Virtual Environment.

NAM-Initiated Server Job.

A server application job that is initiated by NAM/VE. See also Non-NAM/initiated Server Job.

Network

An interconnected set of host computer systems, terminals, workstations, and unit record equipment.

Network Access Method/Virtual Environment

The access method that resides on NOS/VE and allows network applications programs on the host to exchange information with communications networks. NAM/VE conforms to the CDNA.

Network Address

A network address specifies the location of an application within a network. A network address consists of three parts: a network identifier, a system identifier, and an application identifier.

Network Application

An application that can communicate with peer applications over a data path called a network connection.

Network Connection

A data path over which applications communicate.

Network Transfer Facility (NTF)

The network applications that support fully-symetrical queued file transfers between a NOS/VE host and another host within the network. NTF supports IBM's Network Job Entry (NJE) protocol and HASP multileaving protocol for communication between hosts.

Non-NAM/Initiated Server Job

A server job whose corresponding server application contains mechanisms to initiate the server application job.

NTF

See Network Transfer Facility

P

Permanent File Transfer Facility (PTF)

The network applications that support file transfers between NOS/VE systems on the network.

Protocol

A set of conventions that must be followed for communications between the computer-related resources in a network.

PTF

See Permanent File Transfer Facility.

Q

Queue File Transfer Facility (QTF)

The network applications that support queued file transfers between host systems on the network.

QTF

See Queue File Transfer Facility

R

Ring

The level of hardware protection given a file or segment. A file is protected from unauthorized access by tasks executing in higher rings.

S

Server Application

A network application that is authorized to accept connections from clients. See also Client Application.

Server Application Job

The job submitted to the system that performs the services provided by the requested server application.

Server Job Validation

The validation performed when the server application job is submitted to the system. For non-NAM/initiated servers, the validation information must be a LOGIN command that is the first command in the server job. For NAM-initiated servers, you can specify a source for the validation information.

Single-Connection Application

A network application for which one instance of execution can support only one connection.

T

T-to-A

See Terminal-to-Application Processing.

Terminal-to-Application Processing

A type of network processing that enables the exchange of data between applications programs that reside on host computers and user terminals or workstations. In this case, protocol conversions occur so that transmitted data is understood both at the host and the terminal or workstation.

Terminal Interface Program (TIP)

A program that provides an interface for terminals connected to a DI. A TIP defines line and terminal attributes.

Timesharing Service

The NOS/VE network application that allows interactive terminals to connect to NOS/VE.

TIP

See Terminal Interface Program.

Related Manuals B

Related Manuals B

All NOS/VE, related NOS (or NOS/BE manuals), and related hardware manuals are listed in table B-1. If your site has installed the online manuals, you can find an abstract for each NOS/VE manual in the online System Information manual. To access this manual, enter:

```
/explain
```

Ordering Printed Manuals

To order a printed Control Data manual, send an order form to:

Control Data Corporation
Literature and Distribution Services
308 North Dale Street
St. Paul, Minnesota 55103

To obtain an order form or to get more information about ordering Control Data manuals, write to the above address or call (612) 292-2101. If you are a Control Data employee, call (612) 292-2100.

Accessing Online Manuals

To access the online version of a printed manual, log in to NOS/VE and enter the online title on the EXPLAIN command (table B-1 supplies the online titles). For example, to see the NOS/VE Commands and Functions manual, enter:

```
/explain manual=scl
```

The examples in some printed manuals exist also in the online Examples manual. To access this manual, enter:

```
/explain manual=examples
```

When EXAMPLES is listed in the Online Manuals column in table B-1, that manual is represented in the online Examples manual.

Table B-1. Related Manuals

Manual Title	Publication Number	Online Manuals[1]
NOS/VE Site Manuals:		
CYBER 930 Computer System Guide to Operations Usage	60469560	
CYBER Initialization Package (CIP) Reference Manual	60457180	
Desktop/VE Host Utilities Usage	60463918	
MAINTAIN_MAIL[2] Usage		MAIM
NOS/VE Accounting Analysis System Usage	60463923	
NOS/VE Accounting and Validation Utilities for Dual State Usage	60458910	
NOS/VE LCN Configuration and Network Management Usage	60463917	
NOS/VE Network Management Usage	60463916	
NOS/VE Operations Usage	60463914	
NOS/VE System Performance and Maintenace Volume 1: Performance Usage	60463915	
NOS/VE System Performance and Maintenance Volume 2: Maintenance Usage	60463925	
NOS/VE User Validation Usage	60464513	

1. This column lists the title of the online version of the manual and indicates whether the examples in the printed manual are in the online Examples manual.

2. To access this manual, you must be the administrator for MAIL/VE.

(Continued)

Table B-1. Related Manuals *(Continued)*

Manual Title	Publication Number	Online Manuals[1]
NOS/VE User Manuals:		
EDIT_CATALOG Usage		EDIT_ CATALOG
EDIT_CATALOG for NOS/VE Summary	60487719	
Introduction to NOS/VE Tutorial	60464012	
NOS/VE Advanced File Management Tutorial	60486412	AFM_T
NOS/VE Advanced File Management Usage	60486413	AFM
NOS/VE Advanced File Management Summary	60486419	
NOS/VE Commands and Functions Quick Reference	60464018	SCL
NOS/VE File Editor Tutorial/Usage	60464015	EXAMPLES
NOS/VE Object Code Management Usage	60464413	
NOS/VE Screen Formatting Usage	60488813	EXAMPLES
NOS/VE Source Code Management Usage	60464313	EXAMPLES
NOS/VE System Usage	60464014	EXAMPLES
NOS/VE Terminal Definition Usage	60464016	
Screen Design Facility for NOS/VE Usage	60488613	SDF

1. This column lists the title of the online version of the manual and indicates whether the examples in the printed manual are in the online Examples manual.

(Continued)

Table B-1. Related Manuals *(Continued)*

Manual Title	Publication Number	Online Manuals[1]
CYBIL Manuals:		
CYBIL for NOS/VE File Management Usage	60464114	EXAMPLES
CYBIL for NOS/VE Keyed-File and Sort/Merge Interfaces Usage	60464117	EXAMPLES
CYBIL for NOS/VE Language Definition Usage	60464113	CYBIL and EXAMPLES
CYBIL for NOS/VE Sequential and Byte-Addressable Files Usage	60464116	EXAMPLES
CYBIL for NOS/VE System Interface Usage	60464115	EXAMPLES
FORTRAN Manuals:		
FORTRAN Version 1 for NOS/VE Language Definition Usage	60485913	EXAMPLES
FORTRAN Version 1 for NOS/VE Quick Reference		FORTRAN
FORTRAN Version 2 for NOS/VE Language Definition Usage	60487113	EXAMPLES
FORTRAN Version 2 for NOS/VE Quick Reference		VFORTRAN
FORTRAN for NOS/VE Tutorial	60485912	FORTRAN_T
FORTRAN for NOS/VE Topics for FORTRAN Programmers Usage	60485916	
FORTRAN for NOS/VE Summary	60485919	

1. This column lists the title of the online version of the manual and indicates whether the examples in the printed manual are in the online Examples manual.

(Continued)

Table B-1. Related Manuals *(Continued)*

Manual Title	Publication Number	Online Manuals[1]
COBOL Manuals:		
COBOL for NOS/VE Summary	60486019	
COBOL for NOS/VE Tutorial	60486012	COBOL_T
COBOL for NOS/VE Usage	60486013	COBOL and EXAMPLES
Other Compiler Manuals:		
APL for NOS/VE File Utilities Usage	60485814	
APL for NOS/VE Language Definition Usage	60485813	
BASIC for NOS/VE Summary Card	60486319	
BASIC for NOS/VE Usage	60486313	BASIC
LISP for NOS/VE Usage	60486213	
Pascal for NOS/VE Summary Card	60485619	
Pascal for NOS/VE Usage	60485613	PASCAL
Prolog for NOS/VE Quick Reference	60486718	PROLOG
Prolog for NOS/VE Usage	60486713	

1. This column lists the title of the online version of the manual and indicates whether the examples in the printed manual are in the online Examples manual.

(Continued)

Table B-1. Related Manuals *(Continued)*

Manual Title	Publication Number	Online Manuals[1]
VX/VE Manuals:		
C/VE for NOS/VE Quick Reference		C
C/VE for NOS/VE Usage	60469830	
DWB/VX Introduction and User Reference Tutorial/Usage	60469890	
DWB/VX Macro Packages Guide Usage	60469910	
DWB/VX Preprocessors Guide Usage	60469920	
DWB/VX Text Formatters Guide Usage	60469900	
VX/VE Administrator Guide and Reference Tutorial/Usage	60469770	
VX/VE An Introduction for UNIX Users Tutorial/Usage	60469980	
VX/VE Programmer Guide Tutorial	60469790	
VX/VE Programmer Reference Usage	60469820	
VX/VE Support Tools Guide Tutorial	60469800	

1. This column lists the title of the online version of the manual and indicates whether the examples in the printed manual are in the online Examples manual.

(Continued)

Table B-1. Related Manuals *(Continued)*

Manual Title	Publication Number	Online Manuals[1]
VX/VE Manuals (Continued):		
VX/VE User Guide Tutorial	60469780	
VX/VE User Reference Usage	60469810	
Data Management Manuals:		
DM Command Procedures Reference Manual	60487905	
DM Concepts and Facilities Manual	60487900	
DM Error Message Summary for DM on CDC NOS/VE	60487906	
DM Fundamental Query and Manipulation Manual	60487903	
DM Report Writer Reference Manual	60487904	
DM System Administrator's Reference Manual for DM on CDC NOS/VE	60487902	
DM Utilities Reference Manual for DM on CDC NOS/VE	60487901	
Information Management Manuals:		
IM/Control for NOS/VE Quick Reference	L60488918	CONTROL
IM/Control for NOS/VE Usage	60488913	
IM/Quick for NOS/VE Tutorial	60485712	

1. This column lists the title of the online version of the manual and indicates whether the examples in the printed manual are in the online Examples manual.

(Continued)

Table B-1. Related Manuals *(Continued)*

Manual Title	Publication Number	Online Manuals[1]
Information Management Manuals (Continued):		
IM/Quick for NOS/VE Summary	60485714	
IM/Quick for NOS/VE Usage		QUICK
CDCNET Manuals:		
CDCNET Access Guide	60463830	CDCNET_ ACCESS
CDCNET Batch Device User Guide	60463863	CDCNET_ BATCH
CDCNET Commands Quick Reference	60000020	
CDCNET Configuration and Site Administration Guide	60461550	
CDCNET Diagnostic Messages	60461600	
CDCNET Conceptual Overview	60461540	
CDCNET Network Analysis	60461590	
CDCNET Network Configuration Utility		NETCU
CDCNET Network Configuration Utility Summary Card	60000269	
CDCNET Network Operations	60461520	
CDCNET Network Performance Analyzer	60461510	
CDCNET Product Descriptions	60460590	
CDCNET Systems Programmer's Reference Manual Volume 1 Base System Software	60462410	
CDCNET Systems Programmer's Reference Manual Volume 2 Network Management Entities and Layer Interfaces	60462420	

1. This column lists the title of the online version of the manual and indicates whether the examples in the printed manual are in the online Examples manual.

(Continued)

Table B-1. Related Manuals *(Continued)*

Manual Title	Publication Number	Online Manuals[1]
CDCNET Manuals (Continued):		
CDCNET Systems Programmer's Reference Manual Volume 3 Network Protocols	60462430	
CDCNET Terminal Interface Usage	60463850	
NOS Version 2 Manuals:		
NOS 2 Installation Handbook	60459320	
NOS 2 Operations Handbook	60459310	
NOS 2 Analysis Handbook	60459300	
NOS/BE Version 1.5 Manuals:		
NOS/BE Installation Handbook	60494300	
NOS/BE Operator's Guide	60493900	
Miscellaneous Manuals:		
Applications Directory	60455370	
CONTEXT Summary Card	60488419	
CYBER Online Text for NOS/VE Usage	60488403	CONTEXT
Control Data CONNECT User's Guide	60462560	
Debug for NOS/VE Quick Reference		DEBUG
Debug for NOS/VE Usage	60488213	
Desktop/VE for Macintosh Tutorial	60464502	
Desktop/VE for Macintosh Usage	60464503	

1. This column lists the title of the online version of the manual and indicates whether the examples in the printed manual are in the online Examples manual.

(Continued)

Table B-1. Related Manuals *(Continued)*

Manual Title	Publication Number	Online Manuals[1]
Miscellaneous Manuals (Continued):		
MAIL/VE Summary Card	60464519	
MAIL/VE Usage		MAIL_VE
NOS Online Maintenance Software Reference Manual	60454200	
NOS/VE Diagnostic Messages Usage	60464613	MESSAGES
NOS/VE Examples Usage		EXAMPLES
NOS/VE System Information		NOS_VE
Programming Environment for NOS/VE Usage		ENVIRON-MENT
Programming Environment for NOS/VE Summary	60486819	
Remote Host Facility Usage	60460620	
Hardware Manuals:		
CYBER 170 Computer Systems Models 825, 835, and 855 General Description Hardware Reference	60459960	
CYBER 170 Computer Systems, Models 815, 825, 835, 845, and 855 CYBER 180 Models 810, 830, 835, 840, 845, 850, 855, and 860 Codes Booklet	60458100	
CYBER 170 Computer Systems, Models 815, 825, 835, 845, and 855 CYBER 180 Models 810, 830, 835, 840, 845, 850, 855, and 860 Maintenance Register Codes Booklet	60458110	

1. This column lists the title of the online version of the manual and indicates whether the examples in the printed manual are in the online Examples manual.

(Continued)

Table B-1. Related Manuals *(Continued)*

Manual Title	Publication Number	Online Manuals[1]
Hardware Manuals (Continued):		
HPA/VE Reference	60461930	
Virtual State Volume II Hardware Reference	60458890	
7021-31/32 Advanced Tape Subsystem Reference	60449600	
7221-1 Intelligent Small Magnetic Tape Subsystem Reference	60461090	

1. This column lists the title of the online version of the manual and indicates whether the examples in the printed manual are in the online Examples manual.

Default Network Application Definitions C

Default Network Application Definitions C

This appendix gives the default application definitions for network applications created by the NOS/VE installation process. This appendix also explains how to activate the definition using MANNA subcommands.

Timesharing Application

The define_timesharing_application option of the DEFINE_NETWORK command produces the following subcommands for defining the Timesharing application:

```
define_server server=osa$timesharing protocol=cdna_virtual_terminal ..
    nam_initiated=true
  add_titles titles="title or titles supplied by the site"
  change_maximum_connections maximum_connections=200
  change_connection_priority connection_priority=6
  change_server_validation capability=none ring=3 system_privilege=false
  change_client_validation capability=timesharing
  change_server_job job=$null validation_source=client maximum_connections=1
end_define_server
```

The Timesharing application is activated by the MANNA subcommand:

```
activate_server osa$timesharing
```

Desktop Environment (Desktop/VE) Application

The DEFINE_DESKTOP_ENVIRONMENT command produces the following subcommands for defining the Desktop Environment client application:

```
define_client client=desktop_ve protocol=cdna_virtual_terminal
end_define_client
```

This application is activated by the MANNA subcommand:

```
activate_client client=desktop_ve
```

Permanent File Transfer Facility (PTF) Applications

Client Application

The DEFINE_PTF command produces the following subcommands for defining the Permanent File Transfer Facility client application:

```
define_client client=osa$file_transfer_client protocol=cdna_session
  change_maximum_connections maximum_connections=40
  change_client_validation system_privilege=true
  change_application_identifier application_identifier=2001
  change_connection_priority connection_priority=4
end_define_client
```

This application is activated by the MANNA subcommand:

```
activate_client client=osa$file_transfer_client
```

Server Application

The DEFINE_PTF command produces the following subcommands for defining the Permanent File Transfer Facility server application:

```
define_server server=osa$file_transfer_server protocol=cdna_session ..
    nam_initiated=false
  add_titles title=ptfs$xxx "Where xxx is the name by which remote
    users access this server. The title or list of titles is supplied
    by the site."
  change_maximum_connections maximum_connections=40
  change_server_validation system_privilege=true
  add_client_address system_identifier=nosve application_identifier=2001
  add_client_address system_identifier=cdcnet "allows any NOS gateway DI"
  change_accept_connection accept_connection=false
  change_connection_priority connection_priority=4
end_define_server
```

This application is activated by the MANNA subcommand:

```
activate_server server=osa$file_transfer_server
```

Queue File Transfer Facility (QTF) Applications

Client Application

The DEFINE_QTF command produces the following subcommands for defining the Queue File Transfer Facility client application:

```
define_client client=osa$queue_transfer_client protocol=cdna_session
  change_connection_priority connection_priority=4
  change_maximum_connections maximum_connections=40
  change_client_validation system_privilege=true
  change_application_identifier application_identifier=2006
end_define_client
```

This application is activated by the MANNA subcommand:

```
activate_client client=osa$queue_transfer_client
```

Server Application

The DEFINE_QTFS command produces the following subcommands for defining the Queue File Transfer Facility server application:

```
define_server server=osa$queue_transfer_server protocol=cdna_session ..
     nam_initiated=false
  change_connection_priority connection_priority=4
  change_maximum_connections maximum_connections=40
  change_server_validation system_privilege=true
  change_accept_connection accept_connection=false
  add_client_address system_identifier=nosve application_identifier=2006
  add_client_address system_identifier=cdcnet application_identifier=all
  add_titles qtfs$xxxx broadcast_registration=true "Where xxxx is the
    name by which remote users access this server. The title or list of titles
    is supplied by the site."
end_define_server
```

This application is activated by the MANNA subcommand:

```
activate_server server=osa$queue_transfer_server
```

Batch Transfer Applications

Status and Control Facility (SCF) Application

The DEFINE_SCF command produces the following subcommands for defining the Status and Control Facility client application:

```
define_client client=osa$status_control_fac_client protocol=cdna_session
  change_application_identifier application_identifier=2002 "SCF/VE identifier"
  change_maximum_connections maximum_connections=20
  change_client_validation system_privilege=true
  change_connection_priority connection_priority=5
end_define_client
```

This application is activated by the MANNA subcommand:

```
activate_client client=osa$status_control_fac_client
```

Batch Transfer Facility (BTF) Application

The DEFINE_BTF command produces the following subcommands for defining the Batch Transfer Facility client application:

```
define_client client=osa$batch_transfer_client protocol=cdna_session
  change_maximum_connections maximum_connections=20
  change_client_validation system_privilege=true
  change_connection_priority connection_priority=2
end_define_client
```

This application is activated by the MANNA subcommand:

```
activate_client client=osa$batch_transfer_client
```

Operate Station Utility (OPES) Application

The DEFINE_OPES command produces the following subcommands for defining the Operate Station Utility application:

```
define_client client=osa$station_operator protocol=cdna_session
  change_application_identifier application_identifier=2003 "OPES identifer"
  change_maximum_connections maximum_connections=20
  change_connection_priority connection_priority=5
end_define_client
```

This application is activated by the MANNA subcommand:

```
activate_client client=osa$station_operator
```

Status and Control Facility Server (SCFS) Application

The DEFINE_SCFS command produces the following subcommands for defining the Status and Control Facility Server application:

```
define_server osa$status_control_fac_server protocol=cdna_session ..
    nam_initiated=false
  change_maximum_connections maximum_connections=40
  change_server_validation system_privilege=true
  add_client_address application_identifier=2002 "SCF/VE identifier"
  add_client_address application_identifier=2003 "OPES identifier"
  add_client_address application_identifier=2004 "NTF identifier"
  add_client_address application_identifier=2005 "OPENTF identifier"
  add_client_address system_identifier=cdcnet application_identifier=all
  add_server_managed_title title_pattern='SCF[AS]$*'
  change_accept_connection accept_connection=false
  change_connection_priority connection_priority=5
end_define_server
```

This application is activated by the MANNA subcommand:

```
activate_server server=osa$status_control_fac_server
```

Batch Transfer Facility Server (BTFS) Application

The DEFINE_BTFS command produces the following subcommands for defining the Batch Transfer Facility Server application:

```
define_server server=osa$batch_transfer_server protocol=cdna_session ..
    nam_initiated=false
  add_title title=btfs$xxx "Where xxx is the job destination name used
    to access this server.  The title or list of titles is supplied by
    the site."
  change_maximum_connections maximum_connections=40
  change_server_validation system_privilege=false
  add_client_address system_identifier=cdcnet application_identifier=all
  change_accept_connection accept_connection=false
  change_connection_priority connection_priority=2
end_define_server
```

This application is activated by the MANNA subcommand:

```
activate_server server=osa$batch_transfer_server
```

Network Transfer Facility (NTF) Applications

Network Transfer Facility (NTF) Application

The DEFINE_NTF command produces the following subcommands for defining the Network Transfer Facility application:

```
define_client client=osa$network_transfer_fac_client protocol=cdna_session
  change_application_identifier application_identifier=2004 "NTF identifier"
  change_maximum_connections maximum_connections=20
  change_client_validation system_privilege=true
  change_connection_priority connection_priority=5
end_define_client
```

This application is activated by the MANNA subcommand:

```
activate_client client=osa$network_tansfer_fac_client
```

Operate Network Transfer Facility (OPENTF) Application

The DEFINE_OPENTF command produces the following subcommands for defining the Operate Network Transfer Facility application:

```
define_client osa$ntf_operator protocol=cdna_session
  change_applications_identifier application_identifier=2005 ..
    "OPENTF identifier"
  change_maximum_connections maximum_connections=20
  change_connection_priority connection_priority=5
end_define_client
```

This application is activated by the MANNA subcommand:

```
activate_client client=osa$ntf_operator
```

File Transfer Protocol (FTP) Applications

Client Application

The DEFINE_FTP command produces the following subcommands for defining the File Transfer Protocol client application:

```
define_client client=osa$ftp_client protocol=cdna_session
  change_maximum_connections maximum_connections=40
end_define_client
```

This application is activated by the MANNA subcommand:

```
activate_client client=osa$ftp_client
```

Server Application

The DEFINE_FTPS command produces the following subcommands for defining the File Transfer Protocol server application:

```
define_client client=osa$ftp_server protocol=cdna_session
  change_maximum_connections maximum_connections=40
  change_client_validation system_privilege=true
end_define_client
```

This application is activated by the MANNA subcommand:

```
activate_client client=osa$ftp_server
```

Simple Mail Transfer Protocol (SMTP) Applications

Client Application

The DEFINE_SMTP command produces the following subcommands for defining the Simple Mail Transfer Protocol client application:

```
define_client client=osa$smtp_client protocol=cdna_session
  change_maximum_connections maximum_connections=100
end_define_client
```

This application is activated by the SCL procedure:

```
activate_smtp
```

Server Application

The DEFINE_SMTPS command produces the following subcommands for defining the Simple Mail Transfer Protocol server application:

```
define_client client=osa$smtp_server protocol=cdna_session
  change_maximum_connections maximum_connections=100
  change_client_validation system_privilege=true
end define_client
```

This application is activated as a system task by the Internet Daemon (INETD) when an SMTP connection request is received on port 25 (the Simple Mail Transfer Protocol well-known port).

Interprocess Communication (IPC) Application

The DEFINE_IPC_APPLICATIONS command produces the following subcommands for defining the Interprocess Communication client application:

```
define_client client=osa$ipc_applications protocol=cdna_session
  change_maximum_connections maximum_connections=40
  change_client_validation system_privilege=true
end_define_client
```

This application is activated as a system job by the MANNA subcommand:

```
activate_client client=osa$ipc_applications
```

Error Messages D

Error Messages D

Diagnostic Description Format

The following information is provided for each message:

- The condition code.
- The full mode message text with variable fields represented as follows:

  {file} Represents a file name or file path contained in the actual message.

  {text} Represents replacement text contained in the actual message.
- The condition identifier, followed by a repetition of the condition code.
- A description of the message.
- Suggested user action to correct the problem.
- Further information (when applicable).

Condition identifiers (names) should be used instead of condition codes to test for specific conditions in SCL jobs and procedures. For example, condition code NF 600 has the condition identifier NFE$SF_CALLER_NOT_PRIVILEGED. Refer to condition processing in the NOS/VE System Usage manual for more information about structuring jobs and testing for abnormal conditions. Also, see the $CONDITION_NAME function in the NOS/VE System Usage manual for information about determining and using the condition identifier in an SCL job.

Error Messages

--FATAL NF 600-- Caller not validated to {text}.

Condition Identifier: NFE$SF_CALLER_NOT_PRIVILEGED Code: NF 600

Description: To use the MANAGE_STORE_FORWARD_NETWORK utility, you must be validated for the capability of NETWORK_OPERATION or NETWORK_APPLICATION_MANAGEMENT.

User Action: Ask the site administrator to validate you for the NETWORK_OPERATION or NETWORK_APPLICATION_MANAGEMENT capability.

--INFORMATIVE NF 601-- Combine the multiple definitions of DEFINE_APPLICATION_NAME_SWITCH, that have the destination_group_qualifier value of {text}, next_hop_application value of {text} and the application_qualifiers are a subset of one or the other into one definition.

Condition Identifier: NFE$SF_COMBINE_DUP_APPL_DEF Code: NF 601

Description: The MANAGE_STORE_FORWARD_NETWORK utility has found multiple DEFINE_APPLICATION_NAME_SWITCH directives that contain a subset of identical information.

User Action: Combine all applicable directives into one directive.

--INFORMATIVE NF 602-- Combine the multiple definitions of DEFINE_SOURCE_NAME_SWITCH, that have the name value of {text}, next_hop_name value of {text}, destination_group_qualifier value of {text}, and the application_qualifiers are a subset of one another into one definition.

Condition Identifier: NFE$SF_COMBINE_DUP_SOURCE_DEF Code: NF 602

Description: The MANAGE_STORE_FORWARD_NETWORK utility has found multiple DEFINE_SOURCE_NAME_SWITCH directives that contain a subset of identical information.

User Action: Combine all applicable directives into one directive.

--INFORMATIVE NF 603-- Combine the multiple definitions of DEFINE_DESTINATION_NAME_SWITCH, that have the name value of {text}, next_hop_name value of {text} and the application_qualifiers are a subset of one another into one definition.

Condition Identifier: NFE$SF_COMBINE_DUP_TARGET_DEF Code: NF 603

Description: The MANAGE_STORE_FORWARD_NETWORK utility has found multiple DEFINE_DESTINATION_NAME_SWITCH directives that contain a subset of identical information.

User Action: Combine all applicable directives into one directive.

--FATAL NF 604-- The DESTINATION_GROUP_QUALIFIER {text} has not been defined.

Condition Identifier: NFE$SF_DEST_GROUP_NOT_FOUND Code: NF 604

Description: The destination group qualifier has not been defined.

User Action: Define the destination group by using the MANAGE_STORE_FORWARD_NETWORK directive DEFINE_DESTINATION_GROUP.

--FATAL NF 605-- The multiple definitions of DEFINE_APPLICATION_NAME_SWITCH, that have the destination_group_qualifier value of {text} and overlapping application_qualifiers have different next_hop_applications.

Condition Identifier: NFE$SF_DIF_NHA_SAME_AQ_AND_DGQ Code: NF 605

Description: The MANAGE_STORE_FORWARD_NETWORK utility has found multiple DEFINE_APPLICATION_NAME_SWITCH directives that contain identical values for the parameters APPLICATION_QUALIFIER and DESTINATION_GROUP_QUALIFIER but different values for the parameter NEXT_HOP_APPLICATION.

User Action: Change the NEXT_HOP_APPLICATION to the correct value or change the APPLICATION_QUALIFIER or DESTINATION_GROUP_QUALIFIER parameters to different values.

--FATAL NF 606-- The multiple definitions of DEFINE_SOURCE_NAME_SWITCH, that have the name value of {text}, destination_group_qualifier value of {text} and overlapping application_qualifiers have different next_hop_names.

Condition Identifier: NFE$SF_DIF_NHN_SAME_N_AQ_DGQ Code: NF 606

Description: The MANAGE_STORE_FORWARD_NETWORK utility has found multiple DEFINE_SOURCE_NAME_SWITCH directives that contain identical values for the parameters NAME, APPLICATION_QUALIFIER, and DESTINATION_GROUP_QUALIFIER but different values for the parameter NEXT_HOP_NAME.

User Action: Change the NEXT_HOP_NAME to the correct value or change the NAME or APPLICATION_QUALIFIER or DESTINATION_GROUP_QUALIFIER parameters to different values.

--FATAL NF 607-- The multiple definitions of DEFINE_DESTINATION_NAME_SWITCH, that have the name value of {text} and overlapping application_qualifiers have different next_hop_names.

Condition Identifier: NFE$SF_DIF_NHN_SAME_N_AND_AQ Code: NF 607

Description: The MANAGE_STORE_FORWARD_NETWORK utility has found multiple DEFINE_DESTINATION_NAME_SWITCH directives that contain identical values for the parameters NAME and APPLICATION_QUALIFIER but different values for the parameter NEXT_HOP_NAME.

User Action: Change the NEXT_HOP_NAME to the correct value or change the NAME or APPLICATION_QUALIFIER parameters to different values.

--FATAL NF 608-- MANAGE_STORE_FORWARD_NETWORK has encountered directive errors, see file: {text} for additional error information.

Condition Identifier: NFE$SF_DIRECTIVE_ERRORS Code: NF 608

Description: The MANAGE_STORE_FORWARD_NETWORK utility has encountered fatal directive errors that cannot be processed.

User Action: Look at the output file for the MANAGE_STORE_FORWARD_NETWORK utility for additional information about the directive errors, and then correct the errors.

--FATAL NF 609-- The DEFINE_APPLICATION_NAME_SWITCH, that have the destination_group_qualifier value of {text} and next_hop_application value of {text} has multiple definitions with overlapping application_qualifiers.

Condition Identifier: NFE$SF_DUPLICATE_APPL_DEF Code: NF 609

Description: The MANAGE_STORE_FORWARD_NETWORK utility has found multiple DEFINE_APPLICATION_NAME_SWITCH directives for the same NEXT_HOP_APPLICATION and DESTINATION_GROUP_QUALIFIER values but different APPLICATION_QUALIFIER values.

User Action: Delete or change the invalid directive.

--FATAL NF 610-- The DEFINE_DESTINATION_GROUP, that has the group_name value of {text} has multiple definitions.

Condition Identifier: NFE$SF_DUPLICATE_GROUP_NAMES Code: NF 610

Description: The MANAGE_STORE_FORWARD_NETWORK utility has found multiple DEFINE_DESTINATION_GROUP directives for the same GROUP_NAME value.

User Action: Delete or change the invalid directive.

--FATAL NF 611-- The DEFINE_SOURCE_NAME_SWITCH, that have the name value of {text} and next_hop_name value of {text} has multiple definitions with overlapping application_qualifiers and destination_group_qualifiers.

Condition Identifier: NFE$SF_DUPLICATE_SOURCE_DEF Code: NF 611

Description: The MANAGE_STORE_FORWARD_NETWORK utility has found multiple DEFINE_SOURCE_NAME_SWITCH directives for the same NAME and NEXT_HOP_NAME values but different APPLICATION_QUALIFIER and DESTINATION_GROUP_QUALIFIER values.

User Action: Delete or change the invalid directive.

--FATAL NF 612-- The DEFINE_DESTINATION_NAME_SWITCH, that have the name value of {text} and next_hop_name value of {text} has multiple definitions with overlapping application_qualifiers.

Condition Identifier: NFE$SF_DUPLICATE_TARGET_DEF Code: NF 612

Description: The MANAGE_STORE_FORWARD_NETWORK utility has found multiple DEFINE_ DESTINATION_NAME_SWITCH directives for the same NAME and NEXT_HOP_NAME values but different APPLICATION_QUALIFIER . values.

User Action: Delete or change the invalid directive.

--INFORMATIVE NF 613-- The DEFINE_DESTINATION_GROUP directive for the group_name {text} was not used as a destination_group_qualifier in any other directives.

Condition Identifier: NFE$SF_GROUP_NAME_NOT_USED Code: NF 613

Description: The MANAGE_STORE_FORWARD_NETWORK utility DEFINE_DESTINATION_GROUP directive is not used in any other directive.

User Action: Delete the unused directive.

--INFORMATIVE NF 614-- The DEFINE_DESTINATION_GROUP, that has the group_name {text} has the identical destination names as group_name {text}.

Condition Identifier: NFE$SF_IDENTICAL_DEST_LIST Code: NF 614

Description: The MANAGE_STORE_FORWARD_NETWORK utility has found multiple DEFINE_ DESTINATION_GROUP directives with different GROUP_NAME values but identical destination names.

User Action: Correct the directives to use only one GROUP_NAME and delete the unused directive.

--FATAL NF 615-- An invalid pointer type has been encountered in the procedure: {text}

Condition Identifier: NFE$SF_INTERNAL_ERROR_BAD_PTR Code: NF 615

Description: An internal error has occurred in the MANAGE_STORE_FORWARD_NETWORK utility.

User Action: Reinstall the MANAGE_STORE_FORWARD_NETWORK directive file.

--CATASTROPHIC NF 616-- An invalid pointer type has been encountered in the procedure: {text} please re-install the SYSTEM's Store/Forward Network File.

Condition Identifier: NFE$SF_INTERNAL_ERROR_BAD_RPTR Code: NF 616

Description: An internal error has occurred in the MANAGE_STORE_FORWARD_NETWORK utility.

User Action: Reinstall the MANAGE_STORE_FORWARD_NETWORK directive file.

--ERROR NF 617-- The name value {text} must be no longer than 31 characters in length for a {text}.

Condition Identifier: NFE$SF_NAME_TOO_LONG Code: NF 617

Description: A name value supplied for a MANAGE_STORE_FORWARD_NETWORK directive parameter is too long.

User Action: Shorten the name to a value less than or equal to 31 characters.

--ERROR NF 618-- The name value {text} must be at least 3 characters in length for a {text}.

Condition Identifier: NFE$SF_NAME_TOO_SHORT Code: NF 618

Description: A name value supplied for a MANAGE_STORE_FORWARD_NETWORK directive parameter is too short.

User Action: Lengthen the name to a value greater than or equal to 3 characters.

--FATAL NF 619-- The System's Store/Forward Network File does not exist, please install it.

Condition Identifier: NFE$SF_NO_STORE_FORWARD_NETWORK Code: NF 619

Description: The MANAGE_STORE_FORWARD_NETWORK directive file has not been installed.

User Action: Install the MANAGE_STORE_FORWARD_NETWORK directive file.

--CATASTROPHIC NF 620-- Errors encountered on reading the System's Store/Forward Network File.

Condition Identifier: NFE$SF_READ_NETWORK_FILE_ERROR Code: NF 620

Description: The installed version of the MANAGE_STORE_FORWARD_NETWORK directive file has been corrupted.

User Action: Reinstall the MANAGE_STORE_FORWARD_NETWORK directive file.

--ERROR NF 621-- The string value {text} must be no longer than 31 characters in length for a {text}.

Condition Identifier: NFE$SF_STRING_TOO_LONG Code: NF 621

Description: A string value supplied for a MANAGE_STORE_FORWARD_NETWORK directive parameter is too long.

User Action: Shorten the string to a value less than or equal to 31 characters.

--ERROR NF 622-- The string value {text} must be at least 3 characters in length for a {text}.

Condition Identifier: NFE$SF_STRING_TOO_SHORT Code: NF 622

Description: A string value supplied for a MANAGE_STORE_FORWARD_NETWORK directive parameter is too short.

User Action: Lengthen the string to a value greater than or equal to 3 characters.

--CATASTROPHIC NF 623-- Installation of System's Store/Forward Network File failed due to errors encountered on writing the Store/Forward Network File.

Condition Identifier: NFE$SF_WRITE_NETWORK_FILE_ERROR Code: NF 623

Description: The MANAGE_STORE_FORWARD_NETWORK directive file cannot be installed due to system errors.

User Action: Try to reinstall the MANAGE_STORE_FORWARD_NETWORK directive file.

Index

Index

A

B

C

D

E

F

G

I

L

M

N

O

P

Q

S

T

V

Comments (continued from other side)

Please fold on dotted line;
Seal edges with tape only.

FOLD

FOLD

FOLD

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
First-Class Mail Permit No. 8241 Minneapolis, MN

POSTAGE WILL BE PAID BY ADDRESSEE

CONTROL DATA
Technical Publications
ARH219
4201 N. Lexington Avenue
Arden Hills, MN 55126-9983

We would like your comments on this manual to help us improve it. Please take a few minutes to fill out this form.

Who are you?

- ☐ Manager
- ☐ Systems analyst or programmer
- ☐ Applications programmer
- ☐ Operator
- ☐ Other _____

How do you use this manual?

- ☐ As an overview
- ☐ To learn the product or system
- ☐ For comprehensive reference
- ☐ For quick look-up
- ☐ Other _____

What programming languages do you use? _____

How do you like this manual? Answer the questions that apply.

Yes	Somewhat	No	
☐	☐	☐	Does it tell you what you need to know about the topic?
☐	☐	☐	Is the technical information accurate?
☐	☐	☐	Is it easy to understand?
☐	☐	☐	Is the order of topics logical?
☐	☐	☐	Can you easily find what you want?
☐	☐	☐	Are there enough examples?
☐	☐	☐	Are the examples helpful? (☐ Too simple? ☐ Too complex?)
☐	☐	☐	Do the illustrations help you?
☐	☐	☐	Is the manual easy to read (print size, page layout, and so on)?
☐	☐	☐	Do you use this manual frequently?

Comments? If applicable, note page and paragraph. Use other side if needed.

Check here if you want a reply: ☐

Name _____ Company _____

Address _____ Date _____

_____ Phone _____

Please send program listing and output if applicable to your comment.


CONTROL DATA