INSURANCE

Commcommander

version 2.1.3

Revised 02/08/12

Commcommander

Version 2.1.3

February 2012

CONTENTS

Chapter 1

Commcommander

Introduction

Commcommander™ software is a communications solution consisting of a communications server, software drivers, and a graphical user interface for managing document archive members. These components allow Docucorp products to communicate with different computer operating systems and platforms.

All Commcommander components should only be installed on a computer network, which has been thoroughly *implemented and tested*. For the installation process to proceed smoothly, you should gather together the following technical personnel:

- The Network or Systems Administrator (for the PC server and the MVS host)
- The person responsible for installing Commcommander
- The person responsible for Docucorp applications

With this installation team on-site, you can successfully install and set up your Commcommander communications solution. For more information about the system requirements for each component, see System Requirements Overview.

About The Manual

This manual explains the major components of the Commcommander communications solution. The person(s) installing Commcommander should read the Requirements and Installation sections. The person(s) responsible for Docucorp applications should read the Server and Driver sections. The final section, on VLAMcommander, is intended for the person(s) editing Library members in an Electronic Document Library (EDL).

At the end, you'll find several useful appendices, a glossary of terms, and an index.

- Commcommander Servers

  The servers are general-purpose communications programs used for communicating between PC-based and mainframe transaction program applications. The Commcommander Servers are available for Windows NT, UNIX, and IBM MVS operating systems.

Tip Commcommander Servers on the NT and UNIX platforms will communicate with the Commcommander/MVS Server, and vice versa. For more information, see Commcommander Client/Server.

- Commcommander Drivers

  There are pairs of Commcommander Drivers that run on the PC platform that functions with an associated driver on MVS.

  Commcommander JES Driver
  This driver uploads Metacode and AFP print data streams to MVS host-attached printers, and is available on Windows 9x and NT, UNIX, and IBM MVS operating systems.

  Commcommander VRF Driver
  This driver transfers VRFs (Variable Replacement Files) between Docucorp mainframe and PC products, and is available on Windows 9x and NT, UNIX, and IBM MVS operating systems.

  Commcommander Imagecreate Driver
  This driver sends a data stream from MVS, across your network, to your PC. The driver ensures that the mainframe-based Printing Resources needed in the upcoming data stream-to-image conversions are available.

  Commcommander VLAM Driver
  This driver manages EDL members between operating platforms, including Windows 3.1, 9x, and NT; UNIX; and IBM MVS operating systems.

- **VLAMcommander**™

  A graphical user interface (GUI) for manipulating EDL members between operating platforms. With VLAMcommander, you can load a new or revised member from a LAN-attached PC to mainframe EDLs by simply “clicking a mouse.” VLAMcommander runs as a 16-bit application under Windows 3.1, 9x, and NT.

- Other Docucorp Products that utilize the Commcommander Drivers:

  - **Documanage™** (VLAM Driver)
  - **I.R.I.S.™** (VLAM Driver)
  - **Transall™** (Client DLLs)

Product Overview

In general, the Commcommander server provides Docucorp products with the client/server communications for processing across multiple platforms using

- Advanced Program to Program Communication (APPC), a general-purpose communications program that you can install on a Windows NT server and an MVS host server. Commcommander uses APPC for a connection between the PC server and the MVS host. APPC is an IBM System Network Architecture (SNA) application program interface (API).

- Transmission Control Protocol / Internet Protocol (TCP/IP), a general-purpose communications program that you can install on a Windows 3.1, 9x, or NT client/server and an MVS host server.
- Named Pipes is a general purpose LAN communications program that you can install on a Windows 3.1, 9x, or NT client/server.

The driver(s) are more specific in their purpose. The numbers and kinds of drivers you require depend upon your specific communication needs, because each driver is created to enable the functionality for a particular task.

VLAMcommander software, a Docucorp product, enables users to access the basic functionality of host-based **Virtual Library Access Method**™ (**VLAM**®) software, another Docucorp product. The user can load a new or revised member to a host VLAM library (e.g., EDL or Rulebase) from a LAN-attached PC.

The following diagram shows how you can use VLAMcommander to send a DOS file from the PC to the host.

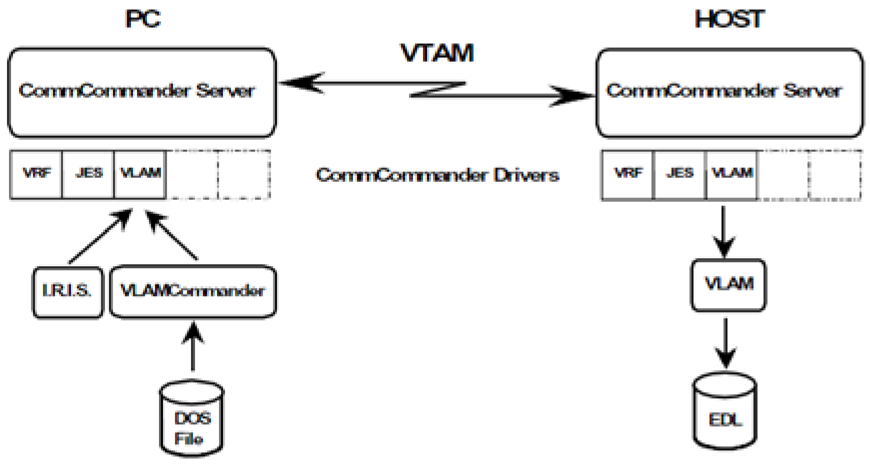


Figure 1: Commcommander family of communications components

You can easily see that new drivers and servers can be added to provide future links to different platforms and operating systems, and additional functionality.

Chapter 2

System Requirements

SYSTEM REQUIREMENTS OVERVIEW

Note If you're accessing files via Docucorp queue, you'll need to install Docucorp's Queue System.

If you're accessing files that reside in your LAN file system, you'll need the required software that provides attachment to your LAN network. The specific software depends on and would be the same as that used to connect any other machine to your network.

Before installation please decide on the following:

- The type of transport protocol you will use for client (application) to file server communications. The choices are, TCP/IP (SOCKETS) or NetBEUI (NAMED PIPES).
- The type of transport protocol you will use for files server to mainframe communications. The choices are TCP/IP (SOCKETS) or SNA. (APPC/MVS or ACF/VTAM).

The following list represents the minimum system requirements per selected platform.

FILE SERVER

Windows NT

- Windows NT Server 4.0 or higher
- 2 MB of local disk space
- For Workstations connecting to Windows NT Server

TCP/IP (SOCKETS) enabled on the LAN attached server machine

-or-

NetBEUI (NAMED PIPES) enabled on the LAN attached server machine

- For Windows NT connecting to MVS Host

TCP/IP (SOCKETS) enabled on the LAN attached server machine

-or-

APPC LU 6.2 Microsoft SNA Server 2.11 or higher

RISC System 6000/AIX

- AIX 5.1 or higher
- 2 MB of local disk space
- For Workstations connecting to RISC System 6000 Server

  TCP/IP (SOCKETS) enabled on the LAN attached server machine
- RISC System 6000 Server connecting to MVS Host

  TCP/IP (SOCKETS) enabled on the LAN attached server machine

  -or-

  APPC LU 6.2 IBM Communications Server for AIX 5.1 or higher

Linux (Intel)

- Red Hat Linux 2.1
- 2 MB of local disk space
- For Workstations connecting to Intel-based Linux server

  TCP/IP (SOCKETS) enabled on the LAN attached server machine
- Intel-based Linux server connecting to MVS Host

  TCP/IP (SOCKETS) enabled on the LAN attached server machine

WORKSTATION

Windows/NT/9x/3.1

TCP/IP — (SOCKETS)

- Windows NT Workstation 4.0 or higher
- Windows 9x Workstation
- Windows 3.1 Workstation
- TCP/IP for Sockets enabled on the LAN attached workstation for access to the server machine
- 2 MB of local disk space

NetBEUI — (NAMED PIPES)

- Windows 9x Workstation
- Windows 3.1 Workstation with Microsoft Network Client 3.0 or higher for access to the server machine

- 2 MB of local disk space

MAINFRAME

MVS/ESA or OS/390

TCP/IP — (SOCKETS)

- MVS/ESA 4.2.0 or higher

  -or-

  OS/390 1.3 or higher
- IBM TCP/IP for MVS 3.1 or higher

SNA — (APPC/MVS)

- MVS/ESA 4.2.0 or higher

  -or-

  OS/390 1.3 or higher

  APPC/MVS for APPC LU 6.2

SNA — (ACF/VTAM)

- MVS/ESA 4.0 or 4.1

  -or-

  OS/390 1.3 or higher
- ACF/VTAM

WARNING! VLAMcommander for Windows requires VLAM 3.1 (or higher) installed on the mainframe. VLAMcommander runs as a 16-bit application under Windows 9x and NT.

The Imagecreate Driver requires SAS/C Runtime Libraries (6.5).

The software versions specified above are the lowest levels known to be supported. More current (or higher levels) may be supported as long as the software vendor has provided upward compatibility.

NETWORK COMMUNICATION

In general, the Commcommander server provides Docucorp products with the client/server communications for processing across multiple platforms using

- Transmission Control Protocol/Internet Protocol (TCP/IP), a general-purpose communications program that you can install on a Windows 3.1, NT, or UNIX client/server and an MVS host server.

  With this method, a Commcommander Server application is started on MVS. This Commcommander Server listens for communications requests from a Commcommander Server that is installed on a server machine. The Commcommander Server, running on MVS, will process all request receive from the Commcommander Server running on the server machine.

  The example stated above describes the process for conversations originating on the PC, for conversation originating on the MVS mainframe this process is reversed. When the Commcommander Server running on the server machine receives a request it invokes the appropriate Commcommander Driver program to complete the specific task. The Commcommander Driver is considered a Transaction Program (TP).

  This method can also be used for communications between a client workstation and a server machine. With this method, a Commcommander Server application is started on a server machine. This Commcommander Server listens for communications requests from a Commcommander Driver that is installed on a workstation. The Commcommander Server, running on the server machine, will process all request receive from the Commcommander Driver running on the workstation.

- Advanced Program-to-Program Communication (APPC), a general-purpose communications program that you can install on a server machine and an MVS host server. Commcommander uses APPC for a connection between the PC server and the MVS host. APPC is an applications program interface (API), which allows access to IBM's System Network Architecture (SNA) facility.

  With this method, the active MVS APPC (SNA) facility waits for communication request from a Commcommander Server running on a server machine. When the MVS APPC facility receives a request it starts a Job known as a Transaction Program (TP). This TP (which is essentially Commcommander Server) will process all request received from the Commcommander Server running on the server machine.

  The example stated above describes the process for conversations originating on the PC, for conversation originating on the MVS mainframe this process is reversed. When the APPC (SNA) facility running on the server machine receives a request it invokes the appropriate Commcommander Driver program to complete the specific task. The Commcommander Driver is considered a Transaction Program (TP).

- Advanced Communication Facility VTAM (ACF/VTAM), a general-purpose communications program that you can install on an MVS host server. Commcommander uses ACF/VTAM for a connection between the PC server and the MVS host.

  With this method, a Commcommander Server application is started on MVS. This Commcommander Server listens for communications requests from a Commcommander Server that is installed on a server machine. The Commcommander Server, running on MVS, will process all request receive from the Commcommander Server running on the server machine.

  The example stated above describes the process for conversations originating on the PC, for conversation originating on the MVS mainframe this process is reversed. ACF/VTAM is not available for PC based systems, for this, an APPC (SNA) facility is used. When the APPC (SNA) facility running on the server machine receives a request it invokes the appropriate Commcommander Driver program to complete the specific task. The Commcommander Driver is considered a Transaction Program (TP).

- Named Pipes is a general purpose LAN communications program that you can install on a Windows 3.1 or NT client/server.

  This method is used for communications between a client workstation and a server machine. With this method, a Commcommander Server application is started on a server machine. This Commcommander Server listens for communications requests from a Commcommander Driver that is installed on a workstation. The Commcommander Server, running on the server machine, will process all request receive from the Commcommander Driver running on the workstation.

Commcommander supports the communication protocols listed above in the following ways:

- For conversations originating on the PC, the available choices for communications between Commcommander Drivers and Commcommander Server/NT/UNIX are:
  - TCP/IP
  - NamedPipes—NT only
- For conversations originating on the PC, the available choices for communications between Commcommander Server/NT/UNIX and Commcommander Server/MVS are:
  - TCP/IP
  - APPC (SNA)—IBM AIX only
- For conversations originating on MVS, the available choices for communications between Commcommander Server/MVS and Commcommander Server/NT/UNIX are:
  - TCP/IP
  - APPC (SNA)—IBM AIX only
  - ACF/VTAM (SNA)—IBM AIX only

	Communications between	
Transmission Protocol	Workstation and Server machine	Server machine and MVS Host
TCP/IP	X	X
APPC		X
ACF/VTAM		X
NamedPipes	X	

Supported Commcommander Protocols

Note TCP/IP MVS communicates with TCP/IP on your Server machine via Commcommander Server Inbound.

APPC/MVS communicates with APPC SNA server software on your server machine (e.g., Microsoft's SNA Server or IBM's Communication Server (SNA)).

ACF/VTAM MVS communicates with APPC SNA server software on your server machine (e.g., Microsoft's SNA Server or IBM's Communication Server (SNA)).

The following are tasks normally done by your network administrator. These are not all the network definitions required for the complete network implementation.

Network Configuration Requirements

File Server To Mainframe

Your host machine and server machine must be network *configured and tested*, and be a fully functional part of your network. These machines must be able to communicate with one another using TCP Internet Protocol or SNA APPC (VTAM LU 6.2).

Application PC to File Server

Your application PC and file server must be network *configured and tested*, and be a fully functional part of your Local Area Network (LAN). These machines must be able to communicate with one another using either the TCP Interface Protocol (Sockets) or NetBEUI (Named Pipes).

If you choose to use "Sockets" to communicate between your client workstation and server, you must have a functional TCP/IP installed, tested, and operational. If you're connecting a Windows 3.1 workstation, you must have a functional TCP/IP Winsock.dll installed.

If you choose to use "Named Pipes" to communicate between your Windows 3.1 workstation and server, you must use Microsoft's Network Client to connect to the file server machine. Windows 9x requires NetBEUI enabled.

For specific network configuration requirements, refer to the following publications:

- (IBM) "Planning: APPC Management" (GC28-1110)
- (IBM) *ACF/VTAM* — "VTAM Resource Definition Reference" (SC31-6412)
- (IBM) "TCP/IP Programming Interface Reference" (SC31-7187-00)
- (IBM) "TCP/IP Customization and Administration Guide" (SC31-7134-01)
- (IBM) InfoExplorer for AIX 5.1 (on installation CD)
- "Microsoft SNA Server: Administration Guide" (SY57265)
- "Microsoft Windows NT Systems Guide" (on installation CD)

Network Communication Profiles

The following are tasks normally done by your network administrator. These are not all the network definitions required for the complete network implementation, rather, this focuses on the profiles that pertain to Docucorp products.

TCP/IP

Your host machine and server must be network configured, tested and a fully functional part of your network. The machines must be able to communicate with one another using TCP/IP for sockets. Commcommander's implementation of TCP/IP uses “stream sockets”. In addition to TCP/IP being installed and configured, there are two addresses used by TCP/IP applications. These are the “host address” and the “port address”.

The “host address” is either defined in a “hosts” file on the client machine (in a client/server relationship), or some installations manage these with TCP/IP utilities, either “Domain Named System” (DNS) or DHCP. Contact your network administrator to determine which method is used at your installation.

The “port address” is defined in a “services” file on both the client and host machine. Whenever a TCP/IP application makes a connection using sockets, the “port address” on both the client and host machines must match.

Commcommander contains drivers that connect both directions, from your PC LAN platform to MVS, and from MVS to your PC LAN platform. If you're implementing a Commcommander driver that originates a conversation on your PC LAN platform, then this is the client side of the conversation, and the following utilities must be invoked from your PC LAN platform. You must invoke these utilities from MVS if the Commcommander driver originates the conversation on MVS. Refer to the specific Commcommander driver to determine where the conversation originates.

The following TCP/IP utilities can be used to demonstrate that your TCP/IP is installed and functioning properly. These utilities should be invoked from the client side of your client/server relationship.

Test Utilities — FTP or Telnet

Both of these utilities use a “host address” and a “port address” to make the connection. They also require both a client and a host implementation. Therefore, if you can successfully “Telnet” or “FTP” from your client to your host, then this demonstrates that your TCP/IP is installed and functioning properly.

Test Utilities — PING

PING is a useful TCP/IP utility that verifies the networking hardware and addresses are installed and defined. PING only uses the “host address” to make its connection, but does not verify the use of “port addresses”. Therefore, PING is a useful utility to begin testing any network failures, especially when the above Telnet or FTP fails.

SNA

Your host machine and server must be network configured, tested and a fully functional part of your SNA network. These machines must be able to communicate with one another using VTAM's LU 6.2 protocol.

Test Utilities — APING

IBM and Microsoft both provide utilities to test the PC server-to-mainframe connection and data transfer capability. One utility is called **APING** and can normally be obtained from the communication software (for IBM, the directory is called UTILITIES; for Microsoft, the directory is called SNA\SAMPLES\BIN). Also, there is a forum on CompuServe where you can retrieve this and other utilities and additional APPC documentation. Sign-on to CompuServe and type **GO APPC** for IBM or **GO MSDN** for Microsoft.

Network Definition and Configuration Assistance

Microsoft

Microsoft SNA Server Help provides a good overview and detailed explanation of the requirements to define and configure both the NT SNA Server and the SNA VTAM definitions.

IBM

IBM provides a diagnostic wizard on their Web site. IBM's APPC/APPN Configuration Wizard is an interactive configuration tool that assists an individual in generating the VTAM APPC communication definitions for the various platforms. This tool is available on the Internet.

Note As with most all Internet Web pages, these addresses/links may change. Contact the vendor's Web Master for the most current links.

To use the Wizard, perform the following steps:

- **IBM Home Page** (www.ibm.com).
  1. Select Products from the menu at the top of the page.
  2. Select Networking, which will take you to www.networking.ibm.com.
- **IBM Networking Home Page** (www.networking.ibm.com).
  1. Select the "Index" section of their web site.
  2. Select the "A's".
  3. Select "APPN Implementer's Workshop".
  4. Select "Other web sites with APPN/DLSw information."
  5. Select "APPC/APPN Configuration Wizard."

Implementation Steps

In order to correctly operate Commcommander, its drivers, and VLAMcommander, you must implement the components in the proper order. Each new component builds on the capabilities of the previous one, so it's imperative that you install and use them correctly.

This check list should help you to ensure you're using all of the correct components:

- (APPC) Install and have operational Windows NT or IBM SNA Services communication software.
- (APPC) APPC/MVS and VTAM definitions must be installed and operational.
- (ACF/VTAM) VTAM definitions must be installed and operational.
- (TCP/IP) Install and have operational TCP/IP on your client and server machines.
- (TCP/IP) Install and have operational TCP/IP for MVS on your mainframe host machine.
- Install the Commcommander Server, a Docucorp product.

  Host Computer

  - Commcommander Server/MVS

  PC Server

  - Commcommander Server/NT

    -or-

    Commcommander Server/UNIX

- Install the Commcommander Driver(s) you'll be using. The following list represents a sample installation.

  Host Computer

  - Commcommander VLAM Driver
    Commcommander JES Driver
    Commcommander VRF Driver

  PC Server

  - Commcommander JES Driver (May be installed on a client workstation)
    Commcommander VRF Driver (May be installed on a client workstation)

  PC Client

  - Commcommander VLAM Driver
    VLAMcommander for Windows

Chapter 3

Installation

INSTALLING COMMCOMMANDER COMPONENTS

Note Prior to installing Commcommander components, you must meet the network's system requirements discussed in System Requirements Overview.

Because Commcommander components are supported on several platforms, this guide provides separate procedures for each platform.

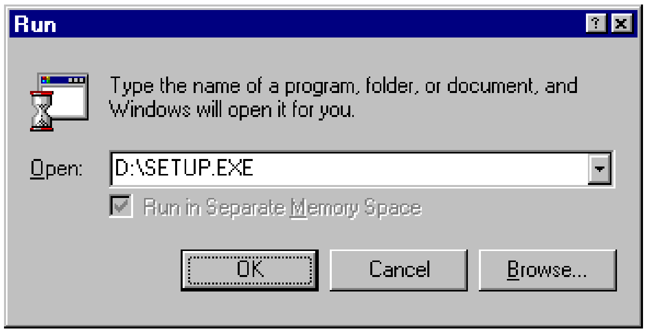
To Install	See
Commcommander components on Windows NT	Installing Components in Windows NT
Commcommander components on UNIX platforms	Installing Components for UNIX
Commcommander components on IBM MVS	Installing MVS Server Components

INSTALLING COMPONENTS IN WINDOWS NT

When you install Commcommander Client/Server components under Windows NT, the procedure follows the standard Windows conventions.

To Install Components in Windows NT

1. Insert the Commcommander Components Installation CD in the CD-ROM drive of the workstation on which you intend to install Commcommander.
2. Choose **Start/Run** from the Windows Desktop File menu and Windows displays the Run dialog box.



Run

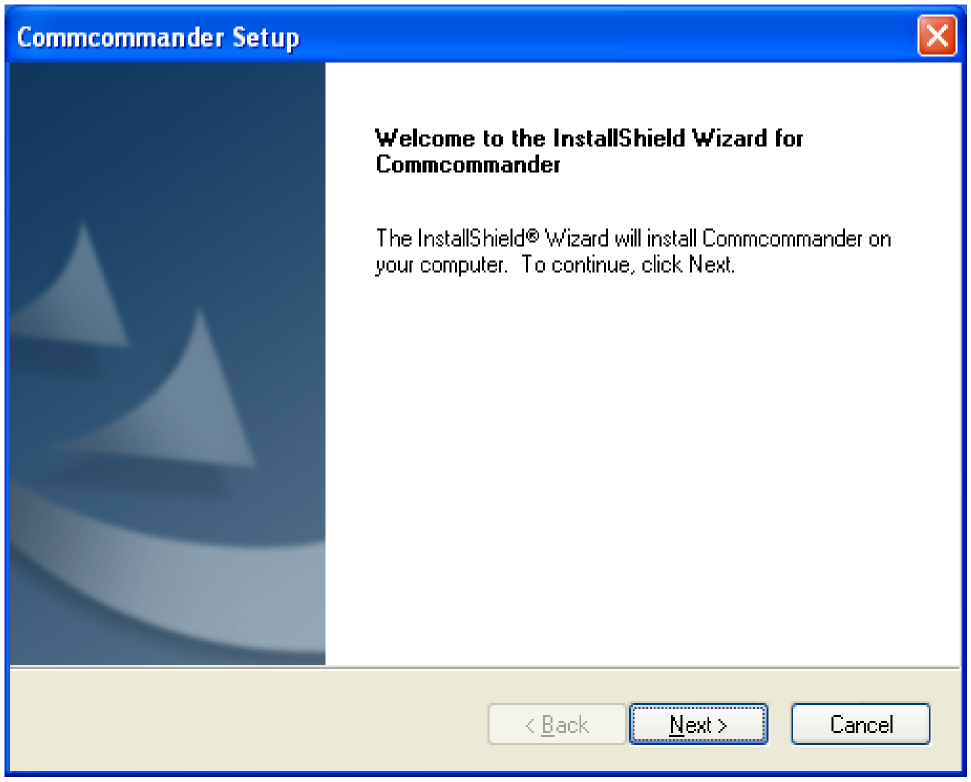
Type the name of a program, folder, or document, and Windows will open it for you.

Open: D:\SETUP.EXE

☑ Run in Separate Memory Space

3. Enter **D:\SETUP** in the Command Line text box and choose **OK**. If the installation CD is in a drive with a different drive letter, you should substitute the appropriate letter.

   The installation routine displays a dialog box indicating the InstallShield Wizard's initialization progress, followed by the Commcommander Setup dialog box.

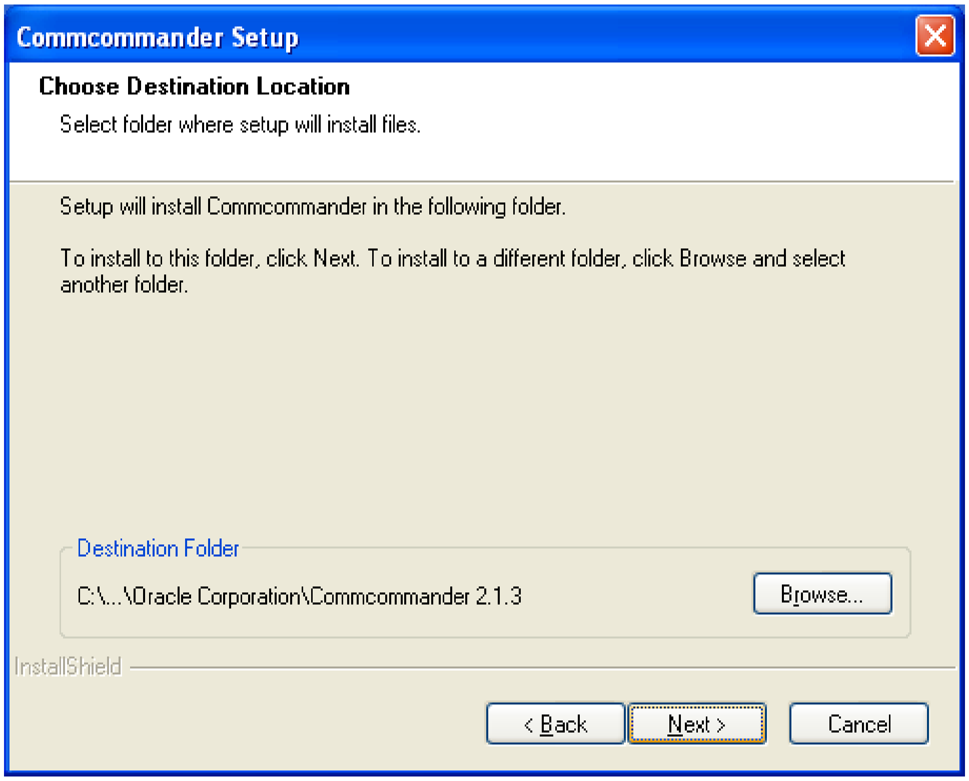


Commcommander Setup

Welcome to the InstallShield Wizard for Commcommander

The InstallShield® Wizard will install Commcommander on your computer. To continue, click Next.

4. Click on **Next** to proceed to the Choose Destination Location Folder dialog, or Back to return to the previous dialog box.



Commcommander Setup

Choose Destination Location

Select folder where setup will install files.

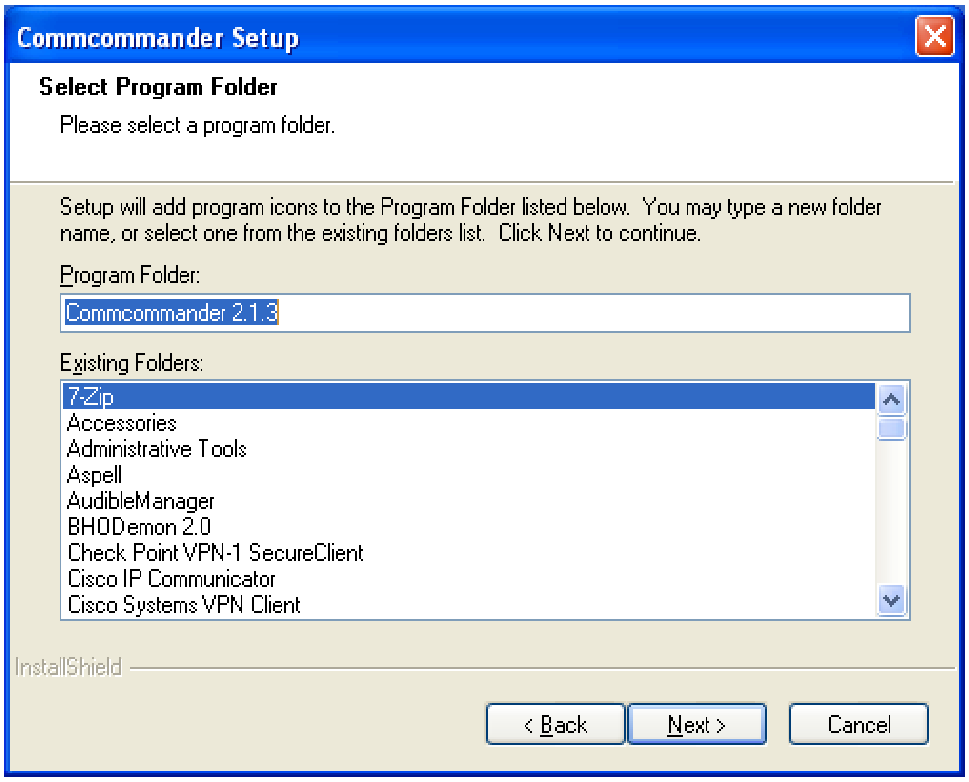
Setup will install Commcommander in the following folder.

To install to this folder, click Next. To install to a different folder, click Browse and select another folder.

Destination Folder

C:\...\Oracle Corporation\Commcommander 2.1.3

5. Accept the Default Destination Folder or select **Next** to accept the Destination Folder and proceed to the Select Program Folder dialog box, or click the **Browse** button to choose another location.

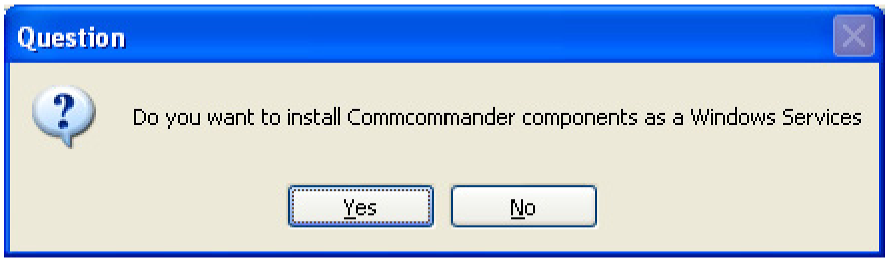


Setup will add program icons to the Program Folder listed below. You may type a new folder name, or select one from the existing folders list. Click Next to continue.

Program Folder:
Commcommander 2.1.3

Existing Folders:
7-Zip
Accessories
Administrative Tools
Aspell
AudibleManager
BHODemon 2.0
Check Point VPN-1 SecureClient
Cisco IP Communicator
Cisco Systems VPN Client

6. Accept the default Program Folder or type in a new name and then select Next to proceed or **Back** to return to the previous dialog box. You are asked to proceed with the installation:

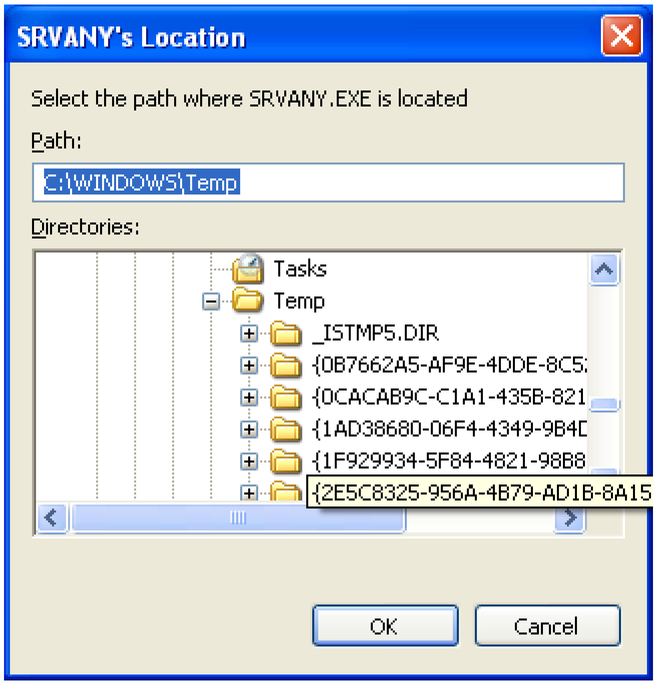


Do you want to install Commcommander components as a Windows Services

7. Select **Yes** here, if you desired to run the application as a Windows Service, otherwise select **No** to proceed to the Start Copying Files dialog.

Note

- If you chose to run as a Windows Service, you must specify the name of the directory where the SRVANY.EXE resides. Commcommander Services are defined to the OS via the SRVANY program. You must install this module on your workstation prior to setting up Commcommander Services. If you don't already have a copy of the SRVANY module, try downloading it from Microsoft (online).



SRVANY's Location

Select the path where SRVANY.EXE is located

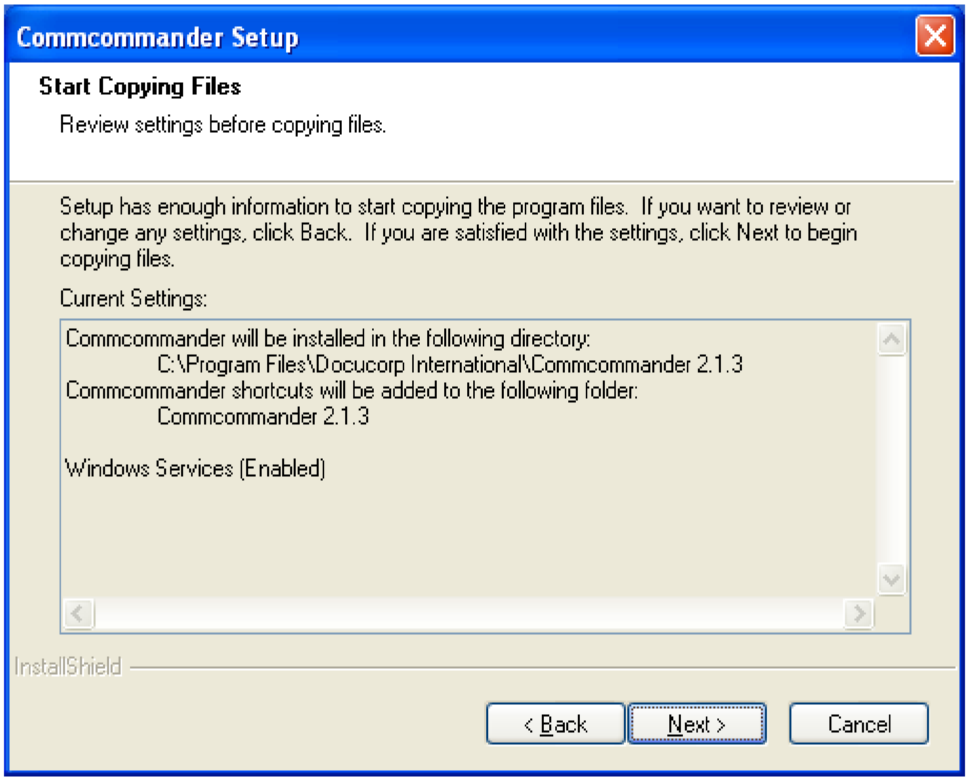
Path:

C:\WINDOWS\Temp

Directories:

- Enter the path of the SRVANY executable module and select **OK** to continue.
  If you do not wish to install the application as a Windows Service, select **Cancel**.

The Start Copying Files dialog box allows you to review your installation specifications.



Start Copying Files

Review settings before copying files.

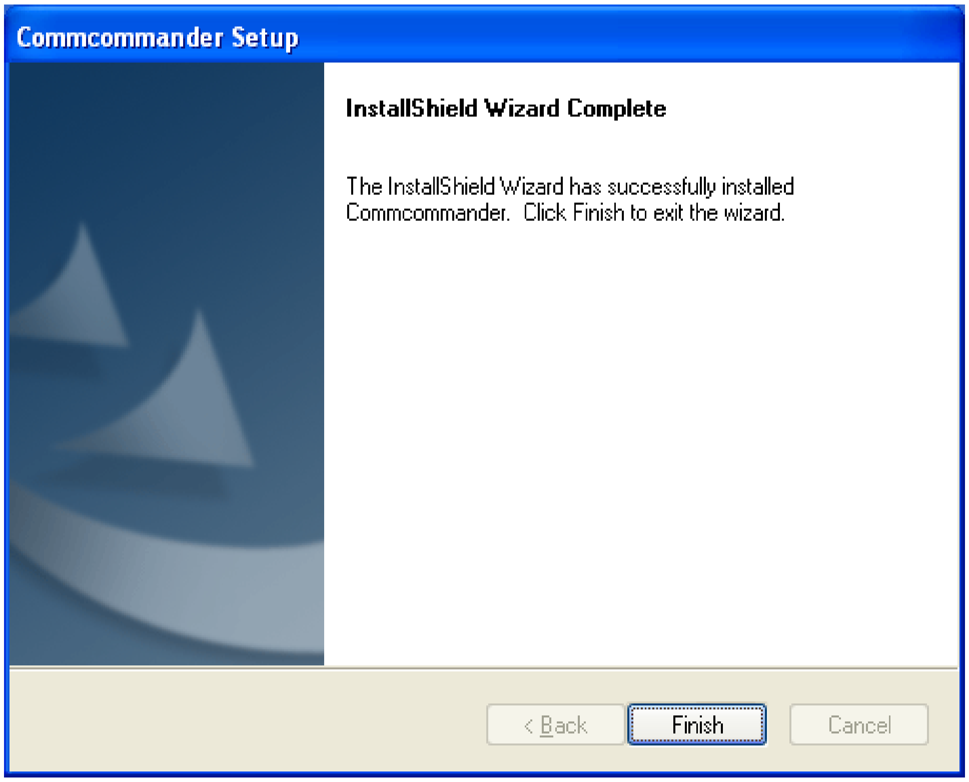
Setup has enough information to start copying the program files. If you want to review or change any settings, click Back. If you are satisfied with the settings, click Next to begin copying files.

Current Settings:

```
Commcommander will be installed in the following directory:
            C:\Program Files\Docucorp International\Commcommander 2.1.3
Commcommander shortcuts will be added to the following folder:
            Commcommander 2.1.3

Windows Services (Enabled)
```

8. Review the installation specification and select Next select **Next** to proceed with installation or **Back** to return to the Choose Destination Location Folder dialog. Choosing **Cancel** here terminates the install.



Commcommander Setup

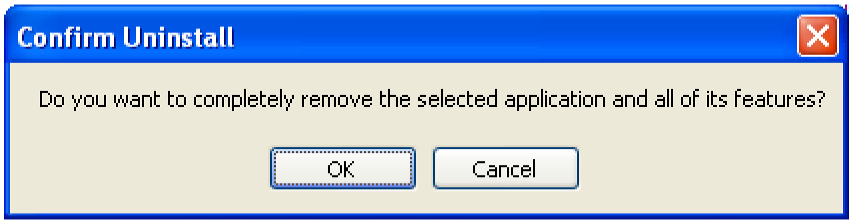
InstallShield Wizard Complete

The InstallShield Wizard has successfully installed Commcommander. Click Finish to exit the wizard.

9. At the completion of the installation, the Wizard Complete dialog box is displayed. Select **Finish** to end the installation procedure.

To Uninstall Components from Windows NT

1. Follow steps 1-2 of the Install Components from Windows NT to launch the Setup installer. This displays the Confirm Uninstall dialog box:

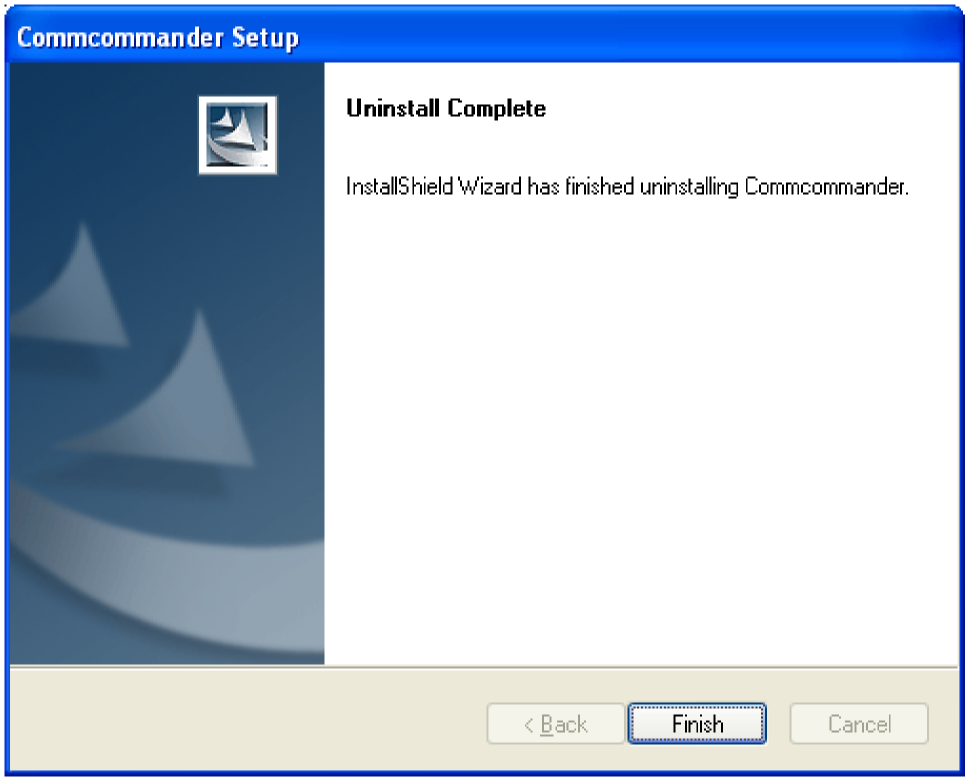


Confirm Uninstall

Do you want to completely remove the selected application and all of its features?

2. Select **OK** to proceed with uninstalling the application, or **Cancel** to terminate the installation procedure.

3. At the completion of the application removal, the Uninstall Complete dialog box is displayed.


Commcommander Setup

Uninstall Complete

InstallShield Wizard has finished uninstalling Commcommander.

4. Select **Finish** to end the uninstall procedure.

Installing Components for UNIX

When you install Commcommander Servers or Drivers for UNIX, the files are in the standard **.tar* file formats.

To Install Components in UNIX

1. Create an installation directory for this application (e.g., **/u/comcmdr**).
2. Copy the installation CD into the installation directory.
3. From a command prompt, change directories to the installation directory.
4. Execute the UNIX “tar” utility from the command line to install the various Commcommander for UNIX components (e.g., “**tar -xvf name.tar**,” where *name.tar* is the name of the Commcommander distribution *tar* file).

   The *tar* file name varies, depending on the version and platform to which you’re installing. The following list represents the current *tar* file names for various platforms; newer release names will be distributed as they’re developed:

- comcmdr.tar (Commcommander Server)—IBM AIX 5, dated 06/11/03
- client.tar (Commcommander Client)—IBM AIX 5, dated 06/11/03

- isijes.tar (JES Driver)—IBM AIX 5, dated 06/11/03
- vlam.tar (VLAM Driver)—IBM AIX 5, dated 06/11/03
- isivrf.tar (VRF Driver)—IBM AIX 5, dated 06/11/03
- comcmdr020101AIX51.tar—IBM AIX 5.1, dated 08/25/04
- client020101AIX51.tar—IBM AIX 5.1, dated 08/25/04
- isijes020101AIX51.tar—IBM AIX 5.1, dated 08/25/04
- vlam020101AIX51.tar—IBM AIX 5.1, dated 08/25/04
- isivrf020101AIX51.tar—IBM AIX 5.1, dated 08/25/04
- comcmdr020101LINUX.tar—Linux 2.1, dated 08/18/04
- client020101LINUX.tar—Linux 2.1, dated 08/18/04
- isijes020101LINUX.tar—Linux 2.1, dated 08/18/04
- vlam020101LINUX.tar—Linux 2.1, dated 08/18/04
- isivrf020101LINUX.tar—Linux 2.1, dated 09/08/04

Installing MVS Server Components

The INSTALLATION Job

The INSTALLATION job copies the remaining files from the distribution tape to the host, allocating all required data sets.

```
//* ** PLACE YOUR JOB CARD HERE
//*
//*********************************************************************
//*  INSTALL  -
//*
//*  COPIES THE COMMCOMMANDER BINARY AND DATA FILES FROM A
//*  TRANSMIT/XMIT FILE TO A Z/OS SYSTEM.
//*
//*  1. RECV1   - RECEIVES A FILE THAT HAS BEEN UPLOADED FROM
//*               WINDOWS (THE FILE IS IN TRANSMIT/XMIT FORMAT)
//*               INTO A PARTITIONED DATASET.
//*
//*  2. RECV2   - RECEIVES A PARTITIONED DATASET FROM RECV1
//*               TO LOAD COMMCOMMANDER RESOURCES.
//*
//*  REPLACE 'ISI.DFCCOM.V02R01.XMITSEQ'
//*     IN RECV1 STEP WITH YOUR UPLOADED DATASET NAME
//*     - THIS DATASET CAN BE DELETED WHEN THE JOB FINISHES -
//*
//*  REPLACE ALL '&HLQ' WITH YOUR HIGH-LEVEL QUALIFIER NAME
//*
//*
//*********************************************************************
//*
//RECV1    EXEC PGM=IKJEFT01,REGION=0M,DYNAMNBR=175
//SYSUADS  DD  DISP=SHR,DSN=SYS1.UADS
//SYSLBC   DD  DISP=SHR,DSN=SYS1.BRODCAST
//*
//XMITFILE DD  DISP=SHR,DSN=ISI.O.DFCCOM.V02R01.XMITSEQ
//*
//SYSPRINT DD  SYSOUT=*
//SYSOUT   DD  SYSOUT=*
//SYSTSOUT DD  SYSOUT=*
//SYSTSPRT DD  SYSOUT=*
//SYSTSIN  DD  *
 RECEIVE INDDNAME(XMITFILE)
         DSN('&HLQ.DFCCOM.V02R01.XMITPDS')
/*
//*
//   SET PDS='&HLQ.DFCCOM.V02R01.XMITPDS'
//*
//RECV2    EXEC PGM=IKJEFT01,REGION=0M,DYNAMNBR=175
//SYSUADS  DD  DISP=SHR,DSN=SYS1.UADS
//SYSLBC   DD  DISP=SHR,DSN=SYS1.BRODCAST
//*
//XMITPD1  DD  DISP=SHR,DSN=&HLQ.DFCCOM.V02R01.XMITPDS(JCL)
//XMITPD2  DD  DISP=SHR,DSN=&HLQ.DFCCOM.V02R01.XMITPDS(LOAD)
//XMITPD3  DD  DISP=SHR,DSN=&HLQ.DFCCOM.V02R01.XMITPDS(AFP)
//XMITPD4  DD  DISP=SHR,DSN=&HLQ.DFCCOM.V02R01.XMITPDS(FONT3820)
//XMITPD5  DD  DISP=SHR,DSN=&HLQ.DFCCOM.V02R01.XMITPDS(FORMDEF)
//XMITPD6  DD  DISP=SHR,DSN=&HLQ.DFCCOM.V02R01.XMITPDS(OVERLAY)
//XMITPD7  DD  DISP=SHR,DSN=&HLQ.DFCCOM.V02R01.XMITPDS(PSEG)
//XMITPD8  DD  DISP=SHR,DSN=&HLQ.DFCCOM.V02R01.XMITPDS(META)
//XMITPD9  DD  DISP=SHR,DSN=&HLQ.DFCCOM.V02R01.XMITPDS(MFONTS)
//XMITPD10 DD  DISP=SHR,DSN=&HLQ.DFCCOM.V02R01.XMITPDS(VFONTS)
//XMITPD11 DD  DISP=SHR,DSN=&HLQ.DFCCOM.V02R01.XMITPDS(SASC)
//*
```

Continued on next page

TAPEREAD JCL (Figure 1 of 2)

Continued from previous page

```
//SYSPRINT DD  SYSOUT=*
//SYSOUT   DD  SYSOUT=*
//SYSTSOUT DD  SYSOUT=*
//SYSTSPRT DD  SYSOUT=*
//*********************************************************************
//*
//*  NOTE THAT SYMBOLIC SUBSTITUTION IS NOT ALLOWED IN
//*     DSN('&HLQ..   . . . ')
//*     YOU MUST REPLACE '&HLQ.' WITH ACTUAL QUALIFIER NAMES
//*
//*  YOU CAN DELETE ANY OF THE FOLLOWING RECEIVE INDDNAME/DSN PAIRS
//*     THAT YOU DON'T NEED TO LOAD.  EACH DELETION SHOULD INCLUDE
//*     RECEIVE INDDNAME AND DSN TOGETHER AS A PAIR
//*
//*********************************************************************
//SYSTSIN  DD  *
 RECEIVE INDDNAME(XMITPD1)
   DSN('&HLQ.DFCCOM.V02R01.JCL') SPACE(2,1),TRACKS
 RECEIVE INDDNAME(XMITPD2)
   DSN('&HLQ.DFCCOM.V02R01.LOADLIB') SPACE(18,1),TRACKS
 RECEIVE INDDNAME(XMITPD3)
   DSN('&HLQ.DFCCOM.V02R01.AFP') SPACE(1,1),TRACKS
 RECEIVE INDDNAME(XMITPD4)
   DSN('&HLQ.DFCCOM.V02R01.AFP.FONT3820') SPACE(12,1),TRACKS
 RECEIVE INDDNAME(XMITPD5)
   DSN('&HLQ.DFCCOM.V02R01.AFP.FORMDEF') SPACE(1,1),TRACKS
 RECEIVE INDDNAME(XMITPD6)
   DSN('&HLQ.DFCCOM.V02R01.AFP.OVERLAY') SPACE(29,1),TRACKS
 RECEIVE INDDNAME(XMITPD7)
   DSN('&HLQ.DFCCOM.V02R01.AFP.PSEG') SPACE(1,1),TRACKS
 RECEIVE INDDNAME(XMITPD8)
   DSN('&HLQ.DFCCOM.V02R01.META') SPACE(1,1),TRACKS
 RECEIVE INDDNAME(XMITPD9)
   DSN('&HLQ.DFCCOM.V02R01.META.MFONTS') SPACE(24,1),TRACKS
 RECEIVE INDDNAME(XMITPD10)
   DSN('&HLQ.DFCCOM.V02R01.META.VFONTS') SPACE(17,1),TRACKS
 RECEIVE INDDNAME(XMITPD11)
   DSN('&HLQ.SASC.C650.RUNTIME.LIBRARY') SPACE(84,1),TRACKS
/*
//*
//DELETE   EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
  DELETE &HLQ.DFCCOM.V02R01.XMITPDS
/*
//
```

TAPEREAD JCL (Figure 2 of 2)

1. Modify the JCL to meet your site's requirements. The **bold** portions of the above JCL represent the site-specific parts.
2. Submit the job.
3. Check the job log for errors when the job is complete. If necessary, correct the errors and resubmit the job.

Chapter 4

Commcommander Server

OVERVIEW

The Commcommander Servers provide Docucorp products with client/server communications for processing across multiple platforms using Transmission Control Protocol/Internet Protocol (TCP/IP) or Advanced Program to Program Communications (APPC).

COMMCOMMANDER CLIENT/SERVER

There are two major components in Commcommander, based on the origination point of the conversation. The first component is when the primary movement of data is uploaded from the PC to the mainframe, the other is when the primary movement of data is downloaded from the mainframe to the PC. There are two partner nodes associated with each of these two major components. One is the outbound message handler and the other is the inbound message handler. The type of message handler that is used depends on where the conversation originates. After the conversation has been established, data is sent back and forth between the two nodes.

The inbound and outbound message handlers are implemented in the Commcommander Servers depending on the protocol and operating system you use. If the data is being moved from the PC to the mainframe, the conversation originates on the PC; therefore, the Commcommander node on the PC is the outbound message handler (**COMCMDRO**™ software is a Docucorp product). The corresponding node on the mainframe is the inbound message handler (**DFCCOMI**™ software is a Docucorp product).

If the data is being moved from the mainframe to the PC, on the other hand, the conversation originates on the mainframe; therefore, the Commcommander node on the mainframe is **DFCCOMO**, while the node on the PC is **COMCMDRI**.

Examples of Docucorp products that originate the conversation on the PC are as follows:

- VLAMcommander for Windows — using Commcommander VLAM Driver
- **I.R.I.S.**™ software — using Commcommander VLAM Driver
- **Transall**™ software — using Commcommander VLAM Driver
- Commcommander JES Driver software

- Commcommander VRF Driver software

Examples of products that originate the conversation on the mainframe are as follows:

- **Imagecreate**™ software — using Commcommander Client Driver
- Commcommander VRF Driver software

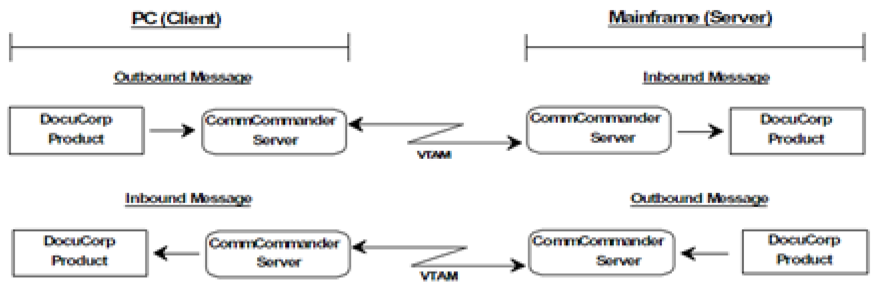


Figure 2: Illustration of the conversation origination

DIFFERENT VERSIONS OF COMMCOMMANDER

If you're operating in an MVS/ESA or OS/390 environment with APPC, you should use the APPC/MVS or TCP/IP version of Commcommander. Otherwise, you should use the ACF/VTAM version of Commcommander. Both versions provide the same functionality, but the installation process and operational details are slightly different.

Operating under ACF/VTAM or TCP/IP is described in . If you're operating in an ACF/VTAM or TCP/IP environment, be sure to read Starting Commcommander in a TCP/IP or ACF/VTAM Environment.

For a description of Commcommander operations under APPC/MVS, see .

COMMCOMMANDER SERVER OUTBOUND

Because Commcommander Server is supported on several platforms with optional protocols, this guide provides separate procedures for each platform protocol combination.

To Set Up	See
Commcommander Server using TCPIP for NT/ UNIX	Setting Up (Outbound) Commcommander Server using TCPIP for NT/ UNIX
Commcommander using SNA for NT	Setting Up (Outbound) Commcommander Server using SNA for NT
Commcommander Server using SNA for AIX	Setting Up (Outbound) Commcommander Server SNA for AIX
Commcommander Server using TCPIP/SNA for MVS	Setting Up (Outbound) Commcommander Server using TCPIP/SNA for MVS

SETTING UP (OUTBOUND) COMMCOMMANDER SERVER USING TCPIP FOR NT/UNIX

Before running Commcommander Server Outbound, certain TCP/IP configuration settings have to be defined and activated. The following describes the types of TCP/ IP configuration settings that are necessary to run Commcommander Server Outbound.

Note After the MVS TCP/IP configuration settings are in place, obtain from your MVS Network or Systems Administrator, the IP address of the of the MVS machine where Commcommander Server Inbound (MVS) will be running. Also, you will need to know the Service Port number assigned to the Commcommander Server Inbound (MVS) application.

Defining the Host IP Address, Service Port Number, and Initialization Entries

Values for host IP address and service port number are used in program-to-program communication within the TCP/IP network. A unique service port number is necessary for the establishment of each TCP/IP session. Commcommander Server Outbound requires two sessions: one for the LAN inbound and one for the outbound connection to the MVS Mainframe. The outbound connection to the MVS mainframe requires a host IP address for the establishment of a session.

To Define the Host IP Address

Use the following steps when you define the IP address:

1. Edit the **HOST** file.

   This file is normally located in the **c:\windows\system32\drivers\etc (NT) or the root /etc (UNIX)** directory. The actual location of the Host file may be in a different directory depending on your Windows/NT system directory name and/ or the specific vendor’s TCP/IP you have installed.

2. Enter the remote host IP address and associate a name with this entry (e.g., **111.212.121.222mvsHost**).
3. Save the **HOST** file.

To Define the Service Port Number

Use the following steps when you define the port number:

1. Edit the **SERVICES** file.

   This file is normally located in the **c:\windows\system32\drivers\etc (NT) or the root /etc (UNIX)** directory. The actual location of the Services file may be in a different directory depending on your Windows/NT system directory name and/or the specific vendor's TCP/IP you have installed.

2. Enter a unique port number and associated name, for your inbound socket connection. The port address you assign must be unique within this services file and it must match the same port address number that you assign on your client's machine.

   Example:

```
GATE0     2048/tcp     #Commcommander client input port number
```

3. Enter a unique port number and associated name, for your outbound socket connection. The port address you assign must be unique within this services file and it must match the same port address number that you assign on your MVS host machine.

   Example:

```
DFCCOMI     4096/tcp     #Commcommander server output MVS port number
```

4. Save the **SERVICES** file.

WARNING! Commcommander Service Port numbers should be greater than 1024, and not a duplicate of any other number in the SERVICES file.

Note If you plan to run multiple Commcommander Servers concurrently on the same machine, add two Service Port entries per Server.

Defining Initialization Entries in the COMCMDR.INI file

Commcommander Server Outbound utilizes an initialization control file, where information is kept to define the necessary runtime protocol parameters. Information on how to maintain the COMCMDR.INI file, see Setting Up (Outbound) Commcommander Initialization Section Entries for NT/UNIX.

SETTING UP (OUTBOUND) COMMCOMMANDER SERVER

USING SNA FOR NT

Before running Commcommander Server Outbound, certain SNA configuration settings must be defined and activated. The following describes the types of SNA configuration settings that are necessary to run Commcommander Server Outbound.

Note After the MVS SNA configuration settings are in place, obtain from your MVS Network or Systems Administrator, the mode and LU values needed to configure NT SNA.

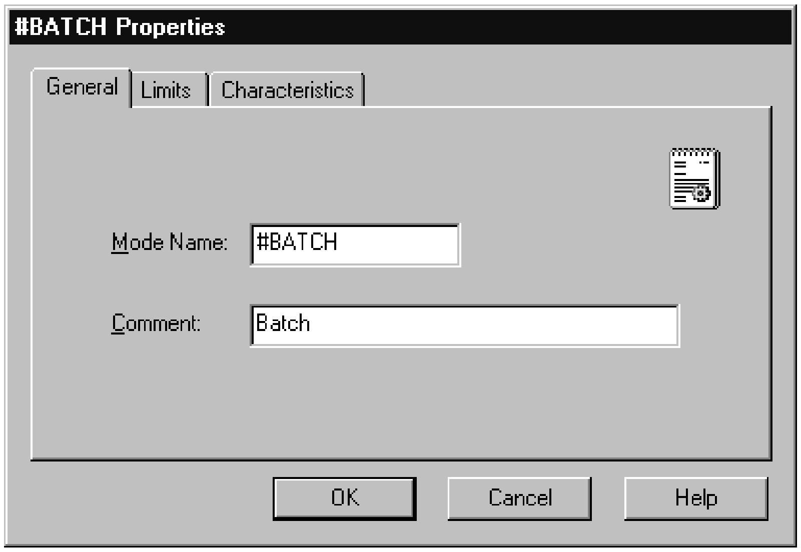
Defining Modes, Symbolic Destinations, and Initialization Entries

Values for modes and symbolic destination names are used in program-to-program communication within the SNA network. These definitions are handled by the SNA Server Administrator's utility. The following describes the values necessary for SNA APPC connections.

To Define the Mode

The APPC Mode Properties are used to establish the SNA session between the two node partners. These variables are used in establishing and negotiating sessions.

The following dialog is an example showing how to define Mode definitions. Examples for the Limits and Characteristics tabs follow this dialog:



Mode Name: #BATCH

Comment: Batch

Figure 3: Microsoft SNA Server/NT APPC Mode Properties

1. Type the following values into the corresponding fields on the **Limits** tab:
   - Parallel Session Limit — **8**
   - Minimum Contention Winner Limit — **4**
   - Partner Min Contention Winner Limit — **0**
   - Automatic Activation Limit — **0**

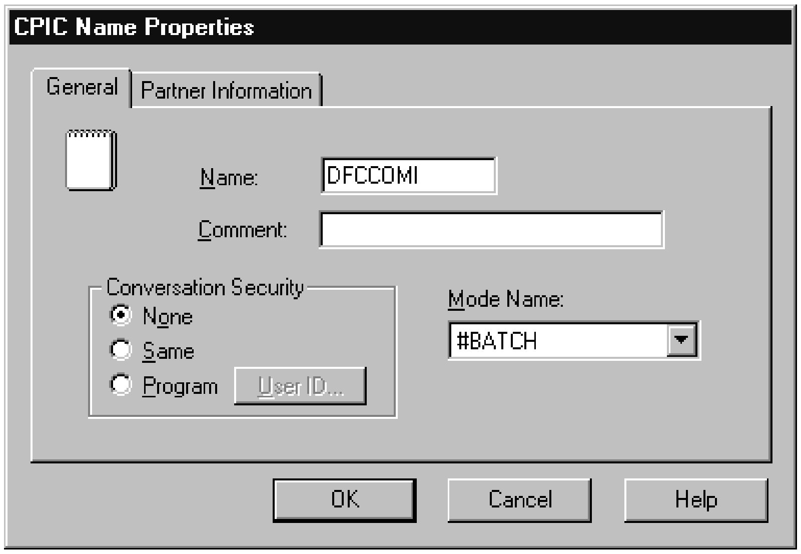
2. Type the following values into the corresponding fields on the **Characteristics** tab:

   - Pacing Send Count — **3**
   - Pacing Receive Count — **3**
   - Max Send RU Size — **1024**
   - Max Receive RU Size — **1024**

To Define the Symbolic Destination Name

The Symbolic Destination Name Properties are used for those Docucorp products where the conversation originates on the PC. This dialog contains the detailed variable information to establish a connection to the partner node in the network.

The following dialog is an example showing how to define Symbolic Destination Names. Examples for the Conversation Security, and the Partner Information tab follow this dialog:



Name: DFCCOMI
Comment:
Conversation Security: [✓ None] Same Program
Mode Name: #BATCH

Figure 4: Microsoft SNA Server/NT CPI-C Symbolic Destination Name Properties

1. In the Conversation Security group box, choose **Program** only if the conversation is secure on the MVS host; otherwise, choose **None**.
2. Type the following values into the corresponding fields on the **Partner Information** tab.

WARNING! The following values are shown only as examples. Please check with your network administrator for the correct entry.

- Partner TPName — the name of the application TP (e.g., **DFCCOMI**).
- Partner LU Name — the fully-qualified LU name (e.g., **MVSESA.ISICOM25**).

To Make the Changes Active

The SNA Server Administrator is used to configure your SNA Server. Have your network administrator perform all the necessary configurations for your server machine.

The following dialog is an example of the SNA Server Manager main dialog. From this dialog you can configure all necessary connection values. Also, use this dialog to Stop or Start the Microsoft SNA Server utility.

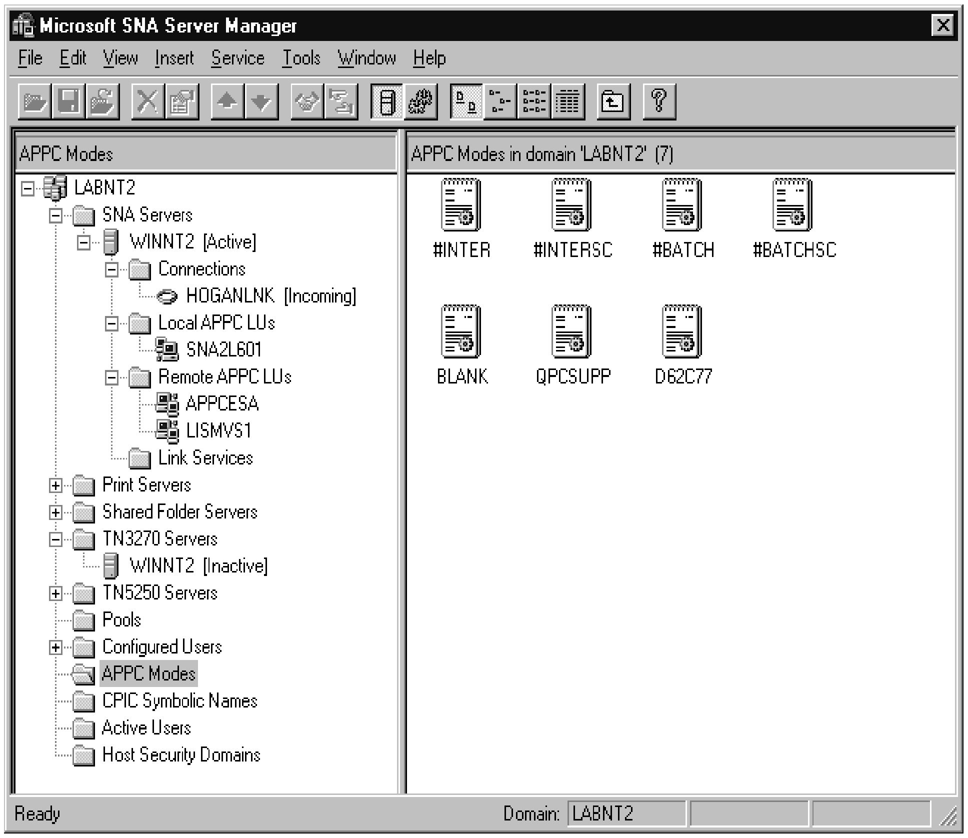


Figure 5: Microsoft SNA Server/NT SNA Server Manager

1. Select **Connections** from the left-hand panel (e.g., **HOGANLNK**).
2. From the **Service** menu, choose **Stop**, then **Start**.

Defining Initialization Entries in the COMCMDR.INI file

Commcommander Server Outbound utilizes an initialization control file, where information is kept to define the necessary runtime protocol parameters. Information on how to maintain the COMCMDR.INI file, see Setting Up (Outbound) Commcommander Initialization Section Entries for NT/UNIX.

SETTING UP (OUTBOUND) COMMCOMMANDER SERVER

SNA for AIX

Before running Commcommander Server Outbound, certain SNA configuration settings must be defined and activated. The following describes the types of SNA configuration settings that are necessary to run Commcommander Server.

Note After the MVS SNA configuration settings are in place, obtain from your MVS Network or Systems Administrator, the mode and LU values needed to configure AIX SNA.

Defining Modes, Side Information, and Initialization Entries

Values for modes and side information profiles are used in program-to-program communication within the SNA network. These definitions are handled by IBM's Communications Server utility.

To access the IBM Communications Server utility, follow the steps listed below:

From the "root" account, on the machine where the AIX Communication software is installed, enter **SMIT** and chose the following menu selections:

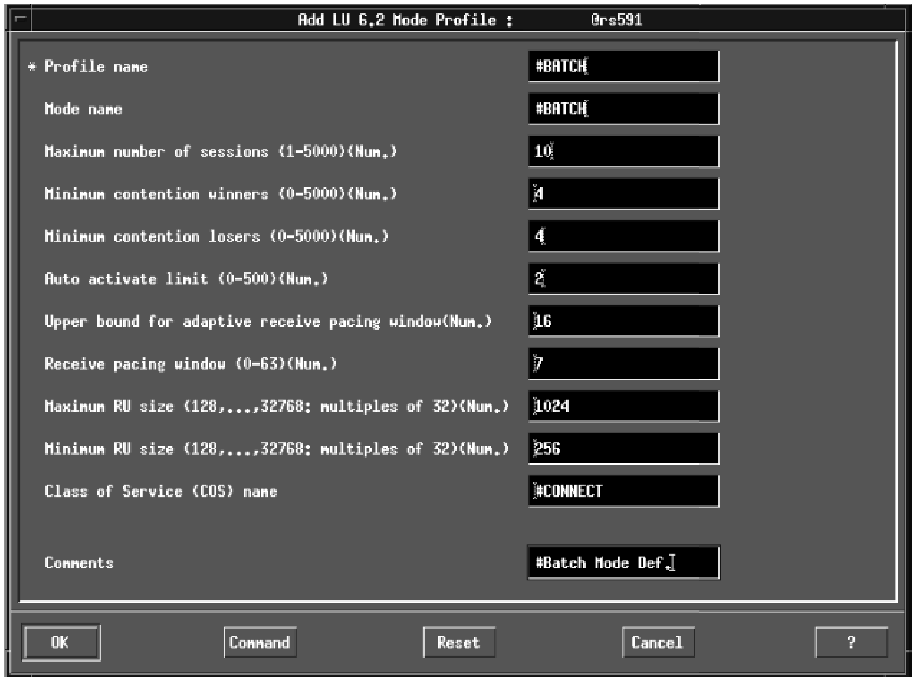
- Communications Applications and Services
- Communications Server for AIX
- Configure SNA Profiles
- Advanced Configuration
- Sessions
- Mode
- CPI Communications Side Information

The following describes the values necessary for SNA APPC connections.

To Define the Mode

The APPC Mode Profile values are used to establish the SNA session between the two node partners. These variables are used in establishing and negotiating sessions.

The following dialog is an example showing how to define Mode definitions. Examples for the Limits and Characteristics follow this
dialog:



Add LU 6.2 Mode Profile : @rs591

* Profile name	#BATCH
Mode name	#BATCH
Maximum number of sessions (1-5000)(Num.)	10
Minimum contention winners (0-5000)(Num.)	4
Minimum contention losers (0-5000)(Num.)	4
Auto activate limit (0-500)(Num.)	2
Upper bound for adaptive receive pacing window(Num.)	16
Receive pacing window (0-63)(Num.)	7
Maximum RU size (128,...,32768; multiples of 32)(Num.)	1024
Minimum RU size (128,...,32768; multiples of 32)(Num.)	256
Class of Service (COS) name	#CONNECT
Comments	#Batch Mode Def.

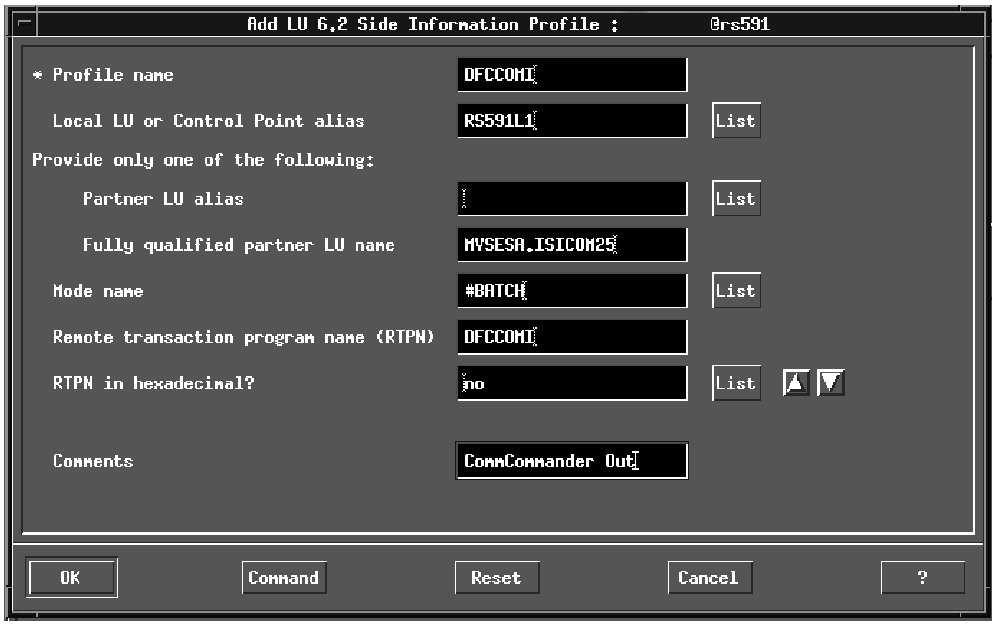
Figure 6: IBM's Communications Server SNA Mode Table Profile

1. Type the following values into the corresponding fields on the **Limits** tab:
   - Maximum number of sessions — **10**
   - Minimum contention winners — **4**
   - Minimum contention losers — **4**
   - Automatic activation limit — **0**
2. Type the following values into the corresponding fields on the **Characteristics** tab:
   - Upper bound for adaptive receive pacing window — **3**
   - Receive pacing window — **7**
   - Maximum RU size — **1024**
   - Minimum RU size — **256**
   - Class of Services — **#CONNECT**

To Define Side Information

The APPC Side Information Profile values are used for those Docucorp products where the conversation originates on the PC. This dialog contains the detailed variable information to establish a connection to the partner node in the network.

The following dialog is an example showing how to define Side Information Profile values. Examples for Partner Information follow this
dialog:



* Profile name	DFCCOMI
Local LU or Control Point alias	RS591L1
Provide only one of the following:	
Partner LU alias	
Fully qualified partner LU name	MVSESA.ISICOM25
Mode name	#BATCH
Remote transaction program name (RTPN)	DFCCOMI
RTPN in hexadecimal?	no
Comments	CommCommander Out

Figure 7: IBM's Communications Server SNA Side Information Profile

- Type the following **Partner Information** values:

WARNING! The following values are shown only as examples. Please check with your network administrator for the correct entry.

- Remote transaction program name — the name of the application TP (e.g., **DFCCOMI**).
- Fully qualified partner LU name — the fully-qualified LU name (e.g., **MVSESA.ISICOM25**).

To Make the Changes Active

1. Using the following menu selection path, select "Verify Configuration Profiles" from the Advanced Configuration menu:
   - Communications Applications and Services
   - Communications Server for AIX
   - Configure SNA Profiles
   - Advanced Configuration
   - Verify Configuration Profiles

2. After a successful verification, if necessary, stop and restart IBM's Communications Server

Defining Initialization Entries in the COMCMDR.INI file

Commcommander Server Outbound utilizes an initialization control file, where information is kept to define the necessary runtime protocol parameters. Information on how to maintain the COMCMDR.INI file, see Setting Up (Outbound) Commcommander Initialization Section Entries for NT/UNIX.

SETTING UP (OUTBOUND) COMMCOMMANDER SERVER USING TCPIP/SNA FOR MVS

To Customize the Outbound Communication Initialization File for MVS

Within the JCL members copied during the TAPEREAD job is the DFCINITO member. DFCINITO creates the DFCINIT dataset used by the Commcommander outbound message handler DFCCOMO.

1. Modify the JCL to meet your site's requirements. The **bold** portions of the JCL below represent the site-specific parts.

```
//-------- JOB CARD HERE --------------
//*
//*          *********************************************************
//*          DFCCOM INITIALIZATION FILE ALLOCATE AND LOAD
//*          Outbound message processor
//*          *********************************************************
//*
//STEP1   EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSIN    DD DUMMY
//SYSUT2   DD DSN=isi.DFCCOM.VnnRnn.DFCCOMO.DFCINIT,
//            DISP=(NEW,CATLG),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=80),
//            UNIT=SYSDA,
//            SPACE=(TRK,(1,2))
//SYSUT1   DD *

*****               Commcommander control entries go here       *****
***** You will set these values after the creation of the       *****
***** DFCINIT dataset                                           *****
*****                     See step 4 below.                     *****

/*
//
```

Figure 8: DFCINITO JCL

2. Submit the job.

3. Check the job log for errors when the job is complete. If necessary, correct the errors and resubmit the job.

4. Edit the DFCINIT dataset (e.g., **isi.DFCCOM.VnnRnn.DFCCOMO.DFCINIT**). Using the keyword definitions listed below, ensure the correct protocol and diagnostic values are set.

Note The following two figures are examples of the DFCINIT entries. Use the first example for TCP/IP applications, the second for APPC/MVS or ACF/VTAM applications. The examples represents the exact keywords needed for each selected protocol.

```
************************ TOP OF DATA *******************
%%% DFCCTL %%%VvRrMmYYMMDD    H
* CTLID-VVRRMM    COMMAND-----------------
* ..+....1....+....2....+....3....+....4....+....5....+..
DFCCTL  VvRrMm    PROTOCOL=TCP/IP
DFCCTL  VvRrMm    TCPNAME=xxxx
DFCCTL  VvRrMm    LOGFILE=OFF
DFCCTL  VvRrMm    TRACEFILE=OFF
DFCCTL  VvRrMm    LOGDATA=RECEIVE_OFF
DFCCTL  VvRrMm    LOGDATA=SEND_OFF
%%% DFCCTL %%%VvRrMmYYMMDD    T
********************** BOTTOM OF DATA *******************
```

Figure 9: Commcommander Server/MVS Communication Initialization File — TCP/IP

```
************************ TOP OF DATA *******************
%%% DFCCTL %%%VvRrMmYYMMDD    H
* CTLID-VVRRMM    COMMAND-----------------
* ..+....1....+....2....+....3....+....4....+....5....+..
DFCCTL  VvRrMm    PROTOCOL=APPC/MVS or aCF/VTAM
DFCCTL  VvRrMm    BASELU=isisp24   (ACF/VTAM only)
DFCCTL  VvRrMm    LOGFILE=OFF
DFCCTL  VvRrMm    TRACEFILE=OFF
DFCCTL  VvRrMm    LOGDATA=RECEIVE_OFF
DFCCTL  VvRrMm    LOGDATA=SEND_OFF
%%% DFCCTL %%%VvRrMmYYMMDD    T
********************** BOTTOM OF DATA *******************
```

Figure 10: Commcommander Server/MVS Communication Initialization File — ACF/VTAM or APPC/MVS

First, note that the data are enclosed within a header and trailer of:

```
%%% DFCCTL %%%VvRrMmYYMMDD     x
```

where

x is represented by either **H** for Header or **T** for Trailer. Don't change the header or trailer records unless advised by a Docucorp representative.

Records beginning with an asterisk (*) are comments. In the above example, two comment lines immediately follow the header record. Each of the non-comment records between the header and trailer contain **DFCCTL VvRrMm** in columns 1 through 14. Then, beginning in column 19, keywords and values are displayed.

The parameters in this file are fixed position and must begin in the positions specified by the scale comment record in the above example file.

Parameter	Value
PROTOCOL	Valid values are **TCP/IP**, or **APPC/MVS**, **ACF/VTAM**. The value defines the communications environment for Commcommander.
TCPNAME	**(TCP/IP only)** The Transmission Control Protocol name used by Commcommander. This is the JOB name of the TCP started task running on MVS, which controls the TCP/IP environment.
BASELU	**(ACF/VTAM only)** The APPC or LU name used by Commcommander. This is the *symbolic name* by which Commcommander is known in your SNA network. Your network administrator defines this name with an APPC macro in the VTAM Application Program Major Node table. You can use ISISP**xx** where **xx** is a number and value (e.g., **ISISP24**).
LOGFILE	Valid values are OFF and ON. When set to ON, Commcommander writes log entries to the DFCLOG file. Log data are recorded entries of selected events in Commcommander processing. Examples are the starting and stopping of conversations.
TRACEFILE	Valid values are OFF and ON. When set to ON, Commcommander writes trace information to the DFCTRACE file. The events and messages traced are determined by Docucorp routines. Relatively low level actions, such as those performed by the inbound message handler (DFCCOMI) and by the host VLAMcommander module (DFCVLM), are recorded as trace data.
LOGDATA (receive)	Valid values are RECEIVE_ON and RECEIVE_OFF. When set to RECEIVE_ON, a copy of the data records sent from Commcommander Server partner node is written to the LOG file. The data records include the actual data uploaded and program control blocks that identify the data and the processing the data are subject to by Commcommander Server/MVS.
LOGDATA (send)	Valid values are SEND_ON and SEND_OFF. When set to SEND_ON, a copy of the data records sent from Commcommander Server/MVS is written to the LOG file. The data records include the actual data downloaded and program control blocks that identify the data and the processing to which the data are subject by Commcommander Server. **NOTE:** Records beginning with "* ", an asterisk followed by a blank space, in positions 1 and 2 are treated as comment records and are ignored. Comment records can be loaded anywhere within the file.

The illustration below shows the relative scope of the message, log, and trace information generated by Commcommander. All message information is included in the generation of the log data. The trace data includes all log and message information. While you might be writing to separate log and messages files, all of the message file information is included in the log file.

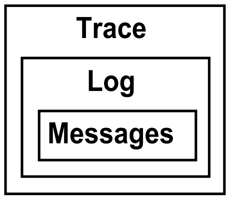


Figure 11: Message, Log, and Trace Information

Note While you can control the generation of trace and log information, message information is always written to the message file.

Setting Up (Outbound) Commcommander Initialization Section Entries for NT/UNIX

Tip If you've already set up the Initialization Entries, skip this section.

The keyword entries in the COMCMDR.INI file are use by Commcommander Server when establishing a session with a partner node. Commcommander Server uses these fields to determine the correct transportation protocol to use and transaction program to invoke.

To Set Up (Outbound) Commcommander Initialization Section Entries for NT/UNIX

1. Edit the COMCMDR.INI file, located in the directory where the COMCMDRO executable is located.
2. Create a section for the protocol information you will be using.
3. Save your changes.

There are three sample sections included in the COMCMDR.INI created during the PC server installation. The first two samples are used with Commcommander Server Inbound (COMCMDRI.EXE) TCP/IP processing. The last sample is used for the Commcommander Server Outbound (COMCMDRO.EXE). The following figure is a sample of the COMCMDR.INI for use with Commcommander Server Outbound.

```
;        Commcommander INI
;        (a semicolon in position one is a comment line)
;----------------------------------------------------------------------------
[GATE0]   Sample Commcommander Server Outbound parameters
;                            ----------------------------------------
;                            Gate name for Outbound Protocol (Gate/Port)
;                                 [default=same as section name above]
Gate=GATE0
;                            ----------------------------------------
;                            Maximum number of outbound conversations
;                                 [default = none]
MaxConversations=1
;                            ----------------------------------------
;                            Symbolic Destination name
;                                 [default = DFCCOMI]
Destination=mvsHost
;                            ----------------------------------------
;                            Symbolic Destination gate/port name (TCP/IP only)
;                                 [default = DFCCOMI]
DestinationPort=DFCCOMI
;                            ----------------------------------------
;                            Input Protocol type for LAN connections
;                                 NAMEDPIPE (NetBEUI)
;                                 SOCKET    (TCP/IP)
;                                 [default = SOCKET]
InputProtocol=SOCKET
;                            ----------------------------------------
;                            Output Protocol type for mainframe connections
;                                 CPIC      (SNA)
;                                 SOCKET    (TCP/IP)
;                                 (default = SOCKET)
OutputProtocol=SOCKET
;                            ----------------------------------------
;                            MVS Userid
;                                 [default=none]
Userid=
;                            ----------------------------------------
;                            MVS Password
;                                 [default=none]
Password=
;                            ----------------------------------------
;                            Trace Level used for problem determination
;                                 (default = 0)
TraceLevel=0
```

Figure 12: Commcommander Server Outbound

The following list describes the keywords and their values, for use with Commcommander Outbound Server.

Option	Purpose
[ ]	The value within the [ ] is the section entry name. This value is case sensitive. Its use to group the following keyword values together under a single name, that may be referenced by the user during Commcommander startup. There may be multiple sections in a single COMCMDR.INI file. The name you chose for each section, is strictly up to you.
Gate	The symbolic name of the port or gate used for inbound communications. This value is case sensitive. If this field is omitted, the section name will be used in its place. The value of this field is used for the inbound connection from the client workstation. The following are the selected protocol descriptions for this field: **TCP/IP for Sockets** This value must match the port name entry in the SERVICES file. **NAMEDPIPE** This value is arbitrary, assigned by the administrator of the Commcommander product. The value entered here must be used on the client workstation to connect to the named pipe.
MaxConversations	Maximum number of outbound conversations. Start with a value of 8 and adjust as needed.
Destination (Symbolic Destination name)	This value is case sensitive. The following are the selected protocol descriptions for this field: **TCP/IP for Sockets** This value must match the IP Address name entry in the ./ETC/HOSTS file, which is assigned to the host machine's IP Address. **CPIC (APPC/MVS and ACF/VTAM)** This value must match the Symbolic Destination Name defined in the Side Information Profile of the SNA Server utility.
DestinationPort (Symbolic Destination Port name)	This value is case sensitive. The following are the selected protocol descriptions for this field: **(TCP/IP for Sockets Only)** This value must match the port name entry in the ./ETC/SERVICES file, which is assigned to the host machine's Commcommander Server Port Number.
InputProtocol	Input Protocol name used for PC LAN connections. The valid values are as follows: **NAMEDPIPE** or **SOCKET**
OutputProtocol	Output Protocol name used for mainframe connections. The valid values are as follows: **CPIC (APPC/MVS and ACF/VTAM)** or **SOCKET**
Userid	**(SNA AIX Only)** This value is the name of an MVS user account ID used in conjunction with Password for mainframe access.

Option	Purpose
Password	**(SNA AIX Only)** This value is the password used in conjunction with Userid for mainframe access. NOTE: If you're using a mainframe security Document Package (such as RACF) and your mainframe resources (jobs and/or data sets) are secured, you'll need to supply values for the Userid and Password fields. The Userid and Password values are used by your mainframe security Document Package, to authorize the use of MVS resources. There are two situations that require different ways of handling the security parameters. • If you've secured the LU 6.2 conversation, supply the User ID and password necessary to authorize the conversation and/or the execution of the MVS Transaction Program (TP). The Commcommander Server/AIX obtains the User ID and password form the COMCMDR.INI file and uses these values with the CPI-C allocate conversation (cmallc) request. • If you don't have (or don't want) your MVS resources secured, and you've specified to RACF that the conversation is "already verified," do not send a User ID or password with CPI-C allocate conversation (cmallc) request. In other words, don't code any values in the COMCMDR.INI User ID and password parameters. A warning message will be issued whenever the Commcommander Server/AIX is started, but you can ignore the warning and processing will continue.
TraceLevel	Trace Level number used for problem determination. If you are having protocol trouble, Docucorp may ask you to generate a trace log, which contains detailed information about the Commcommander API call sequence. Before calling Docucorp's Hotline, try running with a tracelevel of "**5**". This will be helpful information used by the Hotline support analyst when solving your problem.

Using (Outbound) Commcommander Server/NT/UNIX

There are two major components in Commcommander Server, based on the origination point of the conversation. The type of component used depends on where the conversation originates.

If the data is being moved from the PC to the mainframe, the conversation originates on the PC; therefore, the Commcommander Server Outbound (**COMCMDRO**™ software is a Docucorp product) is used.

WARNING! For TCP/IP and ACF/VTAM mainframe environments: The mainframe component must be running before starting Commcommander Server Outbound. For more information, see Starting Commcommander in a TCP/IP or ACF/VTAM Environment.

If the mainframe component is stopped and re-started, you must stop and re-start Commcommander Server.

The following section describes how to start Commcommander Server Outbound.

Tip If you are using the SNA (CPIC) protocol, ensure that SNA Server software is running on your server machine. See the "Setup" section of the selected platform for more information.

.

To Start Commcommander Server Outbound - NT Service

1. Select **Start/Settings/Control Panel/Services** to display the System Services dialog box.
2. Select **Commcommander 213 Outbound**.
3. Select **Start** to start the server application.

To Start Commcommander Server Outbound - Non-NT

Note The Gate name was entered during installation. If you want to change the desired Gate name, perform the following steps:

1. Select the **Commcommander Outbound** icon located in the folder where you installed Commcommander.
2. Click the right mouse button to display the context menu and select **Properties**.
3. Add the required **Gate** parameter to the **Target** command. For more information, see the description of **Gate** in the following section's parameter explanations).
4. Select **OK**.
5. Proceed to the next step.

Double-click the **Commander Outbound** icon to start the process.

To Start Commcommander Server Outbound - UNIX

Change to the Commcommander Server directory from a command prompt and enter the following command to start the Commcommander Server Outbound (comcmdro executable):

```
comcmdro  Gate  [MaxConversations
                Destination/DestinationPort/Userid/Password
                InputProtocol OutputProtocol
                 TraceLevel]
```

Tip The parameters on the command line are positional and mandatory, up to a point. If you want to specify **InputProtocol**, for example, you must include all parameters up to and including **InputProtocol**. **OutputProtocol** and **TraceLevel**, however, are supplied with the values stored in the COMCMDR.INI file.

The Gate parameter is mandatory. The parameter values within the [ ] are optional. If present, these values will override their associated values in the COMCMDR.INI file.

Parameter	Explanation
Gate	The value of this field is used for the inbound connection from the client workstation. This field is mandatory. The value of this field is case sensitive, and should match the section name in the COMCMDR.INI file, where the Commcommander Server Outbound parameters are defined. Commcommander Server Outbound can be started using this value as its only command line parameter (preferred method). Values for the remaining parameters will be taken from the COMCMDR.INI file. If the specified value does not match an entry in the COMCMDR.INI file, all of the Commcommander Server Outbound parameters must be supplied on the command line. In the case stated above, all protocol specific rules apply to the specified value. If you are using TCP/IP Sockets, this must be the name of the port address defined in your services file. If you are using NetBEUI Named Pipes, by supplying a name in this field it names and therefore creates the pipe. This same name must be used on the client workstation machine to connect to the named pipe.
Destination /DestinationPort /Userid /Password	This combination of fields, provide values for the following COMCMDR.INI file section entries: Destination, Destination Port, Userid, and Password. If you wish to specify these fields on a command line startup, each of these fields to be separated by a **"/"**. Listed below are the possible usages for the field: **TCP/IP**Destination/Destination Port (e.g., mvsIPA/dfccomiPORT) The Userid/Password values are not needed for the TCP/IP Socket protocol. **SNA** Destination (e.g., DFCCOMI) **-or-** **(AIX only)** Destination/Destination/Port/Userid/Password (e.g., **DFCCOMI/comi/mvsUser/mvsPassword**) The example above using Userid/Password is only used for the SNA CPIC protocol when the resources used by the Commcommander Server Inbound (MVS), are secure. The Destination name is case-sensitive and must exactly match the name defined in the SNA configuration.
InputProtocol	Input Protocol name used for PC LAN connections. The valid values are as follows: **NAMEDPIPE** or **SOCKET**
OutputProtocol	Output Protocol name used for mainframe connections. The valid values are as follows: **CPIC (APPC/MVS and ACF/VTAM)** or **SOCKET**
TraceLevel	Trace Level number used for problem determination. If you are having protocol trouble, Docucorp may ask you to generate a trace log, which contains detailed information about the Commcommander API call sequence. Before calling Docucorp's Hotline, try running with a TraceLevel of "**5**". This will be helpful information used by the Hotline support analyst when solving your problem.

Commcommander will display "Ready" when it is successfully running.

To Stop Commcommander Server Outbound

Press **Ctrl+C** on the screen where Commcommander Server Outbound is running to end the process. You should stop Commcommander Server Outbound before stopping the SNA Server software.

To Review Messages

Browse any and all Commcommander error messages by viewing the COMCMDR.LOG file, which is located in the same directory as the COMCMDR executable file. The COMCMDR.LOG file contains any protocol errors or trace messages produced during a Commcommander session. The messages are written chronologically from top to bottom. Therefore to view the most recent messages go to the bottom of the file and scroll backwards.

Using (Outbound) Commcommander Server/MVS

Commcommander Server Outbound MVS (DFCCOMO) is invoked via an outbound driver application. See the "Using" section of the desired driver for more information.

Note Ensure that the values in the Communications Initialization File for outbound request are specified and valid. For more information, see To Customize the Outbound Communication Initialization File for MVS.

Commcommander Server Inbound

Because Commcommander Server is supported on several platforms with optional protocols, this guide provides separate procedures for each platform protocol combination.

To Set Up	See
Commcommander Server using TCPIP for NT/UNIX	Setting Up (Inbound) Commcommander Server using TCPIP for NT/UNIX
Commcommander Server using SNA for NT	Setting Up (Inbound) Commcommander Server using SNA for NT
Commcommander Server using SNA for AIX	Setting Up (Inbound) Commcommander Server using SNA for AIX
Commcommander Server using TCPIP/ SNA for MVS	Setting Up (Inbound) Commcommander Server Using TCPIP/SNA for MVS

Setting Up (Inbound) Commcommander Server using TCPIP for NT/UNIX

Before running Commcommander Server Inbound, certain TCP/IP configuration settings have to be defined and activated. The following describes the types of TCP/IP configuration settings that are necessary to run Commcommander Server Inbound.

Defining the Service Port Number, and Initialization Entries

Values for service port number are used in program-to-program communication within the TCP/IP network. A unique service port number is necessary for the establishment of each TCP/IP session.

To Define the Service Port Number

Use the following steps when you define the port number:

1. Edit the **SERVICES** file.

   This file is normally located in the **c:\windows\system32\drivers\etc (NT) or the root /etc (UNIX)** directory. The actual location of the Services file may be in a different directory depending on your Windows/NT system directory name and/or the specific vendor's TCP/IP you have installed.

2. Enter a unique port number and associated name, for your inbound socket connection. The port address you assign must be unique within this services file.

   Example:

```
GATE1     7168/tcp     #Commcommander server input port number
```

3. Save the **SERVICES** file.

WARNING! Commcommander Service Port numbers should be greater than 1024, and not a duplicate of any other number in the SERVICES file.

Defining Initialization Entries in the COMCMDR.INI file

Commcommander Server Inbound utilizes an initialization control file, where information is kept to define the necessary runtime protocol parameters. Information on how to maintain the COMCMDR.INI file, see Setting Up (Inbound) Commcommander Initialization Section Entries for NT/UNIX.

SETTING UP (INBOUND) COMMCOMMANDER SERVER USING SNA FOR NT

Before running Commcommander Server Inbound, certain SNA configuration settings must be defined and activated. The following describes the types of SNA configuration settings that are necessary to run Commcommander Server Inbound.

Defining Transaction Programs

Values for transaction program profiles are used in program-to-program communication within the SNA network. These definitions are handled by the Invokable TP Setup utility located in the directory where Commcommander executable modules are installed **(c:\comcmdr\tprgstry.exe)**.

An invokable TP is one that can be invoked by another TP. Invokable TPs must be registered (through Registry entries or environment variables) on its local system, so that it can be identified to the SnaBase component of the SNA Server client software, as a notification that they are available for incoming requests. The registered information must specify a unique TP name.

The registered information may also specify the local LU alias that the invokable TP will use. If no local LU alias is registered with auto started TPs, the resulting SNA Server configuration can be more flexible in responding to invoking requests.

Invokable TPs must be configured as queued or non-queued.

Queued TPs

If a TP is configured as queued, incoming requests are queued, and then sent only when the invokable TP wishes to accept a conversation. If a copy of the TP is not yet running, one is started when the first incoming request specifies that TP.

Non-queued TPs

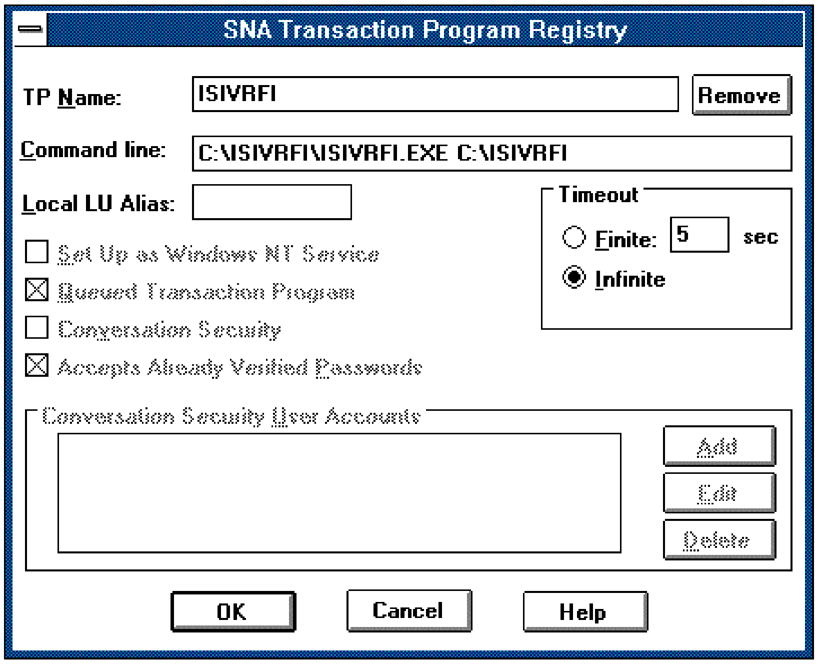
If a TP is configured as non-queued, every time the TP is requested, a new copy will be started.

To Define the Transaction Program (TP)

The SNA Transaction Program Registry values are used for those Docucorp products where the conversation originates on the MVS mainframe.

Use the following dialog when you define the TP values.

Run **TPRGSTRY.EXE** by double-clicking on the TP Setup icon located in the ISI Utilities program group (created during the Commcommander installation), to activate this dialog:



TP Name: ISIVRFI

Command line: C:\ISIVRFI\ISIVRFI.EXE C:\ISIVRFI

Local LU Alias:

Timeout: [ ] Finite: 5 sec [✓ Infinite]

[ ] Set Up as Windows NT Service
[✓] Queued Transaction Program
[ ] Conversation Security
[✓] Accepts Already Verified Passwords

Conversation Security User Accounts

Figure 13: Invokable TP Setup

Note The dialog above is an example of the ISIVRFI driver. Other inbound drivers will use a different name in the entries shown. Refer to the "Setup" section of the desired driver for values to enter in the parameters shown.

Only the TP Name and Command line values need to be entered; the other values are handled by the TP Registry application.

To Make the Changes Active

After making the necessary entries to the Transaction Program Registry and selecting **OK**, the new Invokable TP is ready for use.

Note Before an invokable TP is executed, make sure the **TPSTART** program is running. TPSTART is located in the directory where Commcommander executable modules are installed **(c:\comcmdr\tpstart.exe)**. You can start this program by double clicking on the **TP Start** icon, located in ISI Utilities program group, created during the Commcommander installation procedure.

Setting Up (Inbound) Commcommander Server using SNA for AIX

Before running Commcommander Server Inbound, certain SNA configuration settings must be defined and activated. The following describes the types of SNA configuration settings that are necessary to run Commcommander Server Inbound.

Defining Transaction Programs

Values for transaction program profiles are used in program-to-program communication within the SNA network. These definitions are handled by IBM's Communications Server utility.

To access the IBM Communications Server utility (version 4.2), follow the steps listed below:

From the "root" account, on the machine where the AIX Communication software is installed, enter **SMIT** and chose the following menu selections:

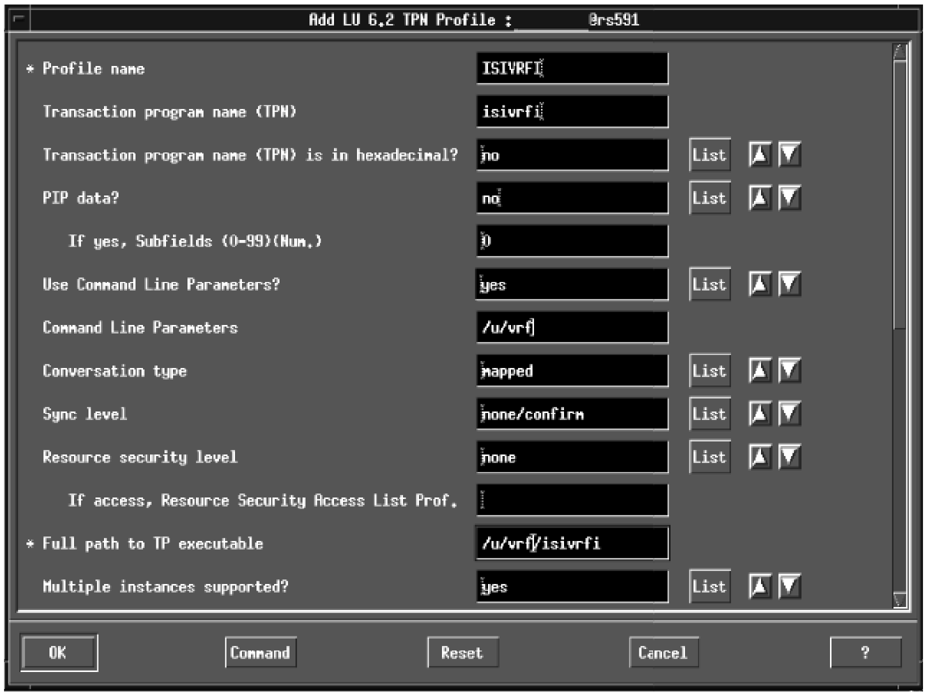
- Communications Applications and Services
- Communications Server for AIX
- Configure SNA Profiles
- Advanced Configuration
- Sessions
- Transaction Program Name (TPN)

The following describes the values necessary for SNA APPC connections.

To Define the Transaction Program (TP)

The APPC Transaction Program Name Profile (TPN) values are used for those Docucorp products where the conversation originates on the MVS mainframe.

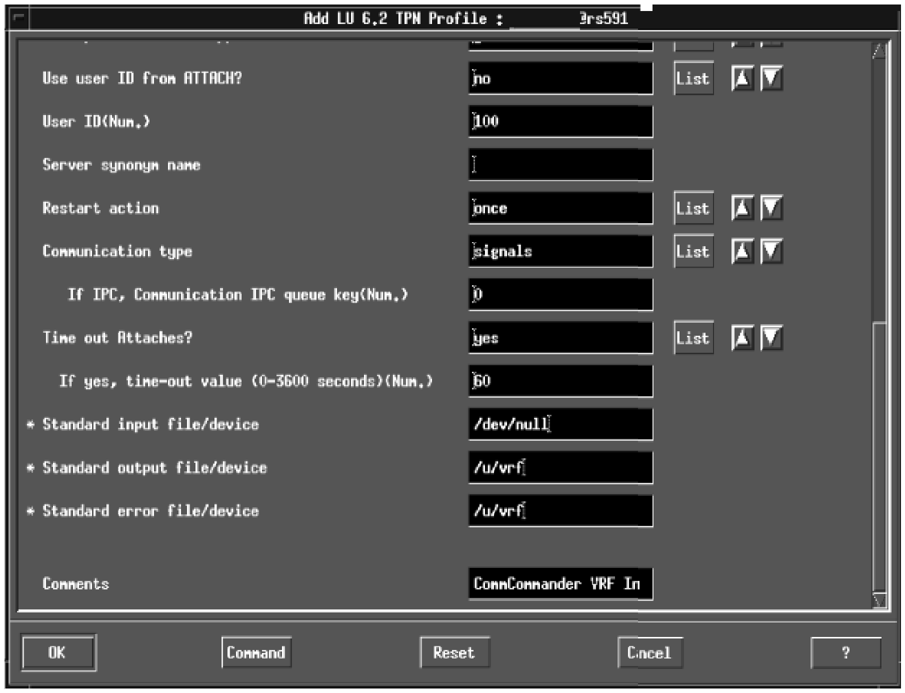
The following describes the values necessary for SNA APPC connections.



Add LU 6.2 TPN Profile : @rs591

* Profile name	ISIVRFI
Transaction program name (TPN)	isivrfi
Transaction program name (TPN) is in hexadecimal?	no
PIP data?	no
If yes, Subfields (0-99)(Num.)	0
Use Command Line Parameters?	yes
Command Line Parameters	/u/vrf
Conversation type	mapped
Sync level	none/confirm
Resource security level	none
If access, Resource Security Access List Prof.	
* Full path to TP executable	/u/vrf/isivrfi
Multiple instances supported?	yes

Figure 14: IBM's Communications Server Transaction Program Name (TPN) Profile — Part 1 of 2



Add LU 6.2 TPN Profile : @rs591

Use user ID from ATTACH?	no
User ID(Num.)	100
Server synonym name	
Restart action	once
Communication type	signals
If IPC, Communication IPC queue key(Num.)	0
Time out Attaches?	yes
If yes, time-out value (0-3600 seconds)(Num.)	60
* Standard input file/device	/dev/null
* Standard output file/device	/u/vrf
* Standard error file/device	/u/vrf
Comments	CommCommander VRF In

Figure 15: IBM's Communications Server Transaction Program Name (TPN) Profile — Part 2 of 2

Tip By coding the “Standard input file” as [/dev/null] and using TraceLevel=n (where n is > 0), will keep transaction processes from hanging the system waiting for a console response (enter) to continue processing.

By coding the “Standard output file” and “Standard error file” to a file on the VRF account (example: [/u/isivrf/stdout] and [/u/isivrf/stderr]), will allow you to view any error and debugging messages by displaying these files. If you code these as [/dev/console], the messages would be routed to the system console and possibly scrolled off the screen and lost.

Note The dialog above is an example of the ISIVRFI driver. Other inbound drivers will use a different name in the entries shown. Refer to the “Setup” section of the desired driver for values to enter in the parameters shown.

To Make the Changes Active

1. From the “root” account, on the machine where the AIX Communication software is installed, enter **SMIT** and chose the following menu selections:
   - Communications Applications and Services
   - Communications Server for AIX
   - Configure SNA Profiles
   - Advanced Configuration
   - Verify Configuration Profiles
2. Stop and then restart the AIX SNA Server (if necessary) after successful verification.

SETTING UP (INBOUND) COMMCOMMANDER SERVER USING TCPIP/SNA FOR MVS

Preparing for use in a Multiple MVS CPU Environment

In a multiple CPU environment, processing takes place on several CPUs which share common resources including DASD and data sets. The operating system ensures that these resources are allocated properly. However, in some instances additional communication between the application program and the operating system is required to control resource utilization. Docucorp software relies on the operating system to maintain information about the use of resources through system Enqueue/ Dequeue macros and specially defined Queue names.

In a multiple CPU environment, the global resource serialization package is responsible for communicating information about system Enqueues (ENQ) and Dequeues (DEQ) from one CPU to another. This is done through the use of an inclusion and exclusion list. If a Queue name is on the **inclusion list**, when an ENQ or DEQ is issued for that queue name the effects of the ENQ or DEQ will be communicated to all of the CPUs on the system.

If a Queue name is on the **exclusion list**, when an ENQ or DEQ is issued for that queue name, the effects of the ENQ or DEQ will take effect **only on the CPU under which processing is currently taking place**.

Some global resource serialization packages treat all queue names as part of the inclusion list unless they are explicitly placed in the exclusion list. Other packages treat all queue names as part of the exclusion list unless they are explicitly placed in the inclusion list. Since all serialization packages vary in the defaults used, **you must verify** that the queue names used by Docucorp will be treated as part of the **inclusion list**.

System Enqueue/Dequeues

Docucorp software uses the following queue names:

ISISGNON	Docuwise
DVSSGNON	Docuvise
HIAMEXTQ	HIAM, Docuwise, Docuvise
HIAMINTQ	HIAM, Docuwise, Docuvise
*** VLAM-MDB**	VLAM, Docuwise, Docuvise, Documerge,
	Commcommander (Inbound) MVS *Multi-Tasking Environment*

You must verify that these queue names will be treated as part of the inclusion list by your global resource serialization software.

* **VLAM-MDB** - Resource name **DFCSERV** in a Commcommander Multi-Tasking Environment is used to control the creation, and updating of the Message, Log, and Trace datasets. It's recommended that this resource name is added to the VLAM-MDB queue, ***exclusion list***, when logging and tracing Commcommander transaction throughput (for debug purposes.)

Logging and tracing should only be performed when gathering information for client support to analyze, and returned to normal as soon as the necessary data is captured.

To Customize the Inbound Communication Initialization File for MVS

Within the JCL members copied during the TAPEREAD job is the DFCINITI member. DFCINITI creates the DFCINIT dataset used by the Commcommander inbound message handler DFCCOMI.

1. Modify the JCL to meet your site's requirements. The **bold** portions of the JCL below represent the site-specific parts.

```
//-------- JOB CARD HERE --------------
//*
//*          ********************************************************
//*          DFCCOM INITIALIZATION FILE ALLOCATE AND LOAD
//*          Outbound message processor
//*          ********************************************************
//*
//STEP1   EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSIN    DD DUMMY
//SYSUT2   DD DSN=isi.DFCCOM.VnnRnn.DFCCOMI.DFCINIT,
//            DISP=(NEW,CATLG),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=80),
//            UNIT=SYSDA,
//            SPACE=(TRK,(1,2))
//SYSUT1   DD *

*****            Commcommander control entries go here          *****
***** You will set these values after the creation of the       *****
***** DFCINIT dataset                                           *****
*****                      See step 4 below.                    *****

/*
//
```

Figure 16: DFCINITI JCL

2. Submit the job.
3. Check the job log for errors when the job is complete. If necessary, correct the errors and resubmit the job.
4. Edit the DFCINIT dataset (e.g., **isi.DFCCOM.VnnRnn.DFCCOMI.DFCINIT**). Using the keyword definitions listed below, ensure the correct protocol and diagnostic values are set.

Note The following two figures are examples of the DFCINIT entries. Use the first example for TCP/IP applications, the second for APPC/MVS or ACF/VTAM applications. The examples represents the exact keywords needed for each selected protocol.

```
************************ TOP OF DATA ********************
%%% DFCCTL %%%VvRrMmYYMMDD    H
* CTLID-VVRRMM    COMMAND-----------------
* ..+....1....+....2....+....3....+....4....+....5....+..
DFCCTL  VvRrMm    PROTOCOL=TCP/IP
DFCCTL  VvRrMm    TCPNAME=xxxx
DFCCTL  VvRrMm    TCPPORT=nnnn
DFCCTL  VvRrMm    LOGFILE=OFF
DFCCTL  VvRrMm    TRACEFILE=OFF
DFCCTL  VvRrMm    LOGDATA=RECEIVE_OFF
DFCCTL  VvRrMm    LOGDATA=SEND_OFF
%%% DFCCTL %%%VvRrMmYYMMDD    T
********************** BOTTOM OF DATA ********************
```

Figure 17: Commcommander Server/MVS Communication Initialization File — TCP/IP

```
************************ TOP OF DATA *******************
%%% DFCCTL %%%VvRrMmYYMMDD    H
* CTLID-VVRRMM    COMMAND-----------------
* ..+....1....+....2....+....3....+....4....+....5....+..
DFCCTL  VvRrMm    PROTOCOL=APPC/MVS or aCF/VTAM
DFCCTL  VvRrMm    BASELU=isisp25   (ACF/VTAM only)
DFCCTL  VvRrMm    LOGFILE=OFF
DFCCTL  VvRrMm    TRACEFILE=OFF
DFCCTL  VvRrMm    LOGDATA=RECEIVE_OFF
DFCCTL  VvRrMm    LOGDATA=SEND_OFF
%%% DFCCTL %%%VvRrMmYYMMDD    T
********************** BOTTOM OF DATA ********************
```

Figure 18: Commcommander Server/MVS Communication Initialization File — ACF/VTAM or APPC/MVS

First, note that the data are enclosed within a header and trailer of:

```
%%% DFCCTL %%%VvRrMmYYMMDD    x
```

where

x is represented by either **H** for Header or **T** for Trailer. Don't change the header or trailer records unless advised by a Docucorp representative.

Records beginning with an asterisk (*) are comments. In the above example, two comment lines immediately follow the header record. Each of the non-comment records between the header and trailer contain **DFCCTL VvRrMm** in columns 1 through 14. Then, beginning in column 19, keywords and values are displayed.

Parameter	Value
PROTOCOL	Valid values are **APPC/MVS**, **ACF/VTAM**, or **TCP/IP**. The value defines the communications environment for Commcommander.
TCPNAME	**(TCP/IP only)** The Transmission Control Protocol name used by Commcommander. This is the JOB name of the TCP started task running on MVS, which controls the TCP/IP environment.
TCPPORT	**(TCP/IP only)** The Transmission Control Protocol port used by Commcommander. This should be a numeric value that is used by TCP to establish a session with a partner application. This address must match the same port address defined for the partner Commcommander Server "SERVICES" file.
BASELU	**(ACF/VTAM only)** The APPC or LU name used by Commcommander. This is the *symbolic name* by which Commcommander is known in your SNA network. Your network administrator defines this name with an APPC macro in the VTAM Application Program Major Node table. You can use ISISP**xx** where **xx** is a number and value (e.g., **ISISP25**).
LOGFILE	Valid values are OFF and ON. When set to ON, Commcommander writes log entries to the DFCLOG file. Log data are recorded entries of selected events in Commcommander processing. Examples are the starting and stopping of conversations.
TRACEFILE	Valid values are OFF and ON. When set to ON, Commcommander writes trace information to the DFCTRACE file. The events and messages traced are determined by Docucorp routines. Relatively low level actions, such as those performed by the inbound message handler (DFCCOMI) and by the host VLAMcommander module (DFCVLM3), are recorded as trace data.

Parameter	Value
LOGDATA (receive)	Valid values are RECEIVE_ON and RECEIVE_OFF. When set to RECEIVE_ON, a copy of the data records sent from Commcommander Server partner node is written to the LOG file. The data records include the actual data uploaded and program control blocks that identify the data and the processing the data are subject to by Commcommander Server/MVS.
LOGDATA (send)	Valid values are SEND_ON and SEND_OFF. When set to SEND_ON, a copy of the data records sent from Commcommander Server/MVS is written to the LOG file. The data records include the actual data downloaded and program control blocks that identify the data and the processing to which the data are subject by Commcommander Server.

Note Records beginning with "*", an asterisk followed by a blank space, in positions 1 and 2 are treated as comment records and are ignored. Comment records can be loaded anywhere within the file.

The illustration below shows the relative scope of the message, log, and trace information generated by Commcommander. All message information is included in the generation of the log data. The trace data includes all log and message information. While you might be writing to separate log and messages files, all of the message file information is included in the log file.

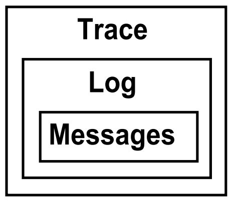


Figure 19: Message, Log, and Trace information

Note While you can control the generation of trace and log information, message information is always written to the message file.

To Update the Startup JCL (for TCP/IP and ACF/VTAM versions)

Below is the JCL to start the mainframe Inbound Message Handler (DFCCOMI). The message handler runs as a started task until terminated by the TERMINATOR job. This JCL can be found in the STARTUP member of the JCL library created as part of the installation process. The **bold** portions represent the portions you must change to meet your site's requirements. When the changes are complete, submit the job.

```
//---------- JOB CARD HERE ----------
//*
//*            ********************************************************
//*            ISI COMMUNICATIONS STARTUP  -  COMMCOMMANDER SERVER/MVS
//*            ********************************************************
//*
//JOBLIB   DD   DISP=SHR,DSN=isi.DFCCOM.VnnRnn.LOADLIB
//*         DD   DISP=SHR,DSN=isi.product.VnnRnn.LOADLIB  (for VLAM)
//DFCCOMI  EXEC PGM=DFCCOMI,4M
//SYSUDUMP DD SYSOUT=*
//SYSABEND DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//DEBUG    DD SYSOUT=*
//DFCMSG   DD SYSOUT=*,DCB=BLKSIZE=2048
//DFCLOG   DD SYSOUT=*,DCB=BLKSIZE=2048
//DFCTRACE DD SYSOUT=*,DCB=BLKSIZE=2048
//DFCINIT  DD DISP=SHR,DSN=isi.DFCCOM.VnnRnn.DFCCOMI.DFCINIT
//* -----------------------------------------------------------------
//*
//*           The following file required for the VLAM Driver
//*
//* -----------------------------------------------------------------
//*VLMLIBNI DD DISP=SHR,DSN=isi.DFCCOM.VnnRnn.VLMLIBNI
//* -----------------------------------------------------------------
//*
//*           The following statements required for the  VRF Driver
//*           Upload to MVS
//*
//* -----------------------------------------------------------------
//*                                             DFCVRFI input control
//*DFCVRFCT DD DISP=SHR,DSN=isi.DFCCOM.VnnRnn.DFCVRFI.CTL
//*                                             DFCVRFI output VRF
//*DFCVRFOT DD DISP=SHR,DSN=isi.DFCCOM.VnnRnn.DFCVRFOT.VRF
//*                                             DFCVRFI output AUDIT
//*DFCVRFAU DD DISP=SHR,DSN=isi.DFCCOM.VnnRnn.DFCVRFI.AUDIT
//* -----------------------------------------------------------------
//*
//*           The following statements required for the JES Driver
//*
//* -----------------------------------------------------------------
//*
//*           DCB information should be the same for all JESO print
//*           files. Use the largest LRECL and BLKSIZE of all DD'S
//*
//*           OUTPUT statement NAME must be same as DDNAME
//*           Following will route to Metacode printer
//*
//*DFCJESM  OUTPUT CLASS=V,
//*DFCJESM  DD SYSOUT=(,),FREE=CLOSE,
//*            OUTPUT=*.DFCJESM,
//*            DCB=(RECFM=VBM,LRECL=8205,BLKSIZE=8209)
//*
//*           -----------------------------------------------------
//*           OUTPUT statement NAME must be same as DDNAME
//*           following will route to AFP printer
//*
```

Continued on next page

Continued from previous page

```
//*DFCJESA  OUTPUT CLASS=W,FORMDEF=A10111,DATACK=BLOCK
//*DFCJESA  DD SYSOUT=(,),FREE=CLOSE,
//*            OUTPUT=*.DFCJESA,
//*            DCB=(RECFM=VBM,LRECL=8205,BLKSIZE=8209)
//*
//
```

Figure 20: STARTUP JCL

Modify the following values when setting up the Startup JCL:

Parameter	Value
Job Card	Replace with your MVS job card.
JOBLIB	Specify the Commcommander Load Library. If you're installing and using the Commcommander VLAM Driver, you'll concatenate the Load Library containing VLAM.
DFCINIT	The Commcommander Initialization File. For comprehensive information on the DFCINIT file, see To Customize the Inbound Communication Initialization File for MVS. NOTE: The following JCL statements are specific to the selected Commcommander Driver you plan to use. See the "Setup" section of each desired driver for additional JCL changes.
VLMLIBNI	The Commcommander VLAM Driver/MVS Settings File.
DFCVRFCT	The Commcommander VRF Driver/MVS Control File.
DFCVRFOT	The Commcommander VRF Driver/MVS Output File.
DFCVRFAU	The Commcommander VRF Driver/MVS Audit File.
DFCJESM	The Commcommander JES Driver/MVS Output Destination for XEROX print streams.
DFCJESA	The Commcommander JES Driver/MVS Output Destination for IBM AFP print streams.

WARNING! The Commcommander VLAM Driver requires Docucorp's Virtual Library Access Method (VLAM), which is distributed with other Docucorp products (e.g., Documerge).

You must update the Commcommander/MVS Server JCL to include the Load Library of the product containing VLAM. This Loadlib should be concatenated below the Commcommander Server Joblib.

Note It's recommended that you use the JCL members copied from the distribution tape. The examples in this manual are only for illustrative purposes.

Note You should set up a separate (Outbound-style) DFCINIT file for use with the Terminate JCL. For more information, see To Customize the Outbound Communication Initialization File for MVS.

To Update the Terminate JCL (for TCP/IP and ACF/VTAM versions)

To stop or bring down the mainframe Inbound Message Handler, you must run the DFCUTL communications utility using the TERMINATE command. The **bold** portions represent the portions you must change to meet your site's requirements. When the changes are complete, submit the job. Note that you must start up the Inbound Message Handler before Commcommander can be operational again.

```
//-------- JOB CARD HERE ----------
//*
//*         *********************************************************
//*         ISI COMMUNICATIONS UTILITY  -  STOP COMMCOMMANDER SERVER
//*        *********************************************************
//*
//JOBLIB   DD   DISP=SHR,DSN=isi.DFCCOM.VnnRnn.LOADLIB
//DFCUTL   EXEC PGM=DFCUTL,REGION=2M
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSABOUT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//DEBUG    DD SYSOUT=*
//DFCINIT  DD DISP=SHR,DSN=isi.DFCCOM.VnnRnn.DFCCOMO.DFCINIT
//DFCMSG   DD SYSOUT=*,DCB=BLKSIZE=2048
//DFCLOG   DD SYSOUT=*,DCB=BLKSIZE=2048
//DFCTRACE DD SYSOUT=*,DCB=BLKSIZE=2048
//DFCUTLCT DD *,DCB=BLKSIZE=80
*
* The following control card information is only for TCP/IP
* CCR ID------VVRRMM-DATE-----TPNAME--HOSTNAMEPORTADDR
* ..+....1....+....2....+....3....+....4....+....5....+....6....+....7....+
DFCUTL_TPN    VvRrMmYYMMDD    DFCCOMI hostnameportaddr
DFCUTL_COMMANDTERMINATE
*
* The following control card information is only for ACF/VTAM
* CCR ID------VVRRMM-DATE-----TPNAME--PARTLU--LOGMODE-SERVER--
* ..+....1....+....2....+....3....+....4....+....5....+....6....+....7....+...
* DFCUTL_TPN    VvRrMmYYMMDD    DFCCOMI lulululumodename
* DFCUTL_COMMANDTERMINATE
/*
//
```

Figure 21: TERMINATE JCL

Modify the following values when setting up the Terminate JCL:

Parameter	Value
Job Card	Replace with your MVS job card.
JOBLIB	Specify the Commcommander Load Library.
DFCINIT	The Commcommander Initialization File. For comprehensive information on the DFCINIT file, see To Customize the Outbound Communication Initialization File for MVS.

Parameter	Value
DFCUTLCT	The Commcommander Utility Driver/MVS Control File. **TCPIP** **DFCUTL_TPN** (pos. 1 - 14) — This control card describes the protocol information needed to connect to the Commcommander Server you wish to terminate. The bold entries listed below, describe the keywords for this control card: • **VnRrMmYYMMDD** (pos. 15 - 26) — This is the version, release, modification, year, month, and day of the current Commcommander product. This field is shipped with the correct value, therefore, no modifications are necessary at this time. • **DFCCOMI** (pos. 31 - 38) — This is the name of the Commcommander Server you wish to terminate. This field is shipped with the correct value (DFCCOMI), therefore, no modifications are necessary at this time. • **Hostname** (pos. 39 - 46) — This is the symbolic name assigned to the host machine's IP address, which is specified in the MVS host file. • **Portaddr** (pos. 47 - 54) — This is number of the TCPIP port address used when you started Commcommander Server Inbound (MVS). The value here is the same value specified in the TCPPORT field of the DFCINIT control file. See the "Setup" section of the Commcommander Server Inbound (MVS) for more information.
	DFCUTL_COMMAND (pos. 1 - 14) — This control card describes the command the utility program is to perform. This field is shipped with the correct value (**TERMINATE** — pos. 15 - 30); therefore, no modifications are necessary at this time.
	ACF/VTAM **DFCUTL_TPN** (pos. 1 - 14) — This control card describes the protocol information needed to connect to the Commcommander Server you wish to terminate. The bold entries listed below, describe the keywords for this control card: • **VnRrMmYYMMDD** (pos. 15 - 26) — This is the version, release, modification, year, month, and day of the current Commcommander product. This field is shipped with the correct value, therefore, no modifications are necessary at this time. • **DFCCOMI** (pos. 31 - 38) — This is the name of the Commcommander Server you wish to terminate. This field is shipped with the correct value (DFCCOMI), therefore, no modifications are necessary at this time. • **lulululu** (pos. 39 - 46) — This is name of the logical unit used when you started Commcommander Server Inbound (MVS). The value here is the same value specified in the BASELU field of the DFCINIT control file. See the "Setup" section of the Commcommander Server Inbound for more information. • **modename** (pos. 47 - 54) — This is name of the log mode table used when you started Commcommander Server Inbound (MVS). The value here is the same value specified in the LOGMODE field of the DFCINIT control file. See the "Setup" section of the Commcommander Server Inbound for more information.
	DFCUTL_COMMAND (pos. 1 - 14) — This control card describes the command the utility program is to perform. This field is shipped with the correct value (**TERMINATE** — pos. 15 - 30); therefore, no modifications are necessary at this time.

Note It's recommended that you use the JCL members copied from the distribution tape. The examples in this manual are only for illustrative purposes.

To Update the TP Profile (for APPC/MVS versions)

Locate and edit the TPPROF JCL in the Commcommander MVS JCL library created during installation. The objective of the TPPROF job is to add a transaction program (TP) entry to the network's TP profile. This TP profile is a VSAM data set that is defined as a part of your IBM APPC/MVS installation. The TPPROF job invokes the APPC/MVS administration utility, ATBSDFMU, to make the profile entry. The addition is initiated by the TPADD command and is characterized by the command's associated parameters (e.g., TPNAME, ACTIVE). Refer to the IBM manual "Planning: APPC Management" (GC28-1110) for a detailed explanation of the TPADD command of the ATBSDFMU utility.

The actual entry is a JCL file that invokes a Docucorp transaction program named DFCCOMI. This program is responsible for receiving, interpreting, and responding to messages sent by Commcommander Server partner nodes.

The following TPPROF JCL illustration shows the JCL in its generic form, the form it has after being copied into a data set by the TAPEREAD job. The **bold** portions represent the portions you must change to meet your site's requirements. When the changes are complete, submit the job.

The job configures your APPC host environment to work with the newly-installed Commcommander programs.

```
//---------- JOB CARD HERE ----------
//*
//* ****************************************************************
//*  APPC/VTAM ADMINISTRATION UTILITY  - MVS/ESA  VTAM/APPC/CPIC  *
//*    - MAINTAIN TP PROFILE                                      *
//*    - MAINTAIN SIDE INFORMATION FILE                           *
//*                                                             *
//*****************************************************************
//STEP1  EXEC  PGM=IEFBR14,PARM='TYPRUN=RUN'
//SYSUT1   DD  DSN=isi.DFCCOM.VnnRnn.APPC.ERROR.MSG,
//         DD  DCB=(RECFM=FB,LRECL=133,BLKSIZE=1330),
//         DD  DISP=(NEW,CATLG,DELETE),
//         DD  SPACE=(TRK,(5,5))
//SYSUT2   DD  SYSOUT=*
//SYSPRINT DD  SYSOUT=*
//SYSIN    DD  DUMMY
//*
//STEP2  EXEC  PGM=ATBSDFMU,PARM='TYPRUN=RUN'
//SYSPRINT DD  SYSOUT=*
//SYSSDOUT DD  SYSOUT=*
//SYSSDLIB DD  DISP=SHR,DSN=ibm.APPC.TP.PROFILE
//* SYSSDLIB DD  DISP=SHR,DSN=ibm.APPC.SIDEINFO
//SYSIN    DD  DATA,DLM='/+'
  TPDELETE
     TPNAME(DFCCOMI)
  TPADD
     TPNAME(DFCCOMI)
     ACTIVE(YES)
     TPSCHED_DELIMITER(ENDDATA)
     TAILOR_SYSOUT(NO)
     TAILOR_ACCOUNT(NO)
KEEP_MESSAGE_LOG(ERROR)
     MESSAGE_DATA_SET(isi.DFCCOM.VnnRnn.APPC.ERROR.MSG)
     DATASET_STATUS(MOD)
     CLASS(SHORT)
     TPSCHED_TYPE(STANDARD)
     JCL_DELIMITER(ENDJCL)
//jobname JOB (account-info),'COMMCOMMANDER v.r',MSGLEVEL=1,
//             MSGCLASS=H,
//             CLASS=D,REGION=4M,
//             USER=uuuuuuuu,
//             PASSWORD=pppppppp
//*        *********************************************************
//*        ISI COMMUNICATIONS  -  COMMCOMMANDER SERVER/MVS -APPC/MVS
//*        *********************************************************
//*
//JOBLIB   DD   DISP=SHR,DSN=isi.DFCCOM.VnnRnn.LOADLIB
//*         DD   DISP=SHR,DSN=isi.product.VnnRnn.LOADLIB (for VLAM)
//DFCCOMI  EXEC PGM=DFCCOMI
//SYSUDUMP DD SYSOUT=H
//SYSPRINT DD SYSOUT=H
//DEBUG    DD SYSOUT=H
//ISIWTO   DD SYSOUT=H
//DFCMSG   DD SYSOUT=H,DCB=BLKSIZE=2048
//DFCLOG   DD SYSOUT=H,DCB=BLKSIZE=2048
//DFCTRACE DD SYSOUT=H,DCB=BLKSIZE=2048
//DFCINIT  DD DISP=SHR,DSN=isi.DFCCOM.VnnRnn.DFCCOMI.DFCINIT
```

Continued on next page

Continued from previous page

```
//* ------------------------------------------------------------------
//*
//*          The following file required for the VLAM Driver
//*
//* ------------------------------------------------------------------
//*VLMLIBNI DD DISP=SHR,DSN=isi.DFCCOM.VnnRnn.VLMLIBNI
//* ------------------------------------------------------------------
//*
//*          The following statements required for the  VRF Driver
//*          Upload to MVS
//*
//* ------------------------------------------------------------------
//*                                         DFCVRFI input control
//*DFCVRFCT DD DISP=SHR,DSN=isi.DFCCOM.VnnRnn.DFCVRFI.CTL
//*                                         DFCVRFI output VRF
//*DFCVRFOT DD DISP=SHR,DSN=isi.DFCCOM.VnnRnn.DFCVRFOT.VRF
//*                                         DFCVRFI output AUDIT
//*DFCVRFAU DD DISP=SHR,DSN=isi.DFCCOM.VnnRnn.DFCVRFI.AUDIT
//* ------------------------------------------------------------------
//*
//*          The following statements required for the JES Driver
//*
//* ------------------------------------------------------------------
//*
//*          DCB information should be the same for all JESO print
//*          files. Use the largest LRECL and BLKSIZE of all DD'S
//*
//*          OUTPUT statement NAME must be same as DDNAME
//*          Following will route to Metacode printer
//*
//*DFCJESM  OUTPUT CLASS=V,
//*DFCJESM  DD SYSOUT=(,),FREE=CLOSE,
//*            OUTPUT=*.DFCJESM,
//*            DCB=(RECFM=VBM,LRECL=8205,BLKSIZE=8209)
//*
//*          -----------------------------------------------------
//*          OUTPUT statement NAME must be same as DDNAME
//*          following will route to AFP printer
//*
//*DFCJESA  OUTPUT CLASS=W,FORMDEF=A10111,DATACK=BLOCK
//*DFCJESA  DD SYSOUT=(,),FREE=CLOSE,
//*            OUTPUT=*.DFCJESA,
//*            DCB=(RECFM=VBM,LRECL=8205,BLKSIZE=8209)
//*
//
ENDJCL
ENDDATA
/*
/+
//
```

Figure 22: TPPROF JCL

Perform the following steps when setting up the TP Profile:

1. Modify the following values:

Parameter	**Value**
Job Card	Replace with your MVS job card.
SYSUT1	The name of the data set to which Commcommander writes information and error messages.

Parameter	Value
SYSSDLIB	Replace with the data set name of the TP profile at your site.
MESSAGE_DATA_SET	The name of the data set to which Commcommander writes information and error messages.
jobname	The MVS Job Name used to identify the TP job to JES on the MVS system. **account-info —** Your MVS Job account information **USER —** Your MVS user identification **PASSWORD —** Your MVS user identification password
JOBLIB	Specify the Commcommander Load Library. If you're installing and using the Commcommander VLAM Driver, you'll concatenate the Load Library containing VLAM.
DFCINIT	The Commcommander Initialization File. For comprehensive information on the DFCINIT file, see To Customize the Inbound Communication Initialization File for MVS. **NOTE:** The following JCL statements are specific to the selected Commcommander Driver you plan to use. See the "Setup" section of each desired driver for additional JCL changes.
VLMLIBNI	The Commcommander VLAM Driver/MVS Settings File.
DFCVRFCT	The Commcommander VRF Driver/MVS Control File.
DFCVRFOT	The Commcommander VRF Driver/MVS Output File.
DFCVRFAU	The Commcommander VRF Driver/MVS Audit File.
DFCJESM	The Commcommander JES Driver/MVS Output Destination for Xerox print streams.
DFCJESA	The Commcommander JES Driver/MVS Output Destination for IBM AFP print streams.

2. Submit the job.
3. Check the job log for errors when the job is complete. If necessary, correct the errors and resubmit the job.

WARNING! The Commcommander VLAM Driver requires Docucorp's Virtual Library Access Method (VLAM), which is distributed with other Docucorp products (e.g., Documerge).

You must update the Commcommander/MVS Server JCL to include the Load Library of the product containing VLAM. This Loadlib should be concatenated below the Commcommander Server Joblib.

SETTING UP (INBOUND) COMMCOMMANDER INITIALIZATION SECTION ENTRIES FOR NT/UNIX

Tip If you've already set up the Initialization Entries, skip this section.

To Set Up (Inbound) Commcommander Initialization Section Entries for NT/UNIX

The keyword entries in the COMCMDR.INI file are used by Commcommander Server when establishing a session with a partner node. Commcommander Server uses these fields to determine the correct transportation protocol to use and transaction program to invoke.

- Edit the COMCMDR.INI file, located in the directory where the COMCMDRI executable is located.
- Create a section for the protocol information you will be using.
- Save your changes.

There are three sample sections included in the COMCMDR.INI created during the PC server installation. The first two samples are used with Commcommander Server Inbound (COMCMDRI.EXE) TCP/IP processing. The last sample is used for the Commcommander Server Outbound (COMCMDRO.EXE). The following figure is a sample of the COMCMDR.INI for use with Commcommander Server Inbound.

Note The dialog below is an example of the CCRCVPRT driver. Other inbound drivers will use a different name in the entries shown. Refer to the “Setup” section of the desired driver for values to enter in the parameters shown.

```
;        Commcommander INI
;        (a semicolon in position one is a comment line)
;-----------------------------------------------------------------
[CCRCVPRT]   Sample Commcommander Server Inbound parameters
;                            ----------------------------------------
;                            Gate name for Inbound Protocol (Gate/Port)
;                                 [default=same as section name above]
Gate=GATE0
;                            ----------------------------------------
;                            Invokable TP fully qualified executable
module
;                            name
;                                  (default = none)
Executable=c:\CCRCVPRT\CCRCVPRT.exe
;                            ----------------------------------------
;                            Invokable TP executable parameter
;                                 (default = none)
Parameter=c:\CCRCVPRT
;                            ----------------------------------------
;                            Temporary Directory - can be the same as
the
;                            executable
;                                 (default = current directory)
TempDir=
;                            ----------------------------------------
;                            Trace Level used for problem determination
;                                 (default = 0)
TraceLevel=0
```

Figure 23: Commcommander Server Inbound

The following list describes the keywords and their values, for use with Commcommander Inbound Server.

Option	Purpose
[ ]	The value within the [ ] is the section entry name. This value is case sensitive. Its use to group the following keyword values together under a single name, that may be referenced by the user during Commcommander startup. There may be multiple sections in a single COMCMDR.INI file. The name you chose for each section, is strictly up to you.
Gate	The symbolic name of the port or gate used for inbound communications. This value is case sensitive. If this field is omitted, the section name will be used in its place. The value of this field is used for the inbound connection from the client workstation. The following are the selected protocol descriptions for this field: **TCP/IP for Sockets** This value must match the port name entry in the SERVICES file.
Executable	The fully-qualified path and module name of the driver's executable module.
Parameter	The value of the parameters associated with the driver.
TempDir	Used by the driver as a working directory. The default value is **the current directory**.
TraceLevel	Trace Level number used for problem determination. If you are having protocol trouble, Docucorp may ask you to generate a trace log, which contains detailed information about the Commcommander API call sequence. Before calling Docucorp's Hotline, try running with a tracelevel of "**5**". This will be helpful information used by the Hotline support analyst when solving your problem.

Using (Inbound) Commcommander Server using TCPIP for NT/UNIX

There are two major components in Commcommander Server, based on the origination point of the conversation. The type of component used depends on where the conversation originates.

If the data is being moved from the mainframe to the PC, the conversation originates on the mainframe; therefore, the Commcommander Server Inbound (**COMCMDRI**™ software is a Docucorp product) is used to invoke the transaction program on the PC.

The following section describes how to start Commcommander Server Inbound.

To Start Commcommander Server Inbound - NT Service

1. Select **Start/Settings/Control Panel/Services** to display the System Services dialog box.
2. Select **Commcommander 213 Inbound**.
3. Select **Start** to start the server application.

To Start Commcommander Server Inbound - Non-NT Service

Note The Gate name was entered during installation. If you want to change the desired Gate name, perform the following steps:

1. Select the **Commcommander Inbound** icon located in the folder where you installed Commcommander.
2. Click the right mouse button to display the context menu and select **Properties**.
3. Add the required **Gate** parameter to the **Target** command. For more information, see the description of **Gate** in the following section's parameter explanations).
4. Select **OK**.
5. Proceed to the next step.

Double-click the **Commander Inbound** icon to start the process.

To Start Commcommander Server Inbound - UNIX

Change to the Commcommander Server directory from a command prompt and enter the following command to start the Commcommander Server Inbound (comcmdri executable):

```
comcmdri  Gate  [TraceLevel]
```

The Gate parameter is mandatory. The parameter values within the [ ] are optional. If present, these values will override their associated values in the COMCMDR.INI file (the parameters on the command line are positional).

Parameter	Explanation
Gate	The value of this field is used for the inbound connection from the client workstation. This field is mandatory. The value of this field is case sensitive, and should match the section name in the COMCMDR.INI file, where the Commcommander Server Inbound parameters are defined. Commcommander Server Inbound can be started using this value as its only command line parameter (preferred method). Values for the remaining parameters will be taken from the COMCMDR.INI file. If the specified value does not match an entry in the COMCMDR.INI file, all of the Commcommander Server Inbound parameters must be supplied on the command line. In the case stated above, all protocol specific rules apply to the specified value. If you are using TCP/IP Sockets, this must be the name of the port address defined in your services file.
TraceLevel	Trace Level number used for problem determination. If you are having protocol trouble, Docucorp may ask you to generate a trace log, which contains detailed information about the Commcommander API call sequence. Before calling Docucorp's Hotline, try running with a tracelevel of "**5**". This will be helpful information used by the Hotline support analyst when solving your problem.

Commcommander will display "Ready" when it is successfully running.

To Stop Commcommander Server Inbound

Press **Ctrl+C** on the screen where Commcommander Server Inbound is running to end the process.

To Review Messages

Browse any and all Commcommander error messages by viewing the COMCMDR.LOG file, which is located in the same directory as the COMCMDR executable file. The COMCMDR.LOG file contains any protocol errors or trace messages produced during a Commcommander session. The messages are written chronologically from top to bottom. Therefore to view the most recent messages go to the bottom of the file and scroll backwards.

USING (INBOUND) COMMCOMMANDER SERVER USING SNA FOR NT/AIX

To Start Commcommander Server Inbound

There are two major components in Commcommander Server, based on the origination point of the conversation. The type of component used depends on where the conversation originates.

If the data is being moved from the mainframe to the PC, the conversation originates on the mainframe; therefore, the SNA Server software is used to invoke the transaction program on the PC.

The following section describes how to start Commcommander Server Inbound.

1. Ensure that SNA Server software is running on your server machine.

2. If you're operating in an NT environment, ensure that the **TPSTART** program is running. **TPSTART** is located in the directory where Commcommander executable modules are installed (**c:\comcmdr\tpstart.exe**). You can start this program by double clicking of the TP Start icon located in "ISI Utilities" program group created during the Commcommander installation procedure.

To Stop Commcommander Server Inbound

By stopping the SNA Server software, invokable TPs will be disabled.

To Review Messages

Browse any and all Commcommander error messages by viewing the COMCMDR.LOG file, which is located in the same directory as the COMCMDR executable file. The COMCMDR.LOG file contains any protocol errors or trace messages produced during a Commcommander session. The messages are written chronologically from top to bottom. Therefore to view the most recent messages go to the bottom of the file and scroll backwards.

USING (INBOUND) COMMCOMMANDER SERVER USING TCPIP/SNA FOR MVS

WARNING! The following instructions only pertain to a TCP/IP or ACF/VTAM environment. If you're operating in an APPC/MVS environment, please see To Operate Commcommander in an APPC/MVS Environment.

Starting Commcommander in a TCP/IP or ACF/VTAM Environment

WARNING! The dataset where you installed the Commcommander program components **must** be the **first** dataset referenced in the STARTUP JCL's **JOBLIB** or **STEPLIB** statement. Multi-tasking TCP/IP **won't** work properly if this isn't the case.

In a TCP/IP or ACF/VTAM environment, one of the host components of Commcommander (the Inbound Message Handler) must be manually started. Once the Inbound Message Handler is running as a started task, the PC and mainframe components can communicate and pass data. The started task remains in memory until it's shut down via the TERMINATE job. Terminating this started task frees machine resources that may be required by other jobs.

In a TCP/IP or ACF/VTAM environment, Commcommander cycles, checking for requests from user workstations or mainframe jobs to be processed. All requests are processed, and then Commcommander waits for a designated period before awakening to check for more requests to process.

To Start Commcommander in a TCP/IP or ACF/VTAM Environment

Submit the STARTUP JCL.

This JCL can be found in the STARTUP member of the JCL library created as part of the installation process.

To Stop Commcommander in a TCP/IP or ACF/VTAM Environment

Run the DFCUTL communications utility using the TERMINATE command to stop or bring down the mainframe Inbound Message Handler.

This JCL can be found in the TERMINAT member of the JCL library created as part of the installation process.

To Operate Commcommander in an APPC/MVS Environment

Starting and stopping Commcommander Server/MVS in an APPC/MVS environment is very easy — simply open an APPC/MVS session.

APPC runs as a started task, which invokes a transaction scheduler (ASCH) that invokes the job specified within the Transaction Program Profile (TPPROF) library. Therefore, there is no job, as such, to start or to stop. For more information, see . The VTAM installation and maintenance requirements for APPC and LU 6.2 include the setup for this kind of task handling. See your communications network administrator or system programmer for more information.

Chapter 5

Commcommander JES Driver

Overview

The Commcommander JES Driver is a Docucorp software product that allows PC users to route print files from the PC to mainframe Metacode, AFP, or line printers. These can be any mainframe JES-attached printers for which the PC print data stream has been specifically generated.

The JES driver can be directed to pull print data streams from either a Docucorp Queue Systems queue, from a specified directory, or both. These specifications reside in a JES Driver initialization file, which also provides a delay timer. When there are no print data streams to transfer, the JES Driver waits for the specified time period, then re-checks the queue, thus reducing the system resource requirements on the PC.

Once the JES Driver is started, it runs continuously, polling a Queue Systems queue or a directory. When an input file enters the system, the JES Driver sends it to the mainframe JES queue then, removes it from its associated Queue Systems queue or directory.

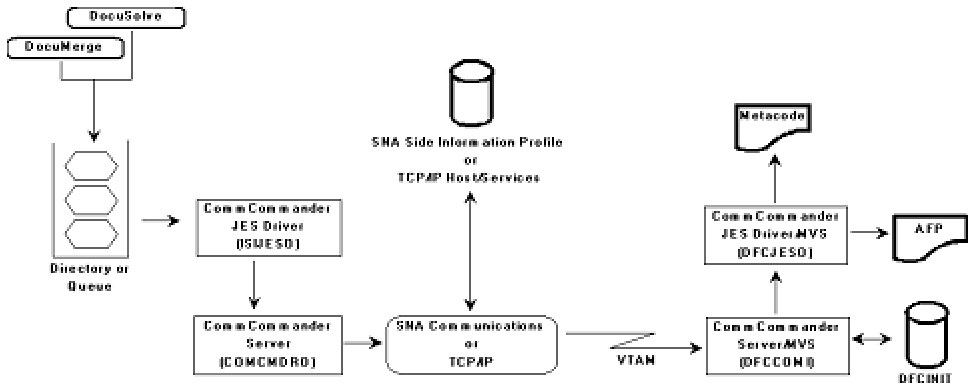

DocuSolve
DocuMerge
Directory or Queue
CommCommander JES Driver (ISWESO)
CommCommander Server (COMCMDRD)
SNA Side Information Profile or TCP/IP Host/Services
SNA Communications or TCP/IP
VTAM
CommCommander Server.MVS (DFCCOMI)
DFCINIT
CommCommander JES Driver.MVS (DFCJESO)
Metacode
AFP

Figure 24: Commcommander JES Driver

SETTING UP COMMCOMMANDER JES DRIVER

SETTING UP COMMCOMMANDER JES DRIVER/NT/ UNIX

To Set Up Commcommander JES Driver

Commcommander JES Driver (ISIJESO) periodically polls a designated directory or queue for input. When an input file is placed in the directory or queue, ISIJESO automatically removes it and sends it to the mainframe JES queue, where it can be printed on an IBM AFP, Xerox Metacode, or line printers.

If you plan to process files from a directory, you must create the directory and specify its name in the appropriate .INI file before executing the JES Drive. Likewise, if you plan to process files from a Docucorp Queue Systems queue, you must create and name the queue, and specify it in the .INI file before executing the JES Driver.

Note Prior to setting up Commcommander JES Driver, you must set up Commcommander Server. For information about setting up Commcommander Server Outbound, see Commcommander Server Outbound.

If you are using TCPIP and you are plan to run this driver from a workstation other than your server machine (where Commcommander Server Outbound is running), you must setup the following:

1. The IP address of the server machine where Commcommander Server Outbound is running. For information on how to set the required values, see To Define the Host IP Address.
2. The Service port address assigned to the Commcommander Server Outbound. For information on how to set the required values, see To Define the Service Port Number.

If you are using Docucorp's Queue system, you will need a services entry for the inbound queue connection, which must be the same port address that the queue daemon is using.

To Set Up the Commcommander JES Driver INI File

Because ISIJESO operates automatically, you must set the options before starting and running the application.

1. Determine the input source of the print data stream to be uploaded and set the "InputSource=" parameter accordingly.
2. Make the necessary changes to the remaining parameters in the ISIJESO.INI file.

Note Remember to remove the semicolon (;) from Column 1 to activate the selected parameters.

To set the options, edit the ISIJESO.INI file.

```
[ISIJESUpload]
;       ISIJESO.INI
;       (a semicolon in position one is a comment line)
;       ------------------------------------------------------------
;                       Select an InputSource type (default F)
;                            F = read input from File
;                            Q = read input from Queue
;                            B = read input from either File or Queue
InputSource=F
;                       Parameters to read from File (default jesup)
;                            JES Input directory
Path=jesup
;                       File Name Pattern to be uploaded (default *.* (all
                        files))
FileName=*.*
;                       Delete JES files after uploading? (default N)
;                            Y = yes
;                            N = no
DeleteFile=N
;                       Prompt user to enter a confirmation for file deletion,
;                       if Delete=Y (default Y)
Prompt=Y
;                       Parameters to read from Queue (default none)
;                         The name of your server machine
;QueueServer=
;                       The name given when the Queue Daemon was started
;                       (default jes)
;QueueName=jes
;                       Queue Type (default SOCKET)
;                            NAMEDPIPE (NetBEUI)
;                            SOCKET    (TCP/IP)
;QueueType=SOCKET
;                       Parameter for transport communications (default SOCKET)
;                            NAMEDPIPE (NetBEUI)
;                            SOCKET    (TCP/IP)
CommCommanderProtocol=SOCKET
;                       Name of CommCmdr server machine
;                      (default none)
CommCommanderServer=
;                       Gate Name when CommCmdr was started
;                       (default none)
CommCommanderGate=GATE0
;                       Number of connection retries (default 5)
CommCommanderConnections=5
;                       MVS DDName of destination JES (must be caps)
;                       (default DFCJESM)
;                            DFCJESA    for AFP
;                            DFCJESM    for META
JESDD=DFCJESM
;                       Temporary Directory (default current directory)
TempDir=
;                       Delay Time in seconds between inactivity (default 10)
DelaySec=10
;                       Trace Level, used for problem determination (default 0)
TraceLevel=0
```

Figure 25: The ISIJESO.INI File

Figure 26:

Note The term “current directory” refers to where the program's executable (.exe) module resides.

If a parameter is labeled as having no default value, that parameter is required and you must specify a setting.

Field	Value
InputSource	**(Required)** This specifies the source of the input print file to ISIJESO. The valid values are: • **F** = read the input from a file • **Q** = read the input from a Docucorp Queue Systems queue • **B** = read the input from either a file or a Docucorp Queue Systems queue The default value is **F**.
Path	**(Valid for InputSource = F only)** The drive and directory name where the input files will be searched. The default value is the current directory, which is where the JES Driver .exe module resides (e.g., **C:\ISIJES** for Windows NT). If you specify a name only, it will be treated as a sub-directory to the current directory. If you want to specify a directory name from the root, you must proceed the name you specify with a back slash “\” (you can precede the directory name with a drive letter). For example, if you want the JES Driver to search a sub-directory named “jesup,” you would code “Path=jesup” which will direct the JES Driver to search for input files in **C:\ISIJES\JESUP**. If you want the JES Driver to search a directory named “jesup” off the root directory of a network drive, you might code the Path parameter as **R:\JESUP**. Remember that you must have already created the directory or sub-directory before running the JES Driver. The default value is **jesup**.
FileName	**(Valid for InputSource = F only)** This directs the JES Driver to either select a specific file from the input directory, or you can specify a pattern using an asterisk “*” to specify a group of files to be selected. • *myfile.dat* — selects only this specific file • **.dat* — selects all files with a file name extension of .dat • *ab*.** — selects all files that begin with the letters “AB” The default value is ***.***.
DeleteFile	**(Valid for InputSource = F only)** This parameter directs the JES Driver to delete the input file from the specified upload directory. If you want the file to be deleted after it has been transferred, code a **Y;** if not, code an **N**. If you choose not to delete the file, the JES Driver creates a subdirectory called *complete* and moves the file there (i.e., the file is copied into the *complete* subdirectory, then deleted from the upload directory). See the note following Using Commcommander JES Driver. The default value is **N**.
Prompt	(**Valid only when Delete=Y is specified**) This parameter controls the prompting of the **DeleteFile=Yes** or **=No** question during execution. Specifying **Prompt=N** skips the **DeleteFile=Yes** or **=No** question during execution, resulting in a seemless operation.
QueueServer (Required)	**(Required if InputSource = Q or B)** The name of the server machine where the Docucorp queue is running. (Check with your LAN Administrator for the network server name.) If you're running the Commcommander JES Driver on the same machine as the Docucorp queue management program (ISIQMSD.EXE), you **must** leave this parameter blank (i.e., code the parameter “**QueueServer=**”). There is no default value.
QueueName	**(Required if InputSource = Q or B)** The name of the Docucorp queue containing the input file. The default value is **jes**.
QueueType	**(Required if InputSource = Q or B)** The type of Docucorp queue which contains the input file. The default value is **SOCKET**.
Commcommander Protocol (Required)	This value must match the input protocol used when the Commcommander Server Outbound was started. Valid values are “SOCKET” and “NAMEDPIPE”. The default value is **SOCKET**.

Field	Value
Commcommander Server (Required)	The name of the server machine where your Commcommander server is running. (Check with your LAN Administrator for the network server name.) If you're running the Commcommander JES Driver on the same machine as the Commcommander Server, you **must** leave this parameter blank (e.g., code the parameter as "**CommCommanderServer=**"). There is no default value.
Commcommander Gate (Required)	The name of the server gate used for Commcommander communication. There is no default value.
Commcommander Connections	The number of times ISIJESO should attempt to connect to Commcommander. The default value is **5**.
JESDD	The DDName of the JCL statement in the mainframe Commcommander Server/MVS job. The default value is **DFCJESM**.
TempDir	The directory where any temporary processing files reside; it can be the same as the .exe directory. For more information about the appropriate entry, see the Path= explanation above. The default value is **the current directory**.
DelaySec	The number of seconds that ISIJESO will sleep during periods of inactivity. If there are no input file or queue to process, ISIJESO will wait for this number of seconds then check to see if there is any input files to transfer. The valid range of values is 0 to 3600. If zero is specified, this will cause ISIJESO to run continuously with no delay. 3600 seconds is equal to one hour max. The default value is **10**.
TraceLevel	Trace Level number used for problem determination. If you are having protocol trouble, Docucorp may ask you, to generate a trace log, which contains detailed information about the Commcommander API call sequence. Before calling Docucorp's Hotline, try running with a tracelevel of "**5**". This will be helpful information used by the Hotline support analyst when solving your problem.

SETTING UP COMMCOMMANDER JES DRIVER/MVS

To Set Up Commcommander JES Driver/MVS

At this time, you should have already installed Commcommander JES Driver/MVS. The installation of the Commcommander JES Driver/MVS consists of loading the program(s) from a tape to the load library. For more information on installing the JES Driver and the specific steps involved, please see Installing MVS Server Components.

After you've installed the Commcommander JES Driver/MVS, you should add the following JCL statements to your Commcommander STARTUP job or the TPPROF JCL:

```
//* -----------------------------------------------------------------
//*
//*        The following statements are required for the JES Driver
//*
//* -----------------------------------------------------------------
//*
//*           DCB information should be the same for all JESO print
//*           files. Use the largest LRECL and BLKSIZE of all DD'S
//*
//*           OUTPUT statement NAME must be same as DDNAME
//*           Following will route to Metacode printer
//*
//DFCJESM  OUTPUT CLASS=V,
//DFCJESM  DD SYSOUT=(,),FREE=CLOSE,
//            OUTPUT=*.DFCJESM,
//            DCB=(RECFM=VBM,LRECL=8205,BLKSIZE=8209)
//*
//*           -----------------------------------------------------
//*           OUTPUT statement NAME must be same as DDNAME
//*           following will route to AFP printer
//*
//DFCJESA  OUTPUT CLASS=W,FORMDEF=A10111,DATACK=BLOCK
//DFCJESA  DD SYSOUT=(,),FREE=CLOSE,
//            OUTPUT=*.DFCJESA,
//            DCB=(RECFM=VBM,LRECL=8205,BLKSIZE=8209)
//*
```

Figure 27: Commcommander STARTUP JCL

You can find examples of the above JCL in the STARTUP or TPPROF member. The DD statements are initially commented out; you should “un-comment” them when you install a new driver.

Parameter	Description
DFCJESM	Used to route print streams to a Xerox Metacode printer.
DFCJESA	Used to route print streams to an IBM AFP printer.

There must be one pair of DD and OUTPUT statements (for each unique set of print attributes) for each printer to which you want the JES Driver to route print data streams. The exact number of statements depends on your specific requirements.

In the following example, you will need five unique pairs of DD and OUTPUT statements.

- Metacode printer for Class H
- Metacode printer for Class V
- AFP printer for Class A with a unique FORMDEF
- AFP printer for Class A with a unique FORMDEF
- AFP printer for Class B with a unique FORMDEF

Note The above pairs of DD and OUTPUT statements are only meant as an example.

The DD name you specify here has to be used in the Commcommander JES Driver .INI central file to properly route the print data stream to the requested output printer.

Using Commcommander JES Driver

Note If, while the JES Driver is running, you manually delete a file that has already been sent (and you've selected the input directory and file option (InputSource=F)), this action causes the JES Driver NOT to recognize any other files in the directory. As a result, no other files will be transferred. The solution is to re-start the JES Driver and be sure not to delete any files when the JES Driver is running.

If you select InputSource=F and indicate that the JES Driver shouldn't delete the input files (DeleteFile=N), the JES Driver creates a subdirectory called *complete* and moves the files (i.e., the files are copied into the *complete* subdirectory, then deleted from the upload directory).

When the JES Driver is configured with InputSource=F and DeleteFile=N, you should always use unique file names since the files are accumulated in the *complete* subdirectory. If, for example, the JES Driver is running and has already moved a file called *myfile1.net*, and then you copy another file into the directory with the same name, the second file overwrites the first and the original file content in the *complete* subdirectory is lost.

Using Commcommander JES Driver/NT/UNIX

To Start Commcommander JES Driver — NT Service

1. Select **Start/Settings/Control Panel/Services** to display the System Services dialog box.
2. Select **JES Outbound**.
3. Select **Start** to start the server application.

To Start Commcommander JES Driver — Non-NT Service

1. Ensure that Commcommander Server Outbound is up and running with the correct gate name. If you are using the SNA protocol, you must also activate the associated SNA Communications software on the server machine.
2. Make sure the Commcommander JES Driver ISIJESO.INI file is pointing to the proper Commcommander Server gate name.
3. If you plan to use a Docucorp Queue Systems queue as input to the JES Driver, you should start the queue by executing **ISIQMSD** as a separate session/ window. This queue daemon may be running on the same or separate server machine.
4. If you're operating in a UNIX environment, skip to the explanation after the "**-or-**" listed below.
   - Double-click on the Commcommander JES Upload Driver icon located in the "ISI Utilities" program group.

     -or-

     From a command prompt, change to the **ISIJES** directory and enter **isijeso** on the command line to start Commcommander JES Driver. This program reads the ISIJESO.INI file to obtain its processing parameters.

     Once you've started ISIJESO, it processes automatically by polling the job directory or queue, removing input files, and sending them to the mainframe JES queue. If you're using an input file directory, the files are either deleted or moved to the *complete* subdirectory, depending on the "DeleteFile=" parameter. If you're using an input queue, the files are automatically deleted from the queue after a successful upload to the host.
5. Start your Documerge application. Make sure that the output files are written into the ISIJESO input directory or queue that is being polled. Additional print files may be added to the directory or queue using standard or other Docucorp utilities. Remember that any one directory or queue must contain the same types of printstream files (e.g., AFP, Metacode, or Line data).

To Stop Commcommander JES Driver

Press **Ctrl+C** on the screen where Commcommander JES Driver is running to end the process. You should stop Commcommander JES Driver before stopping the SNA Server. Additional print files may be added to the directory or queue using standard utilities or other Docucorp utilities. Remember that any one directory or queue must contain the same type of print stream files (e.g., Meta, AFP, or Line).

To Review Messages

Browse any and all Commcommander error messages by viewing the ISIJESO.LOG file, which is located in the same directory as the ISIJESO executable file. The ISIJESO.LOG file contains any protocol errors or trace messages produced during a Commcommander session. The messages are written chronologically from top to bottom. Therefore to view the most recent messages go to the bottom of the file and scroll backwards.

USING COMMCOMMANDER JES DRIVER/MVS

The Commcommander JES Driver/MVS is initiated automatically by the Commcommander Server/MVS and the initiation of a conversation from the partner PC node.

Ensure the appropriate Commcommander Server/MVS is active. For more information, see Using (Inbound) Commcommander Server using TCPIP/SNA for MVS.

Chapter 6

Commcommander VRF Driver

Overview

The Commcommander VRF Drivers are Docucorp software products that allow the transfer of a Docucorp Variable Replacement File (VRF) from the mainframe to the PC, and vice versa. The VRF is normally created by the Documerge Variable Data Reformatter (VDR) program, which is used as input to the Documerge processing and print generation. Also, an error VRF can be created from the Documerge print engine. The obvious advantage of transferring the VRF is the flexibility to create a file on one platform for processing on another.

Although the transfer process is conceptually similar, in that you can download a VRF from the mainframe to the PC or upload a VRF from the PC to the mainframe, the individual steps involved are quite different. In transferring files to the other platform, the recipient of the VRF exerts the most control over the process. When you're downloading a file, for example, the PC Driver can be customized to override certain program parameters specified in the host Driver, and vice versa during an upload procedure.

The Commcommander VRF Driver includes a productivity tool called the *Audit File*. This tool gathers the value(s) stored in the tags of each Merge Set in the VRF.

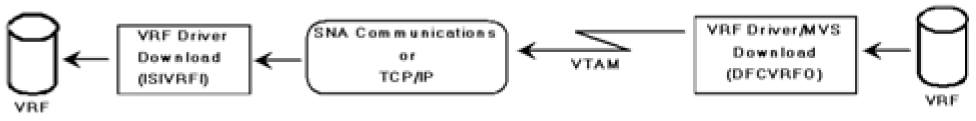


Figure 28: Commcommander VRF Driver — Downloading Overview

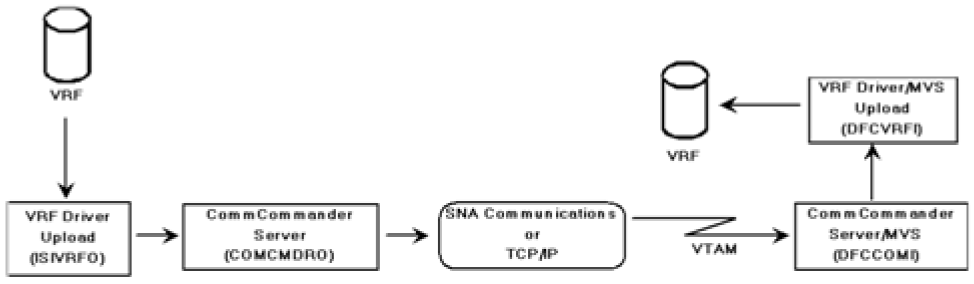


Figure 29: Commcommander VRF Driver — Uploading Overview

UPLOADING A VRF

UPLOADING A VRF (PC – MAINFRAME)

Commcommander VRF Driver/NT/UNIX is a server product which constantly processes jobs as soon as the jobs are available. The server operates as a continuously-running session in Windows NT or UNIX environments.

Commcommander VRF Driver/MVS can only be directed to store the VRF into an MVS data set. The data set name is specified from the PC Driver or can be overridden by the VRF Driver Initialization file. The Commcommander VRF Driver includes a productivity tool called the *Audit File*. This tool gathers the value(s) stored in the tags of each Merge Set in the VRF. For more information, see The Audit File.

Commcommander VRF Driver/MVS is a transaction program that is invoked from Commcommander Server/MVS. The Commcommander VRF Driver is started differently depending on the communications protocol you are using.

TCPIP or **ACF/VTAM**
When the Commcommander VRF Driver is executed, it automatically connects to the Commcommander Inbound Server/MVS (which is already running as a started task). The Commcommander Inbound Server/MVS, in turn, invokes the Commcommander VRF Driver/MVS.

APPC/MVS
When the Commcommander VRF Driver is executed, it invokes the Commcommander Inbound Server Transaction Profile (TP). The Commcommander Inbound Server TP, in turn, invokes the Commcommander VRF Driver/MVS.

The following diagram illustrates the uploading procedure, from the PC to the mainframe.

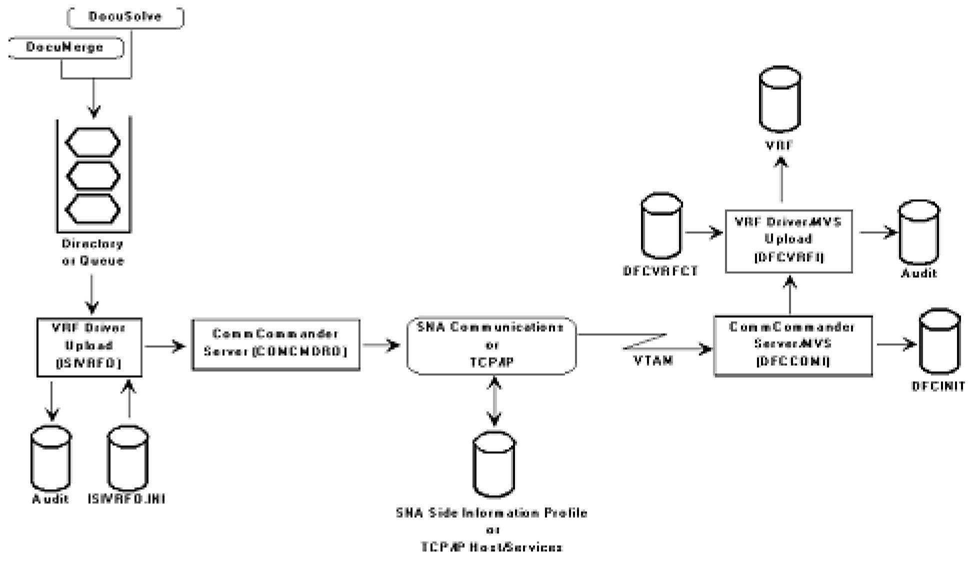

DocuSolve
DocuMerge
Directory or Queue
VRF Driver Upload [ISVRFO]
Audit
ISVRFO.INI
CommCommander Server [CONCMDRO]
SNA Communications or TCP/IP
SNA Side Information Profile or TCP/IP Host/Services
VTAM
CommCommander Server/MVS [DFCCONI]
DFCINIT
VRF Driver/MVS Upload [DFCVRFI]
DFCVRFCT
VRF
Audit

Figure 30: Uploading a VRF from the PC to the mainframe

Setting Up Commcommander VRF Driver (Upload)

To Set Up Commcommander VRF Driver

Commcommander VRF Driver (ISIVRFO) periodically polls a designated queue or directory for input. When an input file is placed in the queue or directory, ISIVRFO automatically removes it and sends it to the mainframe.

If you plan to process files from a directory, you must create the directory and specify its name in the appropriate .INI file before executing the VRF Driver. Likewise, if you plan to process files from a Docucorp Queue Systems queue, you must create and name the queue, and specify it in the .INI file before executing the VRF Driver.

Note Prior to setting up Commcommander VRF Driver, you must set up Commcommander Server. For information about setting up Commcommander Server Outbound, see Commcommander Server Outbound.

To Set Up the Commcommander VRF Driver INI File

1. Determine the input source of the VRF to be uploaded and set the “InputSource=” parameter accordingly.
2. Make the necessary changes to the remaining parameters in the ISIVRFO.INI file.

Note Remember to remove the semicolon (;) from Column 1 to activate the selected parameters.

To set these options, edit the ISIVRFO.INI file, located in the directory where the ISIVRFO executable resides.

```
[ISIVRFUpload]
;       ISIVRFO.INI
;       (a semicolon in position one is a comment line)
;                       Select an InputSource type (default = F)
;                         F = Read VRF from a File
;                         Q = Read VRF from a Queue
;                         B = Read VRF from Both a File and Queue
InputSource=F
;                       ----------------------------------------------------
;                        Parameters to read a VRF from a File (default vrfup)
;                        VRF Input Path
Path=vrfup
;                        File Name pattern to be selected
;                        (default = *.* (all files))
FileName=*.*
;                        Delete VRF data files after uploading? (default N)
;                          Y = Yes
;                          N = No
DeleteFile=N
;                       Prompt user to enter a confirmation for file deletion,
;                       if Delete=Y (default Y)
Prompt=Y
;                       ---------------------------------------------------
;                        Parameters to read a VRF from a Queue (default none)
;                         The name of your server machine
;QueueServer=
;                        The name given when the Queue Daemon was started
;                        (default vrf)
;QueueName=vrf
```

Continued on next page

```
;                        Queue Type  (default = SOCKET)
;                          NAMEPIPE        (NetBEUI)
;                          SOCKET          (TCP/IP)
;QueueType=SOCKET
;                       ---------------------------------------------------
;                       Parameter for transport communications (default
NAMEDPIPE)
;                       The name of your server machines input protocol
;                          NAMEPIPE        (NetBEUI)
;                          SOCKET          (TCP/IP)
CommCommanderProtocol=SOCKET
;                        Name of CommCmdr server machine (case sensitive)
;                        (default none)
CommCommanderServer=
;                        Gate Name when CommCmdr was started (case sensitive)
;                        (default none)
CommCommanderGate=GATE0
;                        Number of connection retries (default 5)
CommCommanderConnections=5
;                       --------------------------------------------------
;                        AuditMode = Yes or No (default N)
;                          Y = Yes
;                          N = No
AuditMode=N
;                        Your Tag Name (default = ***ALL***)
;AuditTag=***ALL***
;                        Path name where audit file is written
;                        (default current directory)
;AuditPath=
;                        Your output audit file name (default vrfup.adt)
;AuditFile=vrfup.adt
;                        --------------------------------------------------
;                        MVS DDName of destination VRF (must be caps)
;                        (default DFCVRFOT)
VRFDD=DFCVRFOT
;                        Temporary Directory (default current directory)
TempDir=
;                        Delay Time in seconds between inactivity (default 10)
DelaySec=10
;                      Set to zero, used for problem determination (default 0)
TraceLevel=0
```

Figure 31: Sample Upload ISIVRFO.INI File

Field	Value
InputSource	**(Required)** This specifies the source of the input file to ISIVRFO. The valid values are: **B** = Both file and Docucorp Queue Systems input **F** = File input only **Q** = Docucorp Queue Systems input only The default value is **F**.

Field	Value
Path	**(Valid for InputSource = F only)** The drive and directory name where the input files will be searched. The default value is the current directory, which is where the VRF Driver .exe module resides (e.g., **C:\ISIVRFO**). If you specify a name only, it will be treated as a sub-directory to the current directory. If you want to specify a directory name from the root, you must proceed the name you specify with a slash “/” (you can precede the directory name with a drive letter). For example, if you want the VRF Driver to search a sub-directory named “vrfup,” you would code “Path=vrfup” which will direct the VRF Driver to search for input files in **C:\ISIVRFO\VRFUP**. If you want the VRF Driver to search a directory named “vrfup” off the root directory of a network drive, you might code the Path parameter as **R:\VRFUP**. Remember that you must have already created the directory or sub-directory before running the VRF Driver. The default value is **vrfup**.
FileName	**(Valid for InputSource = F only)** This directs the VRF Driver to either select a specific file from the input directory, or you can specify a pattern using an asterisk “*” to specify a group of files to be selected. myfile.dat — selects only this specific file *.dat — selects all files with a file name extension of .dat ab*.* — selects all files that begin with the letters “AB” The default value is ***.***.
DeleteFile	**(Valid for InputSource = F only)** This parameter directs the VRF Driver to delete the input file from the specified upload directory. If you want the file to be deleted after it has been transferred, code a **Y;** if not, code an **N**. If you choose not to delete the file, the VRF Driver creates a subdirectory called *complete* and moves the file there (i.e., the file is copied into the *complete* subdirectory, then deleted from the upload directory). See the note following Using Commcommander VRF Driver/ NT/UNIX (Upload). The default value is **N**.
Prompt	(**Valid only when Delete=Y is specified**) This parameter controls the prompting of the **DeleteFile=Yes** or **=No** question during execution. Specifying **Prompt=N** skips the **DeleteFile=Yes** or **=No** question during execution, resulting in a seemless operation.

Field	Value
QueueServer (Required)	**(Required if InputSource = Q or B)** The name of the server machine where the Docucorp queue is running. (Check with your LAN Administrator for the network server name.) If you're running the Commcommander VRF Driver on the same machine as the Docucorp queue management program (ISIQMSD.EXE), you **must** leave this parameter blank (e.g., code the parameter as "**QueueServer=**"). There is no default value.
QueueName	**(Required, if InputSource = Q or B)** If the VRF is routed to a queue, this specifies the name of the Docucorp input queue in which the VRF file resides. The default value is **vrf**.
QueueType	**(Required, if InputSource = Q or B)** The type of Docucorp queue to which the VRF file is written. The default value is **SOCKET**.
CommCommanderProtoc ol (Required)	This value must match the input protocol used when the comcmdr outbound server was started. Valid values are "SOCKET" and "NAMEDPIPE". The default value is **SOCKET**.
Commcommander Server (Required)	The name of the server machine where your Commcommander server is running. (Check with your LAN Administrator for the network server name.) If you're running the Commcommander VRF Driver on the same machine as the Commcommander Server, you **must** leave this parameter blank (e.g., code the parameter as "**CommCommanderServer=**"). There is no default value.
Commcommander Gate (Required)	The name of the server gate used for Commcommander communication. There is no default value.
Commcommander Connections	The number of times ISIVRFO should attempt to connect to Commcommander. The default value is **5**.
AuditMode	Answer **Yes** if you want to generate an audit file for a given tag name; otherwise, answer **No**. For more information on the audit file, see The Audit File. The default value is **N**.
AuditTag	Enter the tag name for which you want to generate an audit file. You can use the special wild card tag name *****ALL***** to receive a report on all tags. The default value is *****ALL*****.
AuditPath	Enter the path name where you want to store the audit file. For more information about the appropriate entry, see the Path= explanation above. The default value is **the current directory**.
AuditFile	Enter the file name you want assigned to the audit file. The default value is **vrfup.adt**.

Field	Value
VRFDD	The DDName of the JCL statement in the mainframe Commcommander Server/MVS job where the VRF is to be written. This parameter can be overridden with the mainframe parameter. The default value is **DFCVRFOT**.
TempDir	The directory where temporary work files, encountered during normal processing, are stored. For convenience, you can specify the same directory or sub-directory where you installed the VRF executable modules. For more information about the appropriate entry, see the Path= explanation above. The default value is **the current directory**.
DelaySec	The number of seconds that ISIVRFO will sleep during periods of inactivity. If there are no input files or queues to process, ISIVRFO will wait for this number of seconds, then check to see if there are any input files to transfer. The valid range of values is between 0 and 3600 seconds (up to one hour). If zero is specified, this will cause ISIVRFO to run continuously, with no delay. The default value is **10**.
TraceLevel	This is for internal Docucorp problem determination and should be zero for normal processing. The default value is **0**.

To Set Up Commcommander VRF Driver/MVS

At this time, you should have already installed Commcommander VRF Driver/MVS. The installation of the Commcommander VRF Driver/MVS consists of loading the program(s) from a tape to the load library. For more information on installing the VRF Driver and the specific steps involved, please see Installing MVS Server Components.

Within the JCL members copied during the TAPEREAD job is the DFCVRFCI member. DFCVRFCI creates the VRF Upload Control File dataset used by the VRF Upload application DFCVRFI.

1. Modify the JCL to meet your site's requirements. The **bold** portions of the JCL below represent the site-specific parts.

```
//---------- JOB CARD HERE ----------
//*          ********************************************************
//*          DFCVRFI CONTROL FILE ALLOCATE AND LOAD
//*          ********************************************************
//*
//*          --------------------------------------------------------
//*                 VRF DRIVER CONTROL FILE FOR DFCVRFI
//*          --------------------------------------------------------
//*
//STEP1  EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSIN    DD DUMMY
//*                                           DFCVRFI input control
//SYSUT2   DD DSN=isi.DFCCOM.VnnRnn.DFCVRFI.CTL,
//            DISP=(NEW,CATLG),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=80),
//            UNIT=SYSDA,
//            SPACE=(TRK,(1,2))
//SYSUT1   DD *

*****              VRF Upload control entries go here
*****
***** You will set these values after the creation of the VRF Upload
***** datasets
*****
*****                         See step 4 below.

/*
//*
//*                                           DFCVRFI output VRF
//STEP2  EXEC PGM=IEFBR14
//SYSUT1   DD DSN=isi.DFCCOM.VnnRnn.DFCVRFOT.VRF,
//         DD DCB=(RECFM=U,BLKSIZE=24576),
//         DD DISP=(NEW,CATLG,DELETE),
//         DD SPACE=(TRK,(2,1))
//SYSUT2   DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSIN    DD DUMMY
//*
//*                                           DFCVRFI output AUDIT
//STEP3  EXEC PGM=IEFBR14
//SYSUT1   DD DSN=isi.DFCCOM.VnnRnn.DFCVRFI.AUDIT,
//         DD DCB=(RECFM=V,BLKSIZE=256),
//         DD DISP=(NEW,CATLG,DELETE),
//         DD SPACE=(TRK,(2,1))
//SYSUT2   DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSIN    DD DUMMY
//*
//
```

Figure 32: DFCVRFCI JCL

2. Submit the job.
3. Check the job log for errors when the job is complete. If necessary, correct the errors and resubmit the job.

4. Edit the DFCVRFCT dataset **(isi.DFCCOM.VnnRnn.DFCVRFI.CTL)** created in Steps 1 through 3, stated above. Using the keyword definitions listed below ensure the desired values are set.

Note The following figure is an example of the DFCVRFCT entries. This example represents the exact keywords needed for VRF Upload.

```
************************ TOP OF DATA ********************
* CVRF-KEYWORDVALUE-----------------
* ..+....1....+....2....+....3....+....4....+....5....+..
DFCVRF_DDN    ddname
DFCVRF_AUD_DDNddname
* CVRF_TAG    tagname
DFCVRF_TAG    ***ALL***
********************** BOTTOM OF DATA ********************
```

Figure 33: DFCVRFCT Control File

Note Records beginning with an asterisk (*) are comments. In the above example, the first two records are comment records. Each of the non-comment records begins with DFCVRF in columns 1 through 6, followed by the unique identifier of the control record. The parameters are position-sensitive, in that each field must begin in the specified position.

Field	Value
DFCVRF_DDN	The JCL ddname of the VRF file you want to create.
DFCVRF_AUD_DDN	(Optional) The name of the output Audit file you want to create.
DFCVRF_TAG	(Optional) The tag name about which you want to generate an output Audit file.

5. Add the following JCL statements to your Commcommander STARTUP job or the TPPROF JCL. You can find examples of STARTUP and TPPROF in the JCL members copied during the TAPEREAD job during installation. The DD statements are initially commented out; you should un-comment them when you install a new driver.

```
//* -----------------------------------------------------------------
//*
//*           The following statements are required for the
//*                   VRF Driver Upload to MVS
//*
//* -----------------------------------------------------------------
//*                                         DFCVRFO input control
//*DFCVRFCT DD DISP=SHR,DSN=isi.DFCCOM.VnnRnn.DFCVRFI.CTL
//*                                         DFCVRFI output VRF
//*DFCVRFOT DD DISP=SHR,DSN=isi.DFCCOM.VnnRnn.DFCVRFOT.VRF
//*                                         DFCVRFI output AUDIT
//*DFCVRFAU DD DISP=SHR,DSN=isi.DFCCOM.VnnRnn.DFCVRFI.AUDIT
//* -----------------------------------------------------------------
//*
```

Figure 34: Commcommander STARTUP JCL

Statement	Description

DFCVRFCT	The DFCVRFCT statement points to the location of your control file. This is the same dataset in Step 1.
DFCVRFOT	You should include one output JCL statement per VRF that points to the location of your output file. This is the same dataset in Step 1.
DFCVRFAU	(Optional) If coded, you must supply a JCL statement that points to the location of your output audit file. This is the same dataset in Step 1.

Note You can store VRFs to more than one MVS output dataset. You should add a set of the above DD statements for each separate output dataset you plan to utilize. The outbound VRF driver(s) started on the partner PC machine, can select an on of these MVS datasets to store the output VRF.

USING COMMCOMMANDER VRF DRIVER/NT/UNIX (UPLOAD)

Note If, while the VRF Driver is running, you manually delete a file that has already been sent (and you've selected the input directory and file option (InputSource=F)), this action causes the VRF Driver NOT to recognize any other files in the directory. As a result, no other files will be transferred. The solution is to re-start the VRF Driver and be sure not to delete any files when the VRF Driver is running.

If you select InputSource=F and indicate that the VRF Driver shouldn't delete the input files (DeleteFile=N), the VRF Driver creates a subdirectory called *complete* and moves the files (i.e., the files are copied into the *complete* subdirectory, then deleted from the upload directory).

When the VRF Driver is configured with InputSource=F and DeleteFile=N, you should always use unique file names since the files are accumulated in the *complete* subdirectory. If, for example, the VRF Driver is running and has already moved a file called *myfile1.net*, and then you copy another file into the directory with the same name, the second file overwrites the first and the original file content in the *complete* subdirectory is lost.

To Start Commcommander VRF Driver — NT Service

1. Select **Start/Settings/Control Panel/Services** to display the System Services dialog box.
2. Select **VRF Outbound**.
3. Select **Start** to start the server application.

To Start Commcommander VRF Driver — Non-NT Service

1. Ensure that Commcommander Server Outbound is up and running with the correct gate name. If you are using the SNA protocol, you must also activate the associated SNA Communications software on the server machine.
2. Make sure the Commcommander VRF Driver ISIVRFO.INI file is pointing to the proper Commcommander Server gate name.

3. If you plan to use a Docucorp Queue Systems queue as input to the VRF Driver, you should start the queue by executing **ISIQMSD** as a separate session/ window. This queue daemon may be running on the same or separate server machine.

4. If you're operating in a UNIX environment skip to the explanation after the "**-or-**" listed below.

   - Double-click on the Commcommander VRF Upload Driver icon located in the "ISI Utilities" program group.

     -or-

     From a command prompt, change to the **ISIVRF** directory and enter **isivrfo** on the command line to start Commcommander VRF Driver. This program reads the ISIVRFO.INI file to obtain its processing parameters.

     Once you've started ISIVRFO, it processes automatically by polling the job directory or queue, reading input files, sending them to the mainframe data set, and removing input files. If you're using an input file directory, the files are either deleted or moved to the *complete* subdirectory, depending on the "DeleteFile=" parameter. If you're using an input queue, the files are automatically deleted from the queue after a successful upload to the host.

WARNING! The input files being uploaded must have unique names. If you want to upload another file with the same name as a previous one, you must stop and then re-start the VRF Driver.

5. Start your Documerge application. Make sure that the output files are written into the ISIVRFO directory or queue.

To Stop Commcommander VRF Driver

Press **Ctrl+C** on the screen where Commcommander VRF Driver is running to end the process. You should stop Commcommander VRF Driver before stopping the SNA Server.

To Review Messages

Browse any and all Commcommander error messages by viewing the ISIVRFO.LOG file, which is located in the same directory as the ISIVRFO executable file. The ISIVRFO.LOG file contains any protocol errors or trace messages produced during a Commcommander session. The messages are written chronologically from top to bottom. Therefore to view the most recent messages go to the bottom of the file and scroll backwards.

To Use Commcommander VRF Driver/MVS

The Commcommander VRF Driver/MVS is invoked automatically by the Commcommander Server/MVS and the initiation of a conversation from the partner PC node.

Ensure the appropriate Commcommander Server/MVS is active. For more information, see Using (Inbound) Commcommander Server using TCPIP/SNA for MVS.

DOWNLOADING A VRF

DOWNLOADING A VRF (MAINFRAME – PC)

Commcommander VRF Driver/MVS is a batch program that can either be executed in-line as an additional job step with the Documerge jobs, or in a separate job to transfer the mainframe VRF to the PC. This VRF can then be processed with other Docucorp products, such as Documerge or DocuSolve.

Commcommander VRF Driver/NT/UNIX can be directed to store the VRF into either a Docucorp Queue Systems queue or a file. The file destination is specified in the VRF Driver Initialization (**.ini**) file with the queue name or path/file name. The Commcommander VRF Driver includes a productivity tool called the *Audit File*. This tool gathers the value(s) stored in the tags of each Merge Set in the VRF. For more information, see The Audit File.

The Commcommander VRF Driver is started differently depending on the communications protocol you are using.

TCPIP
For TCPIP, the Commcommander Server (comcmdri) inbound message handler must be up and running. This will start a PC session that connects and communicates with the VRF Driver on the Host MVS machine.

SNA
For SNA, the VRF Driver/NT/AIX is started from Microsoft's or IBM's SNA services communication products, Advanced Program-to-Program Communications (APPC). When the Commcommander VRF Driver/MVS program is executed, it automatically initiates a conversation with the PC through APPC, which automatically starts a PC session and initiates the PC transaction program. The PC transaction program, in turn, executes Commcommander VRF Driver/NT/AIX.

The following diagram illustrates the downloading procedure, from the mainframe to the PC.

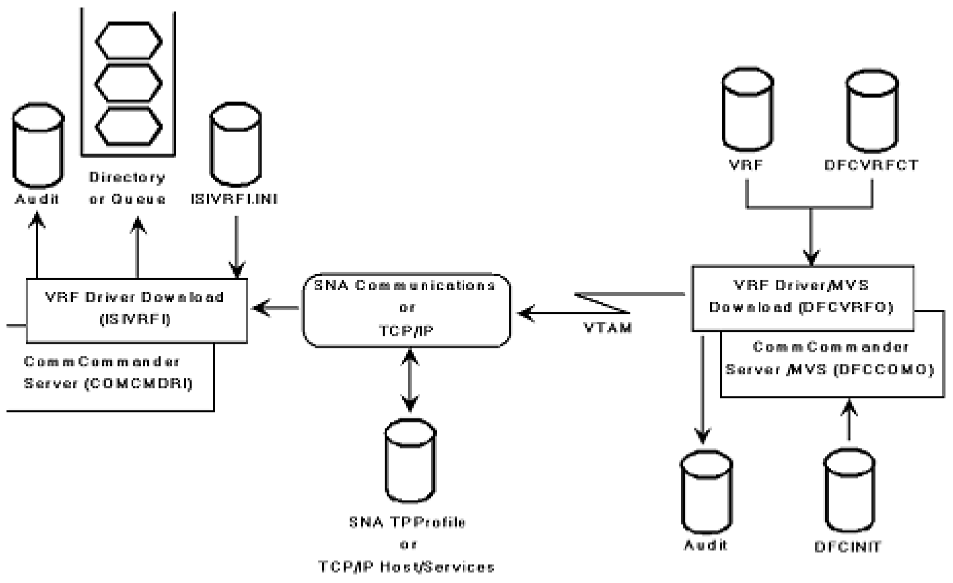

Audit
Directory or Queue
ISIVRFI.INI
VRF Driver Download (ISIVRFI)
CommCommander Server (COMCMDRI)
SNA Communications or TCP/IP
SNA TPProfile or TCP/IP Host/Services
VTAM
VRF
DFCVRFCT
VRF Driver/MVS Download (DFCVRFO)
CommCommander Server /MVS (DFCCOMO)
Audit
DFCINIT

Figure 35: Downloading a VRF from the mainframe to the PC

Setting Up Commcommander VRF Driver (Download)

To Set Up Commcommander VRF Driver/NT

Commcommander VRF Driver (ISIVRFI) is automatically invoked whenever a request is received from the partner node. ISIVRFI is considered a transaction program (TP). A TP must be predefined, via transaction profile, before invocation.

Defining Transaction Profiles and Initialization Entries

TCP/IP

- **NT** — For information on defining transaction program profiles, see Setting Up (Inbound) Commcommander Initialization Section Entries for NT/UNIX.
- **UNIX** — For information on defining transaction program profiles, see Setting Up (Inbound) Commcommander Initialization Section Entries for NT/UNIX.
- **MVS** — For information on the Communication Initialization File, see To Customize the Outbound Communication Initialization File for MVS.

SNA

- **NT** — For information on defining transaction program profiles, see Defining Transaction Programs.

  Use the following values when defining the VRF Driver TP:

  TP Name:**ISIVRFI**
  Command line:**C:\ISIVRF\ISIVRFI.EXE C:\ISIVRFI** (this parameter specifies the location of the VRF Driver .INI file).

- **AIX** — For information on defining transaction program profiles, see Defining Transaction Programs.

  Use the following values when defining the VRF Driver TP:

  Profile name:**ISIVRFI**
  Command line:**/u/isivrf**
  TPN:**ISIVRFI**
  Full Path:**/u/isivrf/isivrfi**
  Standard output/**u/isivrf**
  Standard error/**u/isivrf**

- **MVS** — For information on the Communication Initialization File, see To Customize the Outbound Communication Initialization File for MVS.

To Set Up the Commcommander VRF Driver INI File

Because ISIVRFI operates automatically, you must set the options before the transaction profile is invoked.

1. Determine whether the VRF will be written to a file or a Docucorp Queue Systems queue, and set the "OutputMode=" parameter accordingly.
2. Determine if the outbound message handler, Commcommander VRF Driver/ MVS will specify the file or queue name to which the VRF will be written. If the Commcommander VRF Driver is to override the MVS specification, then change the appropriate parameters in the ISIVRFI.INI file.

   To set these options, edit the appropriate ".INI" file.

Note Remember to remove the semicolon (;) from Column 1 to activate the selected parameters.

```
[ISIVRFDownload]
;       ISIVRFI.INI
;       (a semicolon in position one is a comment line)
;                     ---------------------------------------------
;                     Select an OutputMode type (default F)
;                       F = Write VRF to a File
;                       Q = Put VRF in Queue
OutputMode=F
;                     ---------------------------------------------
;                     Parameters for writing a VRF to a File (default vrfdn)
;                       Output directory where VRF will be written
Path=vrfdn
;                     Output VRF file name (default vrf.dat)
FileName=vrf.dat
;                     Write Option (default R)
;                       N = new file only or
;                       R = replace existing file (or created new)
WriteOption=R
;                     ---------------------------------------------
;                     Parameters for writing a VRF to a queue (default none)
;                       The name of your server machine
;QueueServer=
;                     Name given when the Queue Daemon started (default vrf)
;QueueName=vrf
;                     Queue Type (default SOCKET)
;                          NAMEDPIPE    (NetBEUI)
;                          SOCKET       (TCP/IP)
;QueueType=SOCKET
;                     ---------------------------------------------
;                     Transaction Program Name (SNA only) (default ISIVRFI)
;                       The name of the TP used to invoke the VRF driver
;                          ISIVRFI
TPName=ISIVRFI
;                     ---------------------------------------------
;                     Parameter for transport communications (default SOCKET)
;                       The name of your server machines input protocol
;                          SOCKET   (TCP/IP)
;                          CPIC     (SNA)
InboundProtocol=SOCKET
;                     ---------------------------------------------
;                     AuditMode (default N)
;                          Y = Yes
;                          N = No
AuditMode=N
;                     Your Tag Name (default ***ALL***)
;AuditTag=***ALL***
;                     Your output audit file name (default vrfnd.adt)
;AuditFile=vrfdn.adt
;                     Path name where audit file is written
;                     default current directory)
;AuditPath=
;                     ---------------------------------------------
;                     Optional (default none; see PCMerge Documentation)
;                       Submit VRF parameters writes VRF to a Queue
;                       including the proper PRODDEF information
;ExitApplication=submit
;                     Product definition name (default none)
;                       contains the proddef information
;ExitAppParameters=
;                     Temporary Directory, can be same as .exe
;                     (default current directory)
;                     ----------------------------------------------
TempDir=
;                     Trace Level, used for problem determination (default 0)
TraceLevel=0
```

Field	Value
OutputMode	**(Required)** Specifies the destination of the output VRF file. **F** = create the VRF in a file in the directory specified by the partner node control record or by the Path setting below. **Q** = put the VRF in a Docucorp queue name specified in the QueueName setting below. The default value is **F**.
Path	**(Valid for OutputMode = F only)** The drive and directory name where the VRF will be created. The default value is the current directory, which is where the VRF Driver .exe module resides (e.g., **C:\ISIVRF**). If you specify a name only, it will be treated as a sub-directory to the current directory. If you want to specify a directory name from the root, you must proceed the name you specify with a back slash “\” (you can precede the directory name with a drive letter). For example, if you want the VRF Driver to create a sub-directory named “vrfdn,” you would code “Path=vrfdn” which will direct the VRF Driver to create a file in **C:\ISIVRFI\VRFDN**. If you want the VRF Driver to create a directory named “vrfdn” off the root directory of a network drive, you might code the Path parameter as **R:\VRFDN**. Remember that you must have already created the directory or sub-directory before running the VRF Driver. The default value is **vrfdn**.
FileName	If the VRF is routed to a file, this parameter specifies the file name to which the VRF is written. This parameter overrides any name supplied from the partner node. If no file name is specified here and none is supplied from the partner node, the default value is used. The default value is **vrf.dat**.
WriteOption	If the VRF is routed to a file, this parameter specifies whether a new file is created, or an existing file replaced: **N** = create a new file. If the file name already exists, the VRF Driver will not overwrite, but will issue an error message and issue a Return Code 16, Reason Code 4 back to the partner. **R** = replace the file if it already exists. If the file doesn't exist, a new file will be created. The default value is **R**.
QueueServer (Required)	**(Required if OutputMode = Q)** The name of the server machine where the Docucorp queue is running. (Check with your LAN Administrator for the network server name.) If you're running the Commcommander VRF Driver on the same machine as the Docucorp queue management program (ISIQMSD.EXE), you **must** leave this parameter blank (e.g., code the parameter as “**QueueServer=**”). There is no default value.
QueueName	**(Required, if OutputMode = Q)** If the VRF is routed to a queue, this specifies the name of the Docucorp queue in which the VRF file is written. This parameter overrides any name supplied from the partner node. The default value is **vrf**.
QueueType	**(Required, if OutputMode = Q)** The type of Docucorp queue to which the VRF file is written. The default value is **SOCKET**.
TPName (Windows NT SNA only)	This value is the name of the TP defined in Microsoft SNA Server Symbolic Destination used to invoke the VRF Inbound driver. The default value is **ISIVRFI**.

Field	Value
InboundProtocol	This value must match the protocol used when the Commcommander Server Inbound was started. The valid values are **CPIC** for SNA or **SOCKET** for TCP/IP. The default value is **SOCKET**.
AuditMode	Enter **Y** if you want to generate an audit file for a given tag name; otherwise, enter **N**. For more information on the audit file, see The Audit File. The default value is **N**.
AuditTag	Enter the tag name for which you want to generate an audit file. You can use the special wild card tag name ***ALL*** to receive a report on all tags. The default value is ***ALL***.
AuditFile	Enter the file name you want assigned to the audit file. The default value is **vrfdn.adt**.
AuditPath	Enter the path name where you want to store the audit file. The default value is **the current directory**.
ExitApplication	Enter **SUBMITVRF** to place the VRF in a queue with product definition information. See the PC Documerge documentation for more information.
ExitAppParameters	Enter the name of the product definition file containing the PC Documerge PRODDEF information. See the PC Documerge documentation for more information.
TempDir	The directory where temporary work files, encountered during normal processing, are stored. For convenience, you can specify the same directory or sub-directory where you installed the VRF executable modules. For more information about the appropriate entry, see the Path= explanation above. The default value is **the current directory**.

To Set Up the Commcommander VRF Driver/MVS

At this time, you should have already installed Commcommander VRF Driver/MVS. The installation of the Commcommander VRF Driver/MVS consists of loading the program(s) from a tape to the load library. For more information on installing the VRF Driver and the specific steps involved, please see Installing MVS Server Components.

Within the JCL members copied during the TAPEREAD job are the DFCVRFCO and DFCVRFO member. DFCVRFCO creates the VRF Download Control File dataset used by the VRF Download application DFCVRFO.

1. Modify the DFCVRFCO JCL to meet your site's requirements. The **bold** portions of the JCL below represent the site-specific parts.

```
//---------- JOB CARD HERE ----------
//*          ********************************************************
//*          DFCVRFO CONTROL FILE ALLOCATE AND LOAD
//*          ********************************************************
//*
//*          --------------------------------------------------------
//*                 VRF DRIVER CONTROL FILE FOR DFCVRFO
//*          --------------------------------------------------------
//*
//STEP1  EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSIN    DD DUMMY
//*                                                DFCVRFO output control
//SYSUT2   DD DSN=isi.DFCCOM.VnnRnn.DFCVRFO.CTL,
//            DISP=(NEW,CATLG),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=80),
//            UNIT=SYSDA,
//            SPACE=(TRK,(1,2))
//SYSUT1   DD *

*****               VRF Download control entries go here
*****
***** You will set these values after the creation of the VRF Download datasets
*****
*****                          See step 4 below.

/*
//*
//*                                                DFCVRFI output AUDIT
//STEP3  EXEC PGM=IEFBR14
//SYSUT1   DD DSN=isi.DFCCOM.VnnRnn.DFCVRFO.AUDIT,
//         DD DCB=(RECFM=V,BLKSIZE=256),
//         DD DISP=(NEW,CATLG,DELETE),
//         DD SPACE=(TRK,(2,1))
//SYSUT2   DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSIN    DD DUMMY
//*
//
```

Figure 36: DFCVRFCT JCL

2. Submit the job.
3. Check the job log for errors when the job is complete. If necessary, correct the errors and resubmit the job.
4. Edit the DFCVRFCT dataset **(isi.DFCCOM.VnnRnn.DFCVRFO.CTL)** created in Steps 1 through 3, stated above. Using the keyword definitions listed below ensure the desired values are set.

Note The following figure is an example of the DFCVRFCT entries. This example represents the exact keywords needed for VRF Download.

You need to choose which protocol to be used and select or setup the correct control card. The VRF Control File is shipped with the SNA control card commented out. If you will be using the SNA protocol, un-comment the SNA control card and comment out or omit the TCP/IP control cards.

```
************************ TOP OF DATA ********************
* ------------TCP/IP Example--------------------------------
* DFCVRF_ID---V R M YYMMDD----TPNAME--HOSTNM--PORTNM -
* ..+....1....+....2....+....3....+....4....+....5....+....6....+....
DFCVRF_TPN    020001991202    ISIVRFI hostnameportnum
*
* ------------SNA    Example---remember to uncomment and shift left 2
* DFCVRF_ID---V R M YYMMDD----TPNAME--PARTLU--LOGMODE-
* ..+....1....+....2....+....3....+....4....+....5....+....6....+....
* DFCVRF_TPN    020001991202    ISIVRFI lulululumodename
*
* ---------------------------------------------------------------
* KEYWORD ----CONTROL PARAMETERS ------------------------------
* ---------------------------------------------------------------
* CVRF-KEYWORDVALUE------------------
* ..+....1....+....2....+....3....+....4....+....5....+....6....+....
DFCVRF_DDN    ddname
DFCVRF_PATH
DFCVRF_FILE   vrf.dat
DFCVRF_AUD_DDNDFCVRFAU
* CVRF_TAG    tagname
DFCVRF_TAG    ***ALL***
********************** BOTTOM OF DATA ********************
```

Figure 37: DFCVRFCT Control File

Note Records beginning with an asterisk (*) are comments. In the above example, the first two records are comment records. Each of the non-comment records begins with DFCVRF in columns 1 through 6, followed by the unique identifier of the control record. The parameters are position-sensitive, in that each field must begin in the specified position.

Field	Value
DFCVRF_TPN	This identifies the Transaction Program Name control record.
TPNAME	The Transaction Program Name that has been created on the PC partner node. This can be any name you want. Docucorp recommends this to be ISIVRFI.
HOSTNM	**(TCP/IP)** This is the symbolic name of the partner machine. Your network administrator defines this name in the TCP/IP MVS Host file.
PORTNM	**(TCP/IP)** This is the port number being used by the partner node that identifies Commcommander to the host.
PARTLU	**(SNA)** The APPC or LU name used by Commcommander. This is the symbolic name by which Commcommander on the partner node is known in the SNA network. Your network administrator defines this name with an APPC macro in the VTAM Application Program Major node table. You can use ISISP**xx** where **xx** is a number and value (e.g. **ISISP24**).
LOGMODE	**(SNA)** The Mode table name used to establish the SNA session between the two partner nodes. Your network administrator defines this table and its name in the VTAM network definitions. This same Mode table must be defined on the partner PC node.
DFCVRF_DDN	The ddname of the VRF file you want to create.
DFCVRF_PATH	The fully qualified path name containing the drive, directory and/or sub-directory names where the output VRF is to be written on the partner PC node. For Windows NT, you must supply the drive letter (e.g., **C:\**). This path name can be overridden by the Commcommander VRF Driver PC initialization file parameters. If a path name is not supplied from this control record (or from an overriding parameter), the defaults will be **C:\vrf** (for Windows NT) or **./vrf** (for UNIX).

Field	Value
DFCVRF_FILE	The file name or the Docucorp Queue Systems queue name to which the output VRF will be written on the partner PC node. This file/queue name can be overridden by the Commcommander VRF Driver PC initialization file parameters. If the file/queue name is not supplied from this control record (or from an overriding parameter), the default will be **vrf.dat**.
DFCVRF_AUD_DDN	(Optional) The name of the output Audit file you want to create.
DFCVRF_TAG	(Optional) The tag name about which you want to generate an output Audit file.

5. Modify the DFCVRFO JCL to meet your site's requirements. The **bold** portions of the JCL below represent the site-specific parts.

```
//---------- JOB CARD HERE ----------
//* *******************************************************************
//*  Commcommander VRF Driver/MVS  Download to PC
//*
//*  DFCVRFO - COMMCOMMANDER VRF DRIVER/MVS
//*            TRANSFER VRF FROM MAINFRAME TO PC
//*
//* *******************************************************************
//JOBLIB   DD DISP=SHR,DSN=isi.DFCCOM.V02R00.LOADLIB
//DFCVRFO  EXEC PGM=DFCVRFO,REGION=4M
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//ISIWTO   DD SYSOUT=*
//ISIWTL   DD SYSOUT=*
//DFCINIT  DD DSN=isi.DFCCOM.V02R00.DFCCOMO.DFCINIT,
//            DISP=SHR
//DFCMSG   DD SYSOUT=*,DCB=BLKSIZE=2048
//DFCLOG   DD SYSOUT=*,DCB=BLKSIZE=2048
//DFCTRACE DD SYSOUT=*,DCB=BLKSIZE=2048
//*
//*                      YOUR INPUT VRF FILE
//DFCVRFIN DD DISP=SHR,DSN=isi.your.VRF
//*
//*                      PREALLOCATE THE OUTPUT AUDIT FILE
//DFCVRFAU DD DISP=SHR,DSN=isi.DFCCOM.V02R00.DFCVRFOT.AUDIT
//DFCVRFCT DD DSN=isi.DFCCOM.V02R00.DFCVRFO.CTL
//
//
```

Figure 38: DFCVRFO JCL

Parameter	Value
Job Card	Replace with your MVS/ESA job card.
JOBLIB	Specify the Commcommander Load Library.
DFCINIT	The Commcommander Initialization File. For comprehensive information on the DFCINIT file, see To Customize the Outbound Communication Initialization File for MVS.
DFCVRFIN	The data set name of your VRF input file.
DFCVRFAU	The data set name of your VRF Audit file.
DFCVRFCT	The data set name of you VRF Download Control file. For more information, see previous Step 4.

USING COMMCOMMANDER VRF DRIVER (DOWNLOAD)

To Use Commcommander VRF Driver/NT/UNIX

1. (SNA/AIX) Ensure that SNA Server communication software is up and running.

   -or-

   (TCP/IP) Ensure that Docucorp's Commcommander Server Inbound program is up and running.

2. Ensure the proper settings are specified in the Commcommander VRF Driver/NT ISIVRFI.INI file.

3. Ensure the directory named in the ISIVRFI.INI file already exists if you are directing the VRF to a file. If you are directing the VRF to a Docucorp Queue Systems queue, make sure the queue program **ISIQMSD** is up and running.

   When using the SNA protocol, the VRF transaction program is automatically invoked by Microsoft's SNA Server communications software whenever the partner node's VRF Driver starts the conversation with this local node VRF Driver. For TCP/IP, the TP is invoked by Docucorp's Commcommander Inbound Server.

   Multiple Transaction Program Profiles can be defined in the SNA Server, with each pointing to different directories containing individual copies of Commcommander VRF Driver and its ISIVRFI.INI file. One example in which you might want to use multiple profiles is to have one TPN set up to write to a file, and another to put the VRF in a Docucorp Queue Systems queue. By selecting the TPN name in the partner node control cards, you can select the type of destination to which the VRF will be routed.

To Review Messages

Browse any and all Commcommander error messages by viewing the ISIVRFI.LOG file, which is located in the same directory as the ISIVRFI executable file. The ISIVRFI.LOG file contains any protocol errors or trace messages produced during a Commcommander session. The messages are written chronologically from top to bottom. Therefore to view the most recent messages go to the bottom of the file and scroll backwards.

To Use Commcommander VRF Driver/MVS

The Commcommander VRF Driver/MVS initiates the conversation with the partner PC node. This is a program that can be executed as a job step within an existing job, or as a separate job. You have the option of transferring the VRF dataset to the PC in the same job that creates the VRF or, by allocating the VRF dataset as a permanent file you can transfer it at anytime using a separate job.

Note It's recommended that you use the JCL members copied from the distribution tape. The examples in this manual are for illustrative purposes.

To Start Commcommander VRF Driver Download (PC to Mainframe)

Within the JCL members copied during the TAPEREAD job is the DFCVRFO member.

1. Modify, if necessary, any DFCVRFCT control cards. For more information, see To Set Up the Commcommander VRF Driver/MVS.
2. After making any changes to the control cards, submit the DFCVRFO JCL to MVS for processing.

The Audit File

The Commcommander VRF Driver includes a productivity tool called the *Audit File*. This tool gathers the value(s) stored in one or all tags of each Merge Set in the VRF.

The audit file is labeled as a productivity tool because you can utilize its information to establish and manage the amount of time necessary to complete specific tasks in your production environment. You can use the results of a report based on the audit file to spot problem areas in your work flow and institute new time-saving strategies.

You can use your own in-house reporting software to turn the audit file into an effective management report that highlights areas needing improvement. The file is written as a standard MVS data set and is accessible with any text editor.

Working with the Audit File

In each of the Commcommander VRF Drivers, you first specify whether you want to produce an audit file, then include the tag name for which you want to collect the value. By including your preferences in the control cards for each driver, you can quickly receive a report on the desired tag and its value as dictated by the VRF. You can create up to four audit files — one each for the PC and mainframe drivers on both the upload and download procedures.

The driver accepts the name of a single tag name (e.g., Member_Number), or a wild card value with which you can obtain results on **all** tag names (e.g., ***ALL***). As an added feature, you can edit the VDR and create user-defined tags and group tags. (You can create group tags in the same manner as creating a Documerge Sort Key.)

The following examples illustrate controls cards requesting an audit file.

```
* DFCVRF_ID---
* ------------INPUT PARAMETERS ---------------------------------
* ..+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8
DFCVRF_DDN    DFCVRFOT
DFCVRF_AUD_DDNDFCVRFAU
* CVRF_TAG    NAME
DFCVRF_TAG    ***ALL***
*
```

Figure 39: MVS Control card creating an Audit File

```
; VRF.INI at c:\VRF
   .
   .
;AuditMode=N
;AuditMode=Y
;AuditTag=***ALL***
;AuditTag=POLICY.NUMBER
;AuditFile=XXXXXXX1.DAT
;AuditPath=..\vrfuadt
```

Figure 40: PC Control card creating an Audit File

After you've executed the processing run, a look at the audit file results in the following screen display:

```
02/02/97   16:52:07   ***ALL***                    ISIVRFO AUDIT FILE BEGIN.
02/02/97   16:52:07   ***ALL***                    OUTPUT VRF DDN=DFCVRFOT
02/02/97   16:52:07   ***ALL***                    PARTNER TARGET PATH NAME=
02/02/97   16:52:07   ***ALL***                    PARTNER TARGET FILE NAME=
02/02/97   16:52:07   ***ALL***                    REQUESTED TAG NAME=***ALL***
02/02/97   16:52:07   NAME                         Brandon M. Williams
02/02/97   16:52:07   AGE                          27
02/02/97   16:52:07   SEX                          Male
  .
  .
  .
02/02/97   16:52:07   ***************              END OF MERGE SET
02/02/97   16:52:07   NAME                         Rachel F. Scott
02/02/97   16:52:07   AGE                          37
02/02/97   16:52:07   SEX                          Female
  .
  .
  .
02/02/97   16:52:07   ***************              END OF MERGE SET
02/02/97   16:52:07   NAME                         Bryan B. Masters
02/02/97   16:52:07   AGE                          55
02/02/97   16:52:07   SEX                          Male
  .
  .
  .
02/02/97   16:52:07   ***************              END OF MERGE SET
02/02/97   16:52:07   ***************              END OF FILE VRF DDN=DFCVRFOT
```

Figure 41: The Audit File

The format of the audit file is as follows:

Field	Value
Columns 1-2	contain the date and time stamps of the file. The date and time reflects the original creation of the file: it doesn't change as the file is updated.
Column 3	contains the tag name(s) on which the report was generated.
Column 4	contains the tag data.

The top portion of the audit file contains identification records, including the DD name of the VRF input file, the requested tag name, and other data from the control cards. After these identification records, the file lists out the tag name and tag data for all requested records.

Chapter 7

Commcommander Imagecreate Driver

OVERVIEW

The Commcommander Imagecreate Driver sends a data stream from MVS, across your network, to your PC. The Imagecreate Driver contains two components, CCSNDPRT and CCRCVPRT that run on a MVS Host and PC server machine, respectfully.

Note **CCSNDPRT and CCRCVPRT** replace the DFCIMCO and ISIIMCI utility from the Commcommander 2.0 version.

The data transfer is handled either by TCP/IP or IBM's Advanced Program-to-Program Communications (APPC). In an APPC context, the Imagecreate Driver and CCRCVPRT are transaction programs using LU 6.2 to handle data and message transfers.

The driver ensures that the mainframe-based Printing Resources needed in the upcoming data stream-to-image conversions are available. If not available, the required Printing Resources are downloaded at this time.

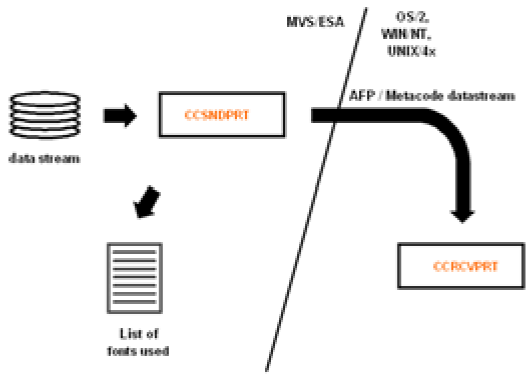


Figure 1: Imagecreate Driver Processing

Downloading a Data Stream (Mainframe – PC)

Commcommander Imagecreate Driver/MVS is a batch program that downloads an image data stream, one document at a time, to be converted by transferring data from a mainframe running MVS to a server running Windows NT or RS6000 UNIX. Separate JCL is provided for Xerox Metacode and IBM AFP data streams.

Commcommander Imagecreate Driver/NT/UNIX can be directed to store the Data Stream into either a Docucorp Queue Systems queue or a file (e.g., LocalDisk). The file destination is specified in the Imagecreate Driver Initialization (**.ini**) file with the QueueType keyword.

The Commcommander Imagecreate Driver is started differently depending on the communications protocol you are using.

TCPIP

For TCPIP, the Commcommander Server (comcmdri) inbound message handler must be up and running. This will start a PC session that connects and communicates with the Imagecreate Driver on the Host MVS machine.

SNA

For SNA, the Imagecreate Driver/NT/UNIX is started from Microsoft's or IBM's SNA services communication products, Advanced Program-to-Program Communications (APPC). When the Commcommander Imagecreate Driver/MVS program is executed, it automatically initiates a conversation with the PC through APPC, which automatically starts a PC session and initiates the PC transaction program. The PC transaction program, in turn, executes Commcommander Imagecreate Driver/NT/ UNIX.

The following diagram illustrates the downloading procedure, from the mainframe to the PC.

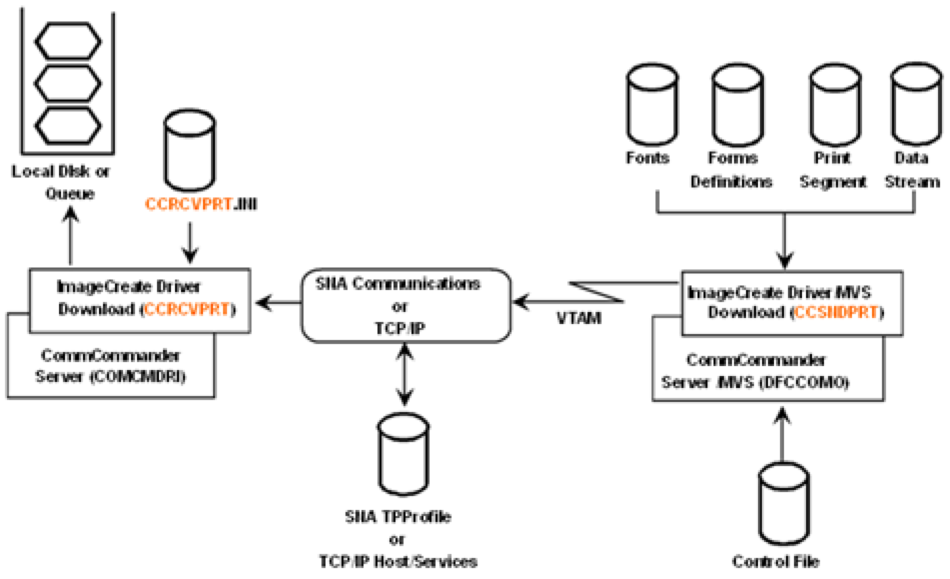

Local Disk or Queue
CCRCVPRT.INI
Fonts
Forms Definitions
Print Segment
Data Stream
ImageCreate Driver Download (CCRCVPRT)
CommCommander Server (COMCMDRI)
SNA Communications or TCP/IP
VTAM
ImageCreate Driver MVS Download (CCSNDPRT)
CommCommander Server MVS (DFCCOMO)
SNA TPProfile or TCP/IP Host/Services
Control File

Figure 2: Downloading a Data Stream from the mainframe to the PC

SETTING UP COMMCOMMANDER IMAGECREATE DRIVER

To Set Up Commcommander Imagecreate Driver

Commcommander Image Create Driver (CCRCVPRT) is automatically invoked whenever a request is received from the partner node. CCRCVPRT is considered a transaction program (TP). A TP must be predefined, via transaction profile, before invocation.

Defining Transaction Profiles and Initialization Entries TCP/IP

NT — For information on defining transaction program profiles, see *Setting Up (Inbound) Commcommander Initialization Section Entries for NT/UNIX* on page 64.

UNIX — For information on defining transaction program profiles, see *Setting Up (Inbound) Commcommander Initialization Section Entries for NT/UNIX* on page 64.

MVS — For information on the Communication Initialization File, see *To Customize the Outbound Communication Initialization File for MVS* on page 37.

SNA

NT — For information on defining transaction program profiles, see *Defining Transaction Programs* on page 47.

Use the following values when defining the Image Create TP:

TP Name:	CCRCVPRT
Command line:	**C:\ CCRCVPRT\CCRCVPRT.EXE C:\ CCRCVPRT** (this parameter specifies the location of the Imagecreate Driver .INI file).

UNIX — For information on defining transaction program profiles, see *Defining Transaction Programs* on page 47.

Use the following values when defining the Image Create TP:

Profile name:	**CCRCVPRT**
Command line:	**/u/CCRCVPRT**
TPN:	CCRCVPRT
Full Path:	**/u/CCRCVPRT/CCRCVPRT**
Standard output	**/u/CCRCVPRT**
Standard error	**/u/CCRCVPRT**

MVS — For information on the Communication Initialization File, see *To Customize the Outbound Communication Initialization File for MVS* on page 37.

To Set Up the Commcommander Imagecreate Driver INI File

Because CCRCVPRT operates automatically, you must set the options before the transaction profile is invoked.

1. Determine whether the Imagecreate output will be written to a file or a Docucorp Queue Systems queue, and set the “QueueType=” parameter accordingly.
2. Determine if the outbound message handler, Commcommander Imagecreate Driver/ MVS, will specify the file or queue name to which the Imagecreate output will be written. If the Commcommander Imagecreate Driver is to override the MVS specification, change the appropriate parameters in the CCRCVPRT.INI file.

To set these options, edit the appropriate “.INI” file.

Note Remember to remove the semicolon (;) from Column 1 to activate the selected parameters.

```
[Docusave]
;      CCRcvPrt.ini    (a semicolon in position one is a comment line)
;
;              The name of the server machine where the
;              Docucorp queue is running.
;              The default value is localhost
ServerName=localhost
;              ---------------------------------------------------------------
;              The name of the QMS Queue that will capture the download
;              print datastreams.
;              The default value is: Docusave
QueueName=Docusave
;              ---------------------------------------------------------------
;              The transport protocol type used by QMS Queue
;              The default value is: TCPIP
QueueType=TCPIP
;              YES - all downloaded resources are written to the specified
;                    JOBS queue.
;              NO - all downloaded resources are written to the specified file
;                   directory.
;                   The default value is NO.
ResourcesToQueue=NO
;               --------------------------------------------------------------
;               The directory where Docucorp Font Printer Resources are stored.
;               The default value is resource.
VfontDir=resource
;               --------------------------------------------------------------
;               The directory where AFP Font Printer Resources are stored.
;               The default value is resource.
AfontDir=resource
;               ---------------------------------------------------------------
;               The directory where AFP PSEG Printer Resources are stored.
;               The default value is resource.
PsegDir=resource
;               ---------------------------------------------------------------
;               The directory where AFP Overlay Printer Resources are stored.
;               The default value is resource.
OverlayDir=resource
;               ---------------------------------------------------------------
;               The directory where Docucorp Forms Definitions are stored.
;               The default value is resource.
FormDefDir=resource
;               ---------------------------------------------------------------
;               The directory where Xerox Metacode font printer resources are stored.
;               The default value is resource.
MfontDir=resource
;               ---------------------------------------------------------------
;               The directory where Xerox Metacode FRMs are stored.
;               The default value is resource.
FrmDir=resource
;               ---------------------------------------------------------------
;               The directory where Xerox Metacode IMGs are stored.
;               The default value is resource.
ImgDir=resource
;               ---------------------------------------------------------------
;               The directory where temporary work files are stored.
;               The default value is the current process directory.
TempDir=
;               ---------------------------------------------------------------
```

Figure 3: Sample Download CCRCVPRT.INI File

Field	Value
ServerName	The name of the server machine where the Docucorp queue is running. (Check with your LAN Administrator for the network server name.) The default value is**: localhost**
QueueName (Required)	The name of the QMS Queue that will capture the download print datastreams. **The default value is: Docusave**
QueueType (Required)	The protocol type use by the QMS Queue when transferring print datastreams. The default value is**: TCPIP**
ResourcesToQueue	Whether necessary job resources are downloaded to a queue or directory. The valid values are as follows: **YES** — all downloaded resources are written to the specified JOBS queue. **NO** — all downloaded resources are written to the specified file directory. The default value is **NO**.
VfontDir	The directory where Docucorp Font Printer Resources are stored. For convenience, you can specify the same directory or sub-directory where you installed the Imagecreate executable modules. For more information about the appropriate entry, see the *Imagecreate Server Installation and User Guide*. The default value is **resource**.
AfontDir	The directory where AFP Font Printer Resources are stored. For your convenience you can specify the same directory or sub-directory where you installed the Imagecreate executable modules. For more information about the appropriate entry, see the *Imagecreate Server Installation and User Guide*. The default value is **resource**.
PsegDir	The directory where AFP PSEG Printer Resources are stored. For convenience, you can specify the same directory or sub-directory where you installed the Imagecreate executable modules. For more information about the appropriate entry, see the *Imagecreate Server Installation and User Guide.* The default value is **resource**.
OverlayDir	The directory where AFP Overlay Printer Resources are stored. For convenience, you can specify the same directory or sub-directory where you installed the Imagecreate executable modules. For more information about the appropriate entry, see the *Imagecreate Server Installation and User Guide*. The default value is **resource**.
FormDefDir	The directory where Docucorp Forms Definitions are stored. For convenience, you can specify the same directory or sub-directory where you installed the Imagecreate executable modules. For more information about the appropriate entry, see the *Imagecreate Server Installation and User Guide*. The default value is **resource**.

Field	Value
MfontDir	The directory where Xerox Metacode font printer resources are stored. For convenience, you can specify the same directory or sub-directory where you installed the Imagecreate executable modules. For more information about the appropriate entry, see the *Imagecreate Server Installation and User Guide*. The default value is **resource**.
FrmDir	The directory where Xerox Metacode FRMs are stored. For convenience, you can specify the same directory or sub-directory where you installed the Imagecreate executable modules. For more information about the appropriate entry, see the *Imagecreate Server Installation and User Guide*. The default value is **resource**
ImgDir	The directory where Xerox Metacode IMGs are stored. For convenience, you can specify the same directory or sub-directory where you installed the Imagecreate executable modules. For more information about the appropriate entry, see the *Imagecreate Server Installation and User Guide*. The default value is **resource**.
TempDir	The directory where temporary work files, encountered during normal processing, are stored. For convenience, you can specify the same directory or sub-directory where you installed the Data Stream executable modules. For more information about the appropriate entry, see the Path= explanation above. The default value is **resource**.

Setting Up Imagecreate Driver/MVS

At this time, you should have already installed Commcommander Imagecreate Driver/MVS. The installation of the Commcommander Imagecreate Driver/MVS consists of loading the program(s) from a tape to the load library. For more information on installing the Imagecreate Driver and the specific steps involved, please see *Installing MVS Server Components* on page 24.

Within the JCL members copied during the TAPEREAD job is the CCSNDPRT member.

1. Modify the JCL to meet your site's requirements. The **bold** portions of the JCL below represent the site-specific parts.

```
//---------- JOB CARD HERE ----------
//*          ********************************************************
//*          Imagecrate  CONTROL  FILE  ALLOCATE  AND  LOAD
//*          ********************************************************
//*
//*          -------------------------------------------------------
//*          Imagecrate  CONTROL  FILE  FOR  CCSNDPRT
//*          -------------------------------------------------------
//*
//STEP1  EXEC   PGM=IEBGENER
//SYSPRINT  DD   SYSOUT=*
//SYSIN     DD    DUMMY
//*               DFCVRFO  output  control
//SYSUT2    DD    DSN=isi.DFCCOM.VnnRnn.CCSNDPRT.CTL,
//                DISP=(NEW,CATLG),
//                DCB=(RECFM=FB,LRECL=80,BLKSIZE=80),
//                UNIT=SYSDA,
//                SPACE=(TRK,(1,2))

//SYSUT1    DD    *

*****           Imagecreate  Download  control  entries  go  here

*****
*****  You will set these values after the creation of the Imagecreate
       Download datasets

*****

*****                           See step 4 below.

/*
//
```

Figure 4: SNDPRTP JCL

2. Submit the job.
3. Check the job log for errors when the job is complete. If necessary, correct the errors and resubmit the job.
4. Edit the PROFILE dataset **(isi.DFCCOM.VnnRnn.CCSNDPRT.CTL)** created in Steps 1 through 3, stated above. Using the keyword definitions listed below ensure the desired values are set.

Note The following figure is an example of the PROFILE entries. This example represents the exact keywords needed for Imagecreate Download.

You need to choose which protocol to be used and select or setup the correct control card.

```
;Imagecreate Driver
; Application control card stements.
{CommCommander}
ServerName=localhost
LUname=LU01
LogMode=LM01
QueueName=PRT
QueueType=DDNAME
TPname=CCRCVPRT
DocsPerJob=2
IncludeProcessCard=YES
VerifyResources=NO
DJDE=E'$$XEROX'
DJDEoffset=1
DJDEskip=8
RAUX=X'1111111111111111'
RPAGE=X'01FFFF20FFFF'
ROFF=X'1212121212121212'
RSTACK=E'REPORT END'
RAUXoffset=1
ROFFoffset=1
RPAGEoffset=1
RSTACKoffset=1
```

Figure 5: CCSNDPRT Control File

NOTE
Records beginning with a semicolon (;) are comments. In the above example, the first two records are comment records.

Field	Value
ServerName	Enter the DDname of the PDS from which the input is routed. This name must reference a DD statement (e.g., IN). **The default value is: localhost**
QueueName	Enter the DDname of the PDS from which the input is routed. This name must reference a DD statement (e.g., IN). **The default value is: PRT**
QueueType	The “type” of PDS from which the input is routed. This name must reference a DD statement (e.g., DDNAME). The only valid value is **DDNAME** . **The default value is: DDNAME**
TPname	The name of the Transaction Program on the server machine, which identifies CCRCVPRT. **The default value is: TP01**
TPname	The name of the Transaction Program on the server machine, which identifies CCRCVPRT. **The default value is: TP01**
LUname	**(TCP/IP)** This is the symbolic name of the partner machine. Your network administrator defines this name in the TCP/IP MVS Host file. **(SNA)** The LU (logical unit) that identifies the server machine where **tpname** is defined. **The default value is: LU01**

Field	Value
LogMode	**(TCP/IP)** This is the port number being used by the partner node that identifies Commcommander to the host. **(SNA)** The name of the Log Mode table under MVS that will be used for the APPC conversation between CCSNDPRT and CCRCVPRT. **The default value is: LM01**
TempDir	This value defines the DDName of the file to be used as a temporary work file. Refere to the installed CCSNDPRT JCL for the proper defining attributes for this dataset. **The default value is: TEMP**
IncludeProcessCard	When this flag is 'Y', a data record in sent to the client containing all of the pertinent information about the datastream to follow. **The default value is: Y**
VerifyResources	When this flag is 'Y', the client side of the application is queried to determine if the necessary resources to process the incoming datastream are present. If not, the resources will be collection from the server side repository and transmitted to the client prior to the datastream transmission. **The default value is: Y**
DJDE	This value is the character string the Xerox laser Printing System uses to identify the DJDE records in your Imagecreate output. The value you enter must match the PREFIX option value of the relevant JDE's IDEN statement. If you enter only the character string, this string constant appears in the JDE in ASCII code. If the JDE uses another code or if you want to specify ASCII explicitly, you can specify the code in the first field of the ID (**A** for ASCII, **E** for EBCDIC, or **X** for Hex). When **A**, **E**, or **X** is used to specify the code, the character string must be enclosed within single quotes. The default value is: E'$$XEROX'
DJDEoffset	The Offset value specifies the number of bytes (beginning at 0) from the beginning of the user portion of the record to the beginning of the prefix string constant of the DJDE record. This value can be a negative number. **The default value is: 1**
DJDEskip	The Skip value specifies the number of bytes (beginning at 0) from the beginning of the DJDE options. This value can be a negative number. **The default value is: 8**
DocsPerJob	This parameter specifies the maximum number of documents included in each imaging job. For example, if your data stream contains 1000 documents and you set **maxDocs=20**, each imaging job contains a maximum of 20 documents. **The default value is: 10**
LeftMargin	Left-hand margin (100 dpi) for line printer data stream. **The default value is: 0**
TopMargin	Top margin (100 dpi) for line printer data stream. **The default value is: 0**
LPI	Output lines per inch. **The default value is: 6**
LPP	Output lines per page. **The default value is: 60**
RAUX	The character string referenced by the test-expression of the relevant JDE's RAUX special processing statement. Specifically, the character string must match the string assigned to the CONSTANT command of the TABLE statement referenced by the CRITERIA statement which in turn is referenced, by the TEST command of the RAUX statement. If you enter only the character string, this string constant appears in the RAUX in ASCII code. If the JDE uses another code or if you want to specify ASCII explicitly, you can specify the code in the first field of the ID (**A** for ASCII, **E** for EBCDIC, or **X** for Hex). When **A** , **E**, or **X** is used to specify the code, the character string must be enclosed within single quotes. **The default value is: X'1111111111111111'**

Field	Value
RPAGE	The character string referenced by the test-expression of the relevant JDE's RPAGE special processing statement. Specifically, the character string must match the string assigned to the CONSTANT command of the TABLE statement referenced by the CRITERIA statement which, in turn is referenced by the TEST command of the RPAGE statement. If you enter only the character string, this string constant appears in the RPAGE in ASCII code. If the JDE uses another code or if you want to specify ASCII explicitly, you can specify the code in the first field of the ID (**A** for ASCII, **E** for EBCDIC, or **X** for Hex). When **A** , **E**, or **X** is used to specify the code, the character string must be enclosed within single quotes. **The default value is: X'01FFFF20FFFF'**
ROFF	The character string referenced by the test-expression of the relevant JDE's ROFF special processing statement. Specifically, the character string must match the string assigned to the CONSTANT command of the TABLE statement referenced by the CRITERIA statement which in turn is referenced, by the TEST command of the ROFF statement. If you enter only the character string, this string constant appears in the ROFF in ASCII code. If the JDE uses another code or if you want to specify ASCII explicitly, you can specify the code in the first field of the ID (**A** for ASCII, **E** for EBCDIC, or **X** for Hex). When **A** , **E**, or **X** is used to specify the code, the character string must be enclosed within single quotes. **The default value is: X'1212121212121212'**
RSTACK	The character string referenced by the test-expression of the relevant JDE's RSTACK special processing statement. Specifically, the character string must match the string assigned to the CONSTANT command of the TABLE statement referenced by the CRITERIA statement which in turn is referenced, by the TEST command of the RSTACK statement. If you enter only the character string, this string constant appears in the RSTACK in ASCII code. If the JDE uses another code or if you want to specify ASCII explicitly, you can specify the code in the first field of the ID (**A** for ASCII, **E** for EBCDIC, or **X** for Hex). When **A** , **E**, or **X** is used to specify the code, the character string must be enclosed within single quotes. **The default value is: E'REPORTEND'**
RAUXoffset	The Offset value specifies the number of bytes (beginning at 0) from the beginning of the user portion of the record to the beginning of the prefix string constant of the DJDE record. This value can be a negative number. **The default value is: 1**
ROFFoffset	The Offset value specifies the number of bytes (beginning at 0) from the beginning of the user portion of the record to the beginning of the prefix string constant of the DJDE record. This value can be a negative number. **The default value is: 1**
RPAGEoffset	The Offset value specifies the number of bytes (beginning at 0) from the beginning of the user portion of the record to the beginning of the prefix string constant of the DJDE record. This value can be a negative number. **The default value is: 1**
RSTACKoffset	The Offset value specifies the number of bytes (beginning at 0) from the beginning of the user portion of the record to the beginning of the prefix string constant of the DJDE record. This value can be a negative number. **The default value is: 1**
DefaultFont	This value is included in the processing record for informational purposes only. It's intended to inform the end user the name of the default font to use, if the actual datatstream fonts are not available. This field is rarely used. **No default value.**
DefaultFont2	This value is included in the processing record for informational purposes only. It's intended to inform the end user the name of the default font to use, if the actual datatstream fonts are not available. This field is rarely used. **No default value.**

1. Modify the CCSNDPRT JCL to meet your site's requirements. The **bold** portions of the JCL represent the site-specific parts.

```
//-------- JOB CARD HERE --------------
//DCUSER JOB ...
//* ******************************************************************
//*  CommCommander CCSNDPRT Driver/MVS Download to PC
//*
//*  CCSNDPRT - COMMCOMMANDER CCSNDPRT DRIVER/MVS
//*             TRANSFER IMAGES FROM MAINFRAME TO PC
//*
//* ******************************************************************
//JOBLIB   DD DISP=SHR,DSN=DCUSER.DOCUSAVE.V03R00.LEVEL00.LOADLIB
//         DD DISP=SHR,DSN=DCUSER.DOCUSAVE.V03R00.PLC.LOADLIB
//         DD DISP=SHR,DSN=DCUSER.SASC.V700.RUNTIME.LIBRARY
//         DD DISP=SHR,DSN=DCUSER.DFCCOM.V02R01.LOADLIB
//*
//CCSNDPRT EXEC PGM=CCSNDPRT
//*CSNDPRT EXEC PGM=CCSNDPRT,PARM='IP=nnn.nnn.nnn.nnn'
//*
//DFCINIT  DD DISP=SHR,DSN=DCUSER.DFCCOM.V02R01.DFCCOMO.DFCINIT
//*
//DFCMSG   DD SYSOUT=*,DCB=BLKSIZE=2048,OUTLIM=2000
//DFCLOG   DD SYSOUT=*,DCB=BLKSIZE=2048,OUTLIM=2000
//DFCTRACE DD SYSOUT=*,DCB=BLKSIZE=2048,OUTLIM=2000
//ISIWTO   DD SYSOUT=*
//*
//SYSPRINT DD SYSOUT=*,OUTLIM=2000
//SYSTERM  DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//PROFILE  DD *
{CommCommander}
TPname=TP01
LUname=LU01
LogMode=LM01
QueueType=PDS
QueueName=PRT
ServerName=localhost
IncludeProcessCard=Y
VerifyResources=Y
DJDE=E'$$XEROX'
DJDEoffset=1
DJDEskip=8
DocsPerJob=10
DPI_OUT=0
COMPRESS=4
LeftMargin=0
TopMargin=0
LPP=60
RAUX=X'1111111111111111'
RPAGE=X'01FFFF20FFFF'
ROFF=X'1212121212121212'
RSTACK=E'REPORTEND'
RAUXoffset1
ROFFoffset1
RPAGEoffset=1
RSTACKoffset=1
DefaultFont=
DefaultFont2=
/*
//*
```

Continued on Next Page

Figure 6: CCSNDPRT JCL (Figure 1 of 2)

Continued from Previous Page

```
//PRT      DD DISP=SHR,DSN=DCUSER.DFCCOM.V02R01.META
//LOG      DD SYSOUT=*
//* ****************************************************************
//*FRM      DD DISP=SHR,DSN=DCUSER.DFCCOM.V02R01.FRM
//*IMG      DD DISP=SHR,DSN=DCUSER.DFCCOM.V02R01.IMG
//* ****************************************************************
//* IBM
//* ****************************************************************
//AFONT    DD DISP=SHR,DSN=DCUSER.DFCCOM.V02R01.AFP.FONT3820
//FORMDEF  DD DISP=SHR,DSN=DCUSER.DFCCOM.V02R01.AFP.FORMDEF
//OVERLAY  DD DISP=SHR,DSN=DCUSER.DFCCOM.V02R01.AFP.OVERLAY
//PSEG     DD DISP=SHR,DSN=DCUSER.DFCCOM.V02R01.AFP.PSEG
//* ****************************************************************
//* XEROX
//* ****************************************************************
//MFONT    DD DISP=SHR,DSN=DCUSER.DFCCOM.V02R01.META.MFONTS
//
```

Figure 7: CCSNDPRT JCL (Figure 2 of 2)

Parameter	Value
Job Card	Replace with your MVS job card.
JOBLIB	Specify the Commcommander Load Library. This job must have access to the SAS/C runtime libraries. You may concatenate the SAS/C runtime libraries in the JOBLIB JCL statement.
DFCINIT	The Commcommander Initialization File. For comprehensive information on the DFCINIT file, see *To Customize the Outbound Communication Initialization File for MVS* on page 37.
SYSTERM, **SYSPRINT,** **SYSOUT**	Ddnames used by the SAS/C runtime environment. If an error occurs during processing, these destinations should be checked for SAS/C warning messages.
DATASTRM	The Metacode data stream to be downloaded and rasterized.
DFCMSG **DFCLOG** **DFCTRACE** **ISIWTO**	Ddnames used by the Commcommander environment. If an error occurs during processing, these destinations should be checked for Commcommander messages.
PROFILE	An input stream data set containing the control card settings for the job parameters. For a description of each of the cards and their settings, see "Metacode SYSIN Control Cards" on page 84.
VFONT	The VFont file created earlier by IMCRVFT. This PDS contains the VFont equivalent of the Xerox fonts. Vfonts aren't needed if you have native Metacode fonts (see next).
MFONT	This PDS contains the native Metacode fonts.
FRM	This PDS contains the native Metacode FRMs.

Parameter	Value
IMG	This PDS contains the native Metacode IMGs.

USING COMMCOMMANDER IMAGECREATE DRIVER

To Use Commcommander Imagecreate Driver/NT/UNIX

1. (SNA/NT) Ensure that SNA Server communication software is up and running.
   -or-
   (TCP/IP) Ensure that Docucorp's Commcommander Server Inbound program is up and running.
2. Ensure the proper settings are specified in the Commcommander Imagecreate Driver CCRCVPRT.INI file.
3. Ensure the directory named in the CCRCVPRT.INI file already exists if you are directing the resource to a file. If you are directing the resource to a Docucorp Queue Systems queue, make sure the queue program **ISIQMSD** is up and running.

   When using the SNA protocol, the Imagecreate transaction program (TP) is automatically invoked by Microsoft's or IBM's SNA Server communications software whenever the partner node's Imagecreate Driver starts the conversation with this local node Imagecreate Driver. For TCP/IP, the TP is invoked by Docucorp's Commcommander Server Inbound.

   Multiple Transaction Program Profiles may be defined, with each pointing to different directories containing individual copies of Commcommander Imagecreate Driver and its CCRCVPRT.INI file. One example in which you might want to use multiple profiles is to have one TPN set up to write to a file, and another to put the resource in a Docucorp Queue Systems queue. By selecting the TPN name in the partner node control cards, you can select the type of destination to which the resource will be routed.

To Review Messages

Browse any and all Commcommander error messages by viewing the CCRCVPRT.LOG file, which is located in the same directory as the CCRCVPRT executable file. The CCRCVPRT.LOG file contains any protocol errors or trace messages produced during a Commcommander session. The messages are written chronologically from top to bottom. Therefore to view the most recent messages go to the bottom of the file and scroll backwards.

To Use Commcommander Imagecreate Driver/MVS

The Commcommander Imagecreate Driver/MVS initiates the conversation with the partner PC node. This is a program that can be executed as a job step within an existing job, or as a separate job. You have the option of transferring the Data Stream to the PC in the same job that creates the Data Stream or, by allocating the Data Stream as a permanent file, you can transfer it at anytime using a separate job.

Note It's recommended that you use the JCL members created during the installation step. The examples in this manual are only for illustrative purposes.

1. Modify, if necessary, any CCSNDPRT control cards.
2. After making any necessary changes, submit the appropriate CCSNDPRT JCL to MVS for processing.

To Review Messages

Browse any and all Commcommander error messages by viewing the SYSOUT and REPORT datasets.

Chapter 8

Commcommander VLAM Driver

Overview

The Commcommander VLAM Driver is a Docucorp software product that allows PC users and PC applications to load, extract, and inquire about data in a mainframe VLAM Library. The VLAM Driver requires the Commcommander Server product as well as the Virtual Library Access Method (VLAM) product component. VLAM is distributed as a component part of other Docucorp products, i.e. Documerge. This server provides the communication component that controls the actual transfer of data between the PC and mainframe.

Note This guide doesn't explain the basic concepts and ideas behind the Docucorp Virtual Library Access Method (VLAM). Please refer to the *VLAM User Guide and Reference* manual for explanations and installation procedures.

PC applications that create data to be stored on a host VLAM Library simply store the output in a file. The driver can read this file, upload or download the data to or from the host via Commcommander, and load the data into or retrieve the data from the specified Library. Also, you can use VLAMcommander for Windows to conveniently access a host Library and directly load or extract data through the VLAM Driver. For more information, see .

How Commcommander and the VLAM Driver Work

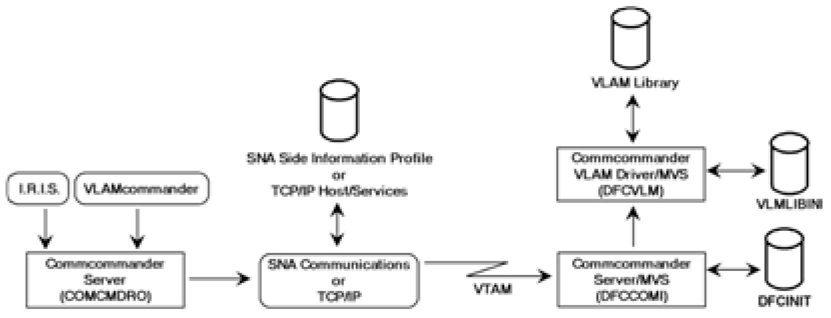

VLAM Library
SNA Side Information Profile or TCP/IP Host/Services
I.R.I.S.
VLAMcommander
Commcommander VLAM Driver/MVS (DFCVLM)
VLMLIBINI
Commcommander Server (COMCMDRO)
SNA Communications or TCP/IP
VTAM
Commcommander Server/MVS (DFCCOMI)
DFCINIT

Figure 8: Commcommander and the VLAM Driver communicate with the host

The Commcommander communication components are installed on the PC server and the host. These components can communicate and transfer data back and forth using communication services that operate within TCP/IP or IBM System Network Architecture (SNA) environments. Communication between the server and the user's Windows workstation is handled by Sockets Named Pipes protocol drivers.

SETTING UP COMMCOMMANDER VLAM DRIVER

SETTING UP COMMCOMMANDER VLAM DRIVER/NT/UNIX

To Set Up Commcommander VLAM Driver/NT/UNIX

The Commcommander VLAM Driver/NT/UNIX is tightly coupled with, and pre-configured by the setup of, those Commcommander Drivers that use VLAM's API. Therefore, there are no additional setup steps to be performed at this time.

SETTING UP COMMCOMMANDER VLAM DRIVER/MVS

To Customize the VLAM Driver Settings File

Within the JCL members copied during the TAPEREAD job is the VLMLIBNI member. VLMLIBNI creates the VLMLIBNI dataset used by the Commcommander VLAM Driver/MVS (DFCVLM).

1. Modify the JCL to meet your site's requirements. The **bold** portions of the JCL below represents the site specific parts.

```
//-------- JOB CARD HERE --------------
//*
//*           ******************************************************
//*           VLAMCOMMANDER INITIALIZATION FILE ALLOCATE AND LOAD
//*           ******************************************************
//*
//STEP1  EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *,DCB=(LRECL=80,BLKSIZE=80)
 GENERATE MAXFLDS=3,MAXLITS=70
 RECORD   FIELD=(80,1,,1),                                            X
          FIELD=(40,'                                       ',,81),   X
          FIELD=(30,'                              ',,121)
//SYSUT2   DD DSN=isi.DFCCOM.VnnRnn.VLMLIBNI,
//            DISP=(NEW,CATLG),
//            SPACE=(TRK,(1,2)),
//            UNIT=SYSDA,
//            DCB=(RECFM=F,LRECL=150,BLKSIZE=150)
//SYSUT1   DD DATA,DCB=(LRECL=80,BLKSIZE=80)
* ID----VRM---DATE--    TYP-DDN-----DISP-DSN---------------------------
--
*
..+....1....+....2....+....3....+....4....+....5....+....6....+....7..
..+.
DFCVLM  VvRrMmYYMMDD    DSN ddname1 SHR  data.set.name.1
DFCVLM  VvRrMmYYMMDD    DSN ddname2 SHR  data.set.name.2
DFCVLM  VvRrMmYYMMDD    DSN ddname3 SHR  data.set.name.3
/*
//
```

Figure 9: VLMLIBNI JCL

2. Submit the job.
3. Check the job log for errors when the job is complete. If necessary, correct the errors and resubmit the job.
4. Edit the VLMLIBNI dataset (e.g., **isi.DFCCOM.VnnRnn.VLMLIBNI**). Use the keyword definitions listed below to ensure the correct protocol values are set. For more information, see Settings File Information.
5. Add the VLMLIBNI DD statement to the Commcommander/MVS Server. For more information, see the JCL examples To Update the Startup JCL (for TCP/IP and ACF/VTAM versions) or To Update the TP Profile (for APPC/MVS versions).

These VLAM datasets are accessed from the Commcommander/MVS Server job DFCCOMI. The User ID in this job must be authorized to access and update these VLAM datasets.

Note **For TCP/IP and ACF/VTAM mainframe environments only:** Be sure to terminate Commcommander on the host before making changes to the Setting File. Once the changes have been made, they'll take effect the next time you start Commcommander.

For now, the Settings File contains no more than the types of information shown below. The file may contain additional types of information in the future.

```
********************************* TOP OF DATA ******************************
* ID----VRM---DATE--    TYP-DDN-----DISP-DSN------------------------------
* ..+....1....+....2....+....3....+....4....+....5....+....6....+....7....+
DFCVLM  VvRvMmYYMMDD    DSN ddname1 SHR  data.set.name.1
DFCVLM  VvRvMmYYMMDD    DSN ddname2 SHR  data.set.name.2
DFCVLM  VvRvMmYYMMDD    DSN ddname3 SHR  data.set.name.3
****************************** BOTTOM OF DATA ****************************
```

Figure 10: Commcommander VLAM Driver/MVS Settings File

Each record in this file contains a variety of information about a single host VLAM library. Provide one record for each VLAM library that can be accessed by the client application running on the PC partner. The values in these records are positional and must begin in the position identified by the scale record in the above example.

The VLAM Driver (DFCVLM) inserts some of the values into skeleton JCL. The JCL invokes programs designed to carry out a wide variety of VLAM tasks. If an error should occur, you may see one of the specified values in an error message. For an explanation of the file contents broken into seven general columns, see Settings File Information.

WARNING! Be sure to remove the above sample entries prior to saving the file.

Settings File Information

The first twenty characters contain diagnostic information used by the VLAM Driver. If any non-comment record (e.g., any record not beginning with an asterisk) doesn't begin with module identifier, version, and format data information, it's rejected as invalid.

Column	Value
ID	The module identifier. DFCVLM is the name of the module that reads this file.
VRM	The version, release, and modification date of the VLAM Driver software.
DATE	The date that identifies the format for this settings file. The date is in the form of *yymmdddd*. As the format of the settings file changes to accommodate additional VLAM Driver features, the date changes to identify the new format.
TYP	Identifies the type of information within this record. At present, the type of information is DSN, for data set name. As new features are added to the VLAM Driver, different information types might be needed. This column is reserved for identifying any future information types.
DDN	A DDname for the VLAM library. This ddname can be any ddname you provide and must be unique within this VLMLIBNI file.
DISP	The disposition (DISP=) of the VLAM library. This disposition values are inserted into the JCL that the VLAM Driver uses to perform VLAM tasks.
DSN	The data set name of the VLAM library on the mainframe (DSN=). These VLAM library datasets should already exist and will have been created and in use by other Docucorp products (e.g., Documerge). **Note:** Records beginning with "* ", an asterisk followed by a blank space, in positions 1 and 2 are treated as comment records and are ignored. Comment records can be located anywhere within the file.

USING COMMCOMMANDER VLAM DRIVER

The Commcommander VLAM Drivers are invoked automatically by Commcommander Drivers that use VLAM's API. The Docucorp products initiate a conversation through the PC Commcommander VLAM Driver which, in turn, requests the Commcommander Server to transfer the data. Commcommander VLAM Driver/MVS is automatically invoked to process the request from the partner node.

To Use Commcommander VLAM Driver

There is no direct user interface to the Commcommander VLAM Driver. For more information about using the Commcommander VLAM Drivers, please refer to Using VLAMcommander.

Chapter 9

VLAMcommander

OVERVIEW

This schematic diagram shows how various applications use VLAMcommander and its communication components. Notice the multiple Libraries: VLAMcommander can write to any host Library specified by the user.

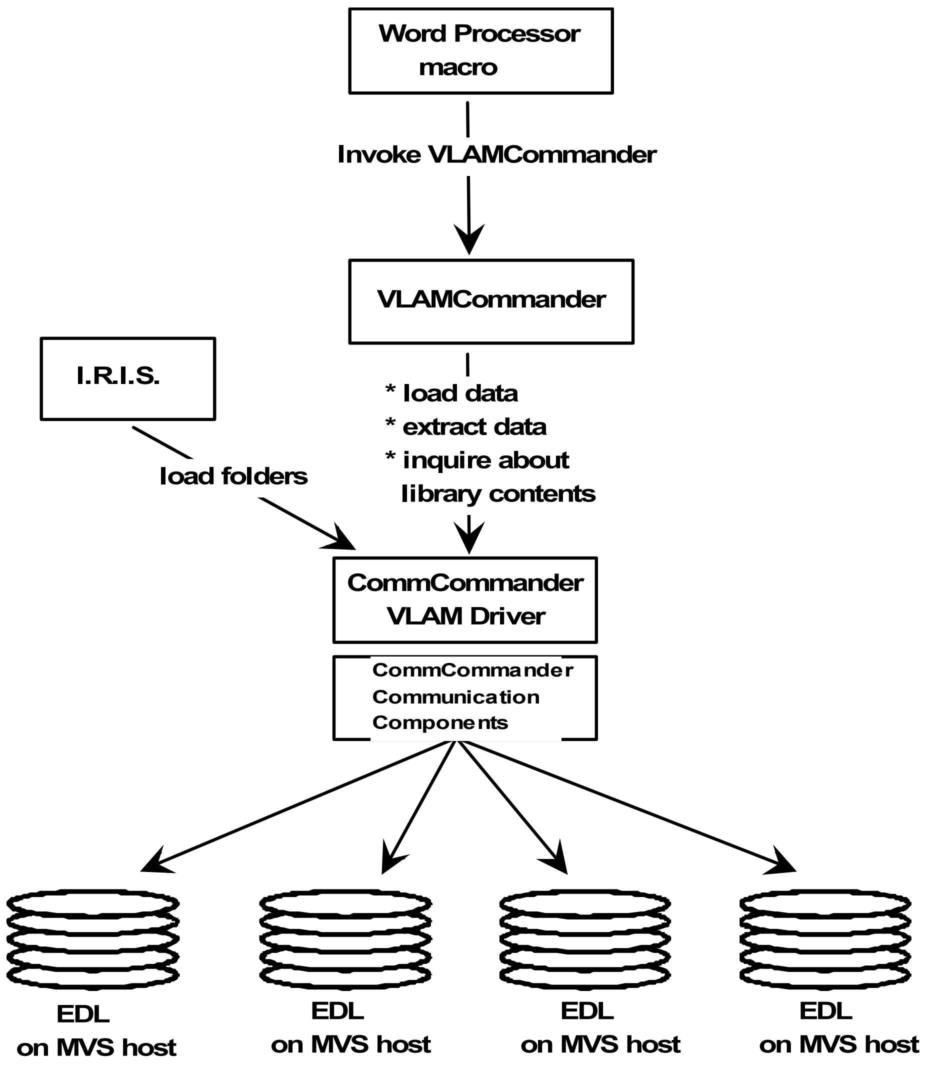


Figure 11: VLAMcommander Overview

VLAMcommander runs as a 16-bit application on Windows 3.1, Windows 9x, or Windows NT. VLAMcommander requires VLAM 3.1 (or higher), which is supplied with other Docucorp products, such as **Documerge**®, **I.R.I.S.**™, **Docuwise**™, and **Docuvise**™. The Commcommander distribution tape contains the load modules for Commcommander and the Docucorp communications software. VLAM, on the other hand, is distributed by the other Docucorp mainframe products. Therefore, any jobs that require VLAMcommander must have both Commcommander and the Docucorp product load libraries specified in the Commcommander/MVS Server JCL.

If you're installing Commcommander but don't have VLAM 3.1 (or higher), you must request and install a separate VLAM distribution tape from Docucorp.

LIBRARY MEMBER BASICS

Before moving on to VLAMcommander functions and particular parts of the dialog box, you must develop a basic understanding of a mainframe VLAM library and its members. If you need additional information, see the *VLAM User Guide*.

The contents of a VLAM library is organized into **members**. The dialog box shows the members in the "Laser Mutual Insurance Company EDL" library. See the list box in the lower half of the dialog in To Use VLAMcommander.

The combination of **member name** and **revision level** uniquely identifies each member. The same dialog box shows two members whose names appear to be the same — Thread 0 Doc 0(3). Notice, however, that each name is followed by a number in parentheses. This number represents the revision levels. Revision levels let you store different versions of a member. For most of our users, a higher revision level means a more recent version: Thread 0 Doc 0(3) is an updated version of Thread 0 Doc 0(2).

To make life a bit easier, each member has a **description** field. Many libraries contain members with names based on form numbers. If you're looking for a particular member and don't know the form number (e.g. 40497-3C) you still might be able to find the member based on its description ("Jewelry Rider," for example).

DTN and **Effective Date** are additional identifiers used by the Docucorp Documerge product. A Document Type Number (DTN) is used to associate the member's data with rules for printing and document assembly. Documerge uses Effective Date fields as selection criteria when choosing among dated versions of a given member.

So far, you've dealt with member identifiers. The other parts of a VLAM library member are called **chains**. Chains are the data components. They can contain almost any kind of binary data, including:

- Document images stored as printer data streams
- Source files for a document composition system
- Document images that can be displayed on your screen
- Document images that can be passed among Windows applications
- Lists of folder contents

The following is a schematic illustration of members within a VLAM library.

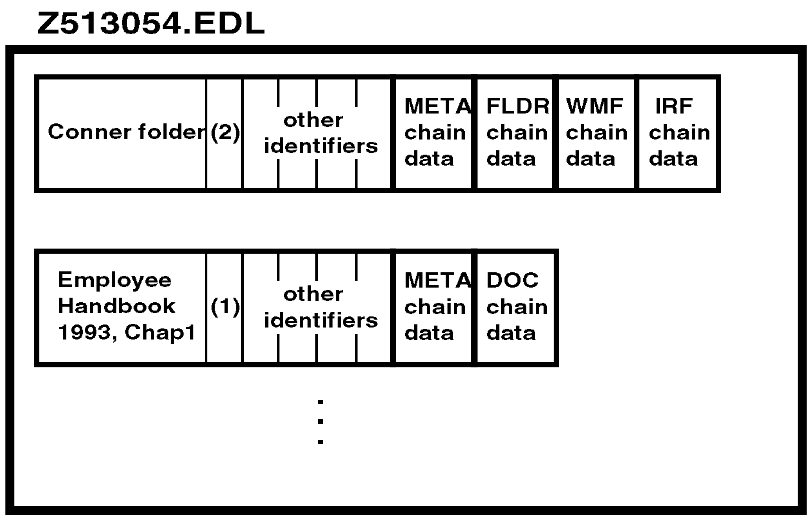


Figure 12: VLAM library structure

Network Communication

The following are tasks normally done by your network administrator. These are not all the network definitions required for the complete network implementation.

TCP/IP Sockets

Your client workstation and server must be network configured, tested and a fully functional part of your TCP/IP network. These machines must be able to communicate with one another using TCP/IP sockets.

NetBEUI Named Pipes

Your client workstation and server must be network configured, tested and a fully functional part of your LAN network. If you are running VLAMcommander on Windows 3.1, you will need to use Microsoft's Network Client to allow these machines to communicate with one another using Named Pipes.

Setup

Starting VLAMcommander

After completing the installation process, you can locate the VLAMcommander icon and initiate the program.

To Start VLAMcommander

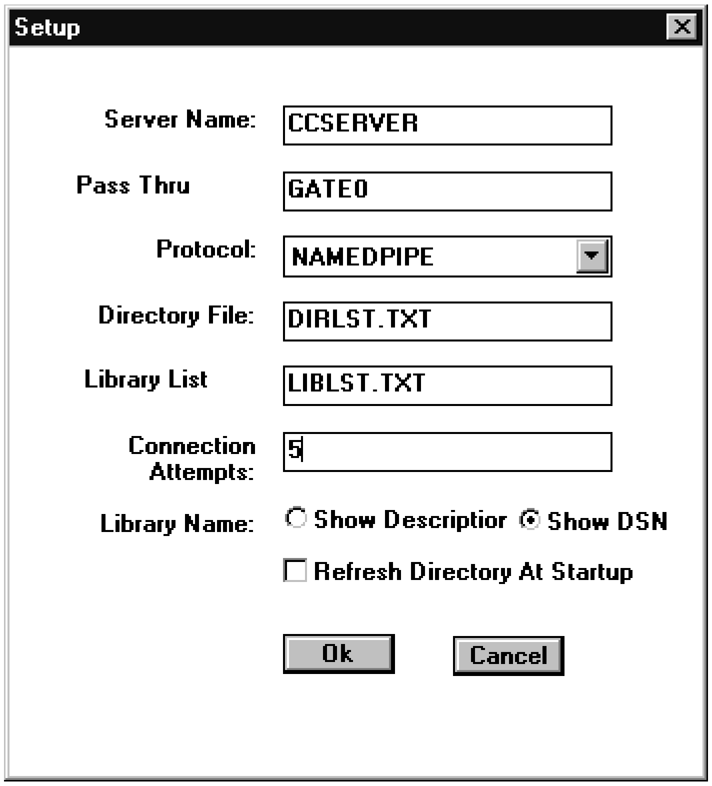
1. Double-click on the VLAMcommander icon and the program displays the VLAMcommander dialog box.
2. Make sure the Commcommander Outbound Server on your NT or UNIX server is up and running and connected to the MVS host.

WARNING! If you're operating in an ACF/VTAM environment, the mainframe Inbound Message Handler must already be started. If this job isn't started, VLAMcommander and Commcommander Server won't be able to communicate with Commcommander Server/MVS.

SETTING UP VLAMCOMMANDER

To Set Up VLAMcommander

1. Click on the **Setup** button from the main VLAMcommander dialog and the program displays the Setup dialog.


Setup

Server Name: CCSERVER
Pass Thru GATE0
Protocol: NAMEDPIPE
Directory File: DIRLST.TXT
Library List LIBLST.TXT
Connection Attempts: 5
Library Name: ( ) Show Descriptior [✓ Show DSN]
☐ Refresh Directory At Startup

2. Complete the various settings according to the following information:

Field	Value
Server Name	The logical name of the LAN server upon which the VLAMcommander modules have been installed. If you don't know the correct name, check with your LAN administrator.
Pass Thru Gate	A logical name associated with the Commcommander Server running on the server machine. The name labels a software gateway through which your PC can communicate with the Commcommander components on the mainframe.

Field	Value
	The Gate name is assigned to Commcommander Server when it is executed on the UNIX or NT server machine. Some sites may install several Commcommander servers. If so, each server is assigned a different Gate name for client/mainframe connections. Different clients may use different *gate* names, yielding faster performance than if they were connecting through one Commcommander server.
Protocol	The type of the input protocol given when Commcommander Server Outbound was started. The protocol type used here MUST match the Commcommander Server Outbound input protocol. (Check with your LAN Administrator for the correct protocol type.) Valid values are NAMEDPIPE or SOCKETS. The default value is **NAMEDPIPE**.
Directory File	The path and name of a file to which VLAMcommander can write member list information. The VLAMcommander dialog box uses this file to display a list of library members. If the file specified in this text box is accidentally erased, you'll see the message "Cannot Open the Directory File" the next time you start VLAMcommander.
Library List	The path and name of a file to which VLAMcommander can write library list information. The VLAMcommander dialog box uses this file to display a list of libraries. If the file specified in this text box is accidentally erased, you'll see the message "Cannot Open the Library List File" the next time you start VLAMcommander.
Connection Attempts	The maximum number of unsuccessful connection queries that can be made before abandoning the attempt to establish a connection with the host.
Library Name	The active radio button affects the library list in the VLAMcommander dialog's Library drop down list box. If you activate Show Description, the list entries are taken from the libraries' description fields. If you activate Show DSN, the entries reflect the libraries' MVS data set names.
Refresh Directory at Startup	When checked, a new member list is compiled whenever you start VLAMcommander. Compiling a member list takes time, though you may wish to sacrifice startup speed for member list accuracy.

3. Click on **OK** to save your selections or **Cancel** to exit without saving.

STOPPING VLAMCOMMANDER

After you've completed your processing and editing of members in the host Library, you can use the Close command to end the program.

To Stop VLAMcommander

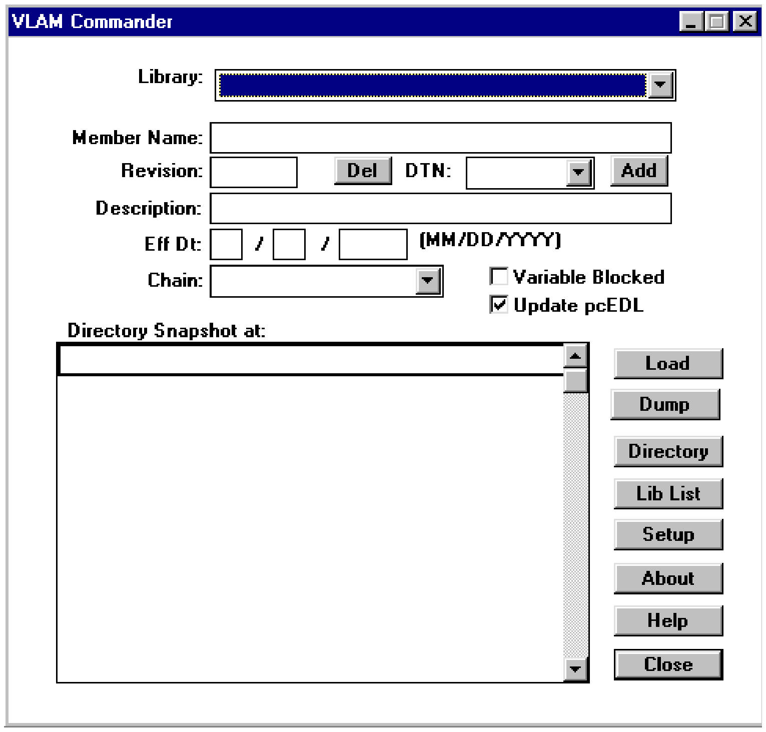
Click on **Close** and VLAMcommander returns you to the Windows Desktop.

USING VLAMCOMMANDER

To Use VLAMcommander

1. Ensure that Commcommander Server is up and running with the correct gate name. For SNA, you must also activate the associated SNA Communications.

2. Double-click on the VLAMcommander icon from the Windows Desktop and you'll see the main VLAMcommander screen.



3. Use the following table. It explains the main items on the screen, including the option buttons.

Field	Value
Library	This drop down list box displays the name of the *active* library, the library whose members are listed in the lower portion of the dialog box.
	Using the drop down list box to select a different library makes your new selection the active library.
Member Name	Every member has a member name. The name may be 1 to 32 characters long.
	Use this text box to enter a member name for loading a new member or for dumping data from an existing member.
Revision	Every member has a revision level. These levels act as version numbers, keeping track of different versions of members that have the same name. A revision level is an integer, ranging in value from 1 to 99999.
	Use this text box to 1) assign a revision level when loading a member or 2) select a member for the Dump command.
DTN	A Document Type Number (DTN) is an integer, 0 — 99999, assigned to each Library member. Documerge uses DTNs to apply formatting rules to documents extracted from the Library. You may need to consult your Documerge project leader for the DTNs to be assigned to the documents you're loading.
	If you don't use Documerge, ignore this text box — VLAMcommander automatically assigns a DTN of zero to the member. **NOTE:** When Documerge assigns a DTN of 0 to the member, any existing DTNs of that same member will be replaced by a single member with DTN 0. For example, if you have a member with DTN 10, 20 and 30 and you upload a new chain to the existing member without specifying a new DTN, it will remove DTNs 10, 20 and 30 and leave you with DTN 0.

Field	Value
	You should only use this text box when members are being loaded: it doesn't affect the dumping of members.
	Use the **Add** and **Delete** buttons beside the DTN drop-down list box to maintain the DTN entries associated with the member. • To remove an entry, select the desired entry and click on **DEL**. • To add an entry, simply type its value and click on **ADD**.
Description	A member description is assigned during loading. Its purpose is to give you a member identifier in addition to the member name. For example, the member name could be the form number and the member description, the form name. The description can be any 36 characters you choose.
Effective Date MM/DD/YYYY	A date, in *mm/dd/yyyy* format, used by Documerge applications. Documerge's VDR component can use effective date fields as selection criteria when choosing among dated versions of a given form. Use this text box to assign an effective date when loading members for Documerge use.
	If you don't use Documerge, ignore this text box — VLAMcommander automatically assigns zeros to the member's effective data fields.
	You should only use this text box when members are being loaded: it doesn't affect the dumping of members.
Chain	A chain is a data component, that portion of the member that holds the document images, composition files, member lists, or any other data you or an application loads to the member. Each member can have up to 32 chains. Choose a descriptive, four character name for the chain's contents. Some chains have reserved names that are set aside for special processing by a variety of Docucorp products. Some of these names are TEXT, DD, IRF, MST, and FLDR.
	When loading a member, use the chain text box to assign a chain name to the data you're loading.
	When dumping data from a member, use the chain text box to specify the chain data that are to be copied to the DOS output file.
Variable blocked	This check box is used only when you're loading data. These are the rules about when you want the box to be checked or unchecked:
	Checked when you're loading data that will be directly extracted by a Docucorp software product (Documerge, for example, extracts document images directly from the Library).
	Unchecked when you're loading data that will be extracted only by you or by a coworker. An example is a word processing file that you wish to store in the Library for convenience. No one will extract this file for use but you or one of your coworkers.
	When this box is checked, VLAMcommander ensures that any internal file markers used to maintain the record format on the mainframe are kept. If the box isn't checked, these markers are stripped.
Directory Snapshot	The date and time indicate when the member list was made. For more information, see Dumping a Library Member (Dump).
Load	Loads a DOS file into the target library. Comprehensive information begins with Loading a Library Member (Load).
Dump	Copies a selected portion of the library into a DOS file. A full description begins with Dumping a Library Member (Dump).
Directory	Lists the members in the selected library. For more information, see Viewing Library Members (Directory).
Lib List	Lists the available mainframe Libraries. For greater detail, see Obtaining a Library List (Lib List).
Setup	Opens a dialog box for specifying network and display settings. For complete information, see Setting Up VLAMcommander.

Field	Value
About	Displays copyright and software version information. For more information, see About VLAMcommander.
Help	Displays VLAMcommander system help. For more information, seeHelp.
Close	Exits VLAMcommander. For more information, see Closing VLAMcommander.

Loading a Library Member (Load)

The Load command puts data into a library. Remember that a library is organized into members, and that each member has a set of identifiers and one or more data components called chains. For more information, see Library Member Basics. Thus, data loaded into a library will be part of a member, a member that you identify by specifying the member name and revision level.

If the member name and revision level you specify is unique to the library, VLAMcommander creates a new library member and loads the data into a chain of the new member.

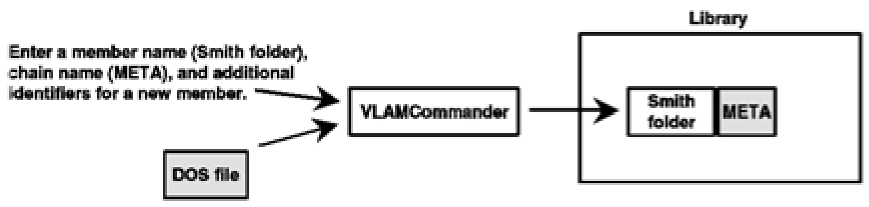

Enter a member name (Smith folder), chain name (META), and additional identifiers for a new member.
DOS file
VLAMCommander
Library
Smith folder
META

Figure 13: Loading a new member

If you use a member name and revision level that matches a member already in the library, the library, the data is loaded to a chain of that member.

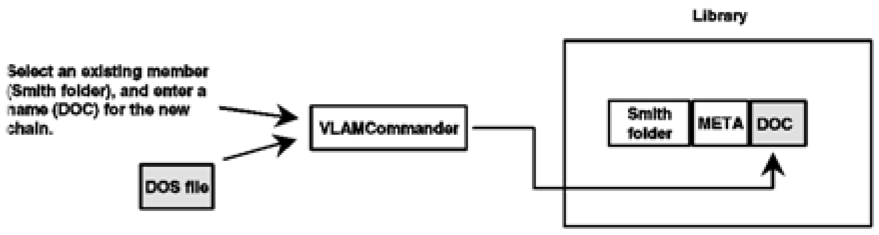
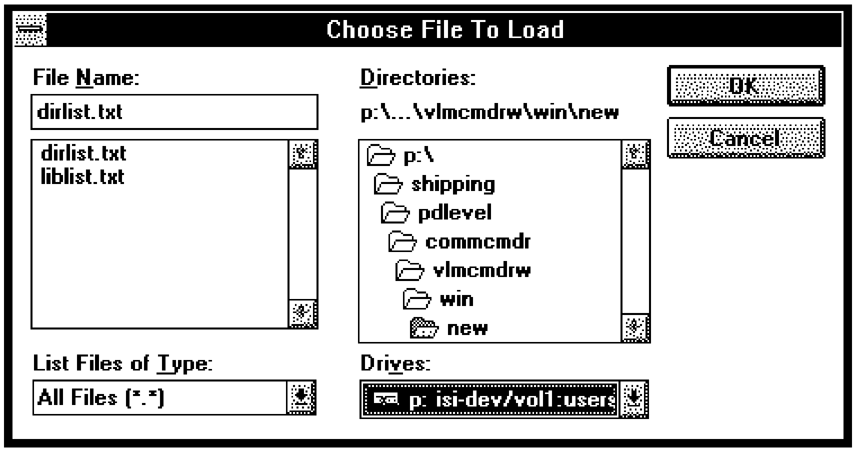

Select an existing member (Smith folder), and enter a name (DOC) for the new chain.
DOS file
VLAMCommander
Library
Smith folder
META
DOC

Figure 14: Revising an existing member

To Load a Library Member

1. Select the target library. Use the Library drop down list box to select the library to which the data will be loaded. The libraries displayed in the drop down list box reflect the entries in the Commcommander VLAM Driver/MVS Settings File. For more information, see Obtaining a Library List (Lib List).

2. Enter the name and revision level of the target member. The member name may be 1 to 32 characters long. Member names are case sensitive: *AB-1*, *Ab-1*, and *ab-1* are regarded as three different member names. The revision level can be any whole number from 0 to 99999. If you enter zero, VLAMcommander loads the data to the version of the member with the highest revision level. For example, if there are two *Smith folder* members, one at revision level 1 and the other at revision level 2, and you specify *Smith folder*, revision level *0*, VLAMcommander loads the data to *Smith folder*, revision level *2*. Specifying revision level zero can free you from having to know the highest revision level among the different versions of a given member.
3. Ensure the text boxes for DTN, description, and effective date have the desired values. If they don't, enter the correct values.
   - Valid DTN values are whole numbers from 0 to 99999. Zero is the default value. For the glossary definition of DTN, see DTN.
   - The description may be any string of characters up to 36 characters long. The default value is no characters.
   - The effective date is in mm/dd/yyyy format. The default value is 00/00/0000. For a glossary definition of effective date, see effective date.
4. Select or enter a chain name. This is the member chain to which the data is loaded. If you select a chain that is already a part of the target member, the load overwrites the chain with the new data.
5. Ensure the correct setting for the variable blocked check box. If you plan to use the data on the mainframe (a print stream or a font, for example), this box must be checked. If you're sending the data to a host VLAM Library for storage purposes only, don't check the box.
6. Click on the **Load** button. The **Choose File to Load** dialog box displays.



File Name: dirlist.txt
dirlist.txt
liblist.txt
Directories: p:\...\vlmcmdrw\win\new
p:\
shipping
pdlevel
commcmdr
vlmcmdrw
win
new
List Files of Type: All Files (*.*)
Drives: p: isi-dev/vol1:users

7. Select the input DOS file, the file containing the data to be loaded and click **OK**.

   VLAMcommander begins loading the data. The cursor changes to an hourglass and the title bar changes to give you the status of the current load. When the cursor changes back from an hourglass, the load is complete.

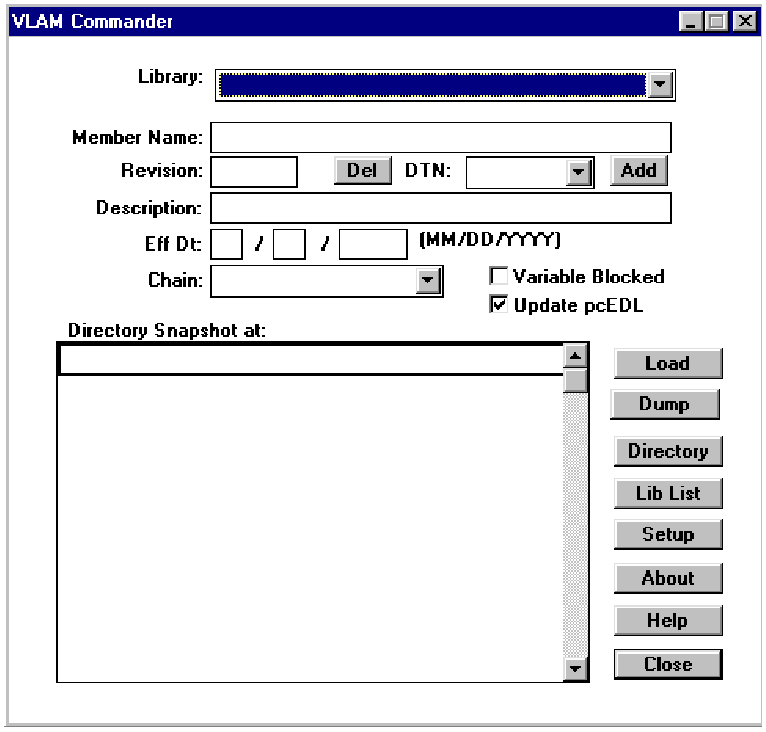
8. (Optional) Click on the **Directory** button to update the member list displayed in the lower portion of the dialog box. Then select the member created or affected by the recent load and check that the identifiers and chain(s) are correct.

To Load a Library Member to pcVLAM or a File-based EDL

1. Edit the ISI.INI file, located in the Windows directory. Under the section name **[VlamCmdr]**, add the following keywords and assign them the desired values:

Keyword	Value
ProgramExit=	The name of the VLAMcommander exit program that allows writing members to an EDL. Depending on the type of target EDL, type one of the following program names: ■ *vlcpcvlm.exe* for a pcVLAM EDL ■ *vlfbvlm.exe* for a file-based EDL
pcEDLLibraryName=	The reference name for the desired EDL (e.g., **VLAMLIB**).

After adding the necessary keywords to the [VlamCmdr] section of the ISI.INI file and starting the VLAMcommander application, you'll notice a new dialog entry located under the **Variable Blocked** entry (e.g., **Update pcEDL**).



2. Indicate your decision about updating the PC-based EDL.

To Revise an Existing Member

You can load to an existing member and overwrite these identifiers:

- Description
- DTN
- Effective date

1. Load a chain to an existing member (load instructions are on the previous page) to make changes to these fields.
2. Enter the new values in the description, DTN, and/or effective date text boxes.
3. Click on the **Load** button.

Dumping a Library Member (Dump)

The dump command copies data from a mainframe VLAM library to a DOS file on your PC. Remember that a library is organized into members and that each member has a set of identifiers and one or more data components called chains. For more information, see Library Member Basics. Thus, the data are taken from a member chain. You must identify the member and chain as well as the DOS file to which the data are written.

These are some ways in which the dump command can be used:

- To print a member, dump the chain that contains the print data stream and route the file to your printer.

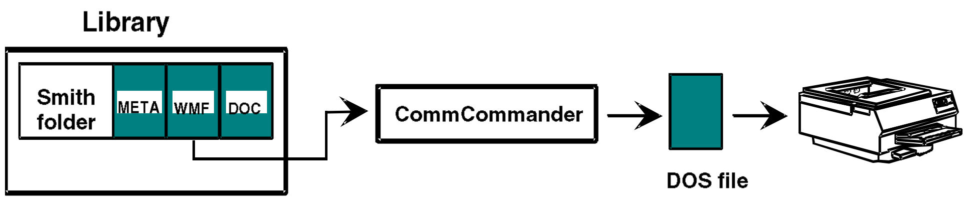


Figure 15: Dumping a chain to the printer

- To store duplicate chain data in a PC VLAM library, dump a chain from the mainframe library and then load the data into your PC library.

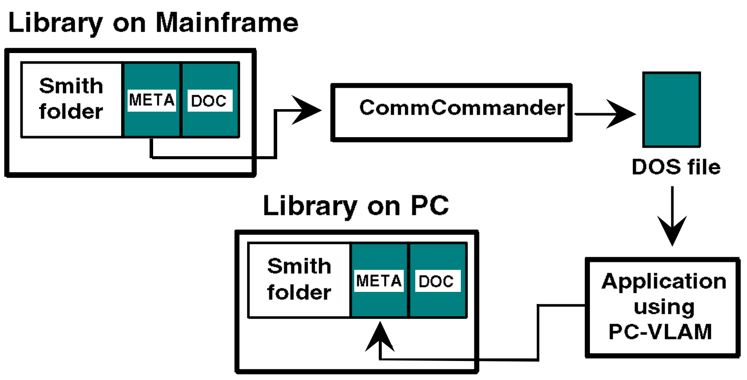


Figure 16: Dumping a chain from a mainframe to a PC library

- You may want to use your mainframe library as the primary depository for all files related to a particular business application. If so, you may want to store word processing files along with their print data stream equivalents. The dump command allows you to retrieve the word processing files from the host library.

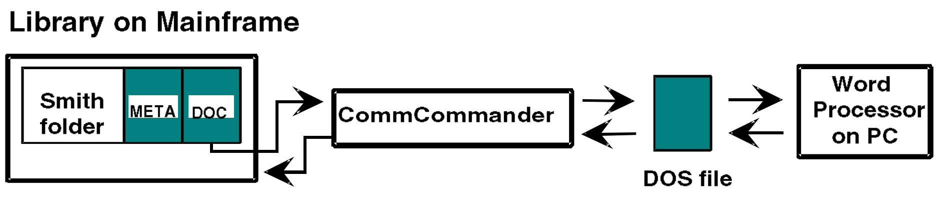
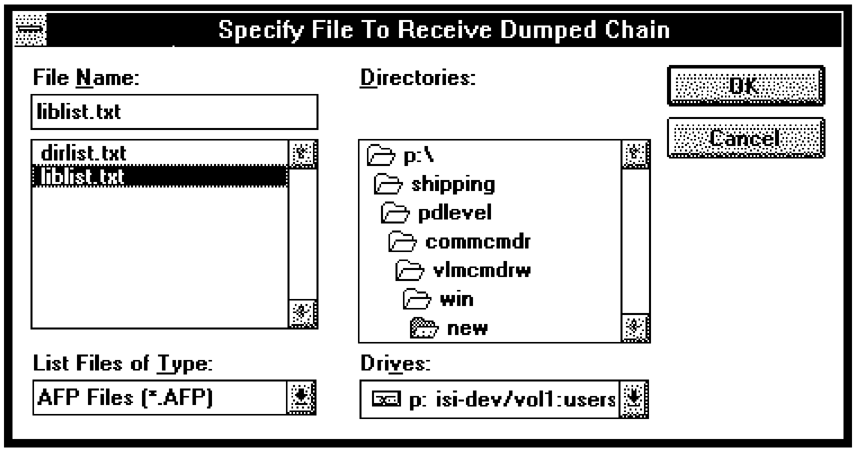


Figure 17: Dumping word processing files

To Dump a Library Member

1. Select the target library. Use the Library drop down list box to select the library from which you want to dump the data. The libraries displayed in the drop down list box reflect the entries in the Commcommander VLAM Driver/MVS Settings File. For more information, see Obtaining a Library List (Lib List).
2. Select the member with the chain to be dumped. There are two ways to select the member:
    - Find the member in the list box and double click on it. After the double click, the text boxes in the upper portion of the dialog display information about the member.
    - Type the member name and revision level in the matching text boxes. If you use zero as the revision level, VLAMcommander dumps from the version of the member with the highest revision level. For example, if there are two *Smith folder* members, one at revision level 1 and the other at revision level 2, and you specify *Smith folder*, revision level *0*, VLAMcommander dumps the data from *Smith folder*, revision level *2*. Specifying revision level zero can free you from having to know the highest revision level among the different versions of a given member.
3. Select the chain. You can type the chain name in the drop down list box or select a chain from the drop down list box list.

4. Click on the **Dump** button. The **Specify File To Receive Dumped Chain** dialog box displays.



File Name: liblist.txt

dirlist.txt
liblist.txt

p:\
shipping
pdlevel
commcmdr
vlmcmdrw
win
new

List Files of Type: AFP Files (*.AFP)

Drives: p: isi-dev/vol1:users

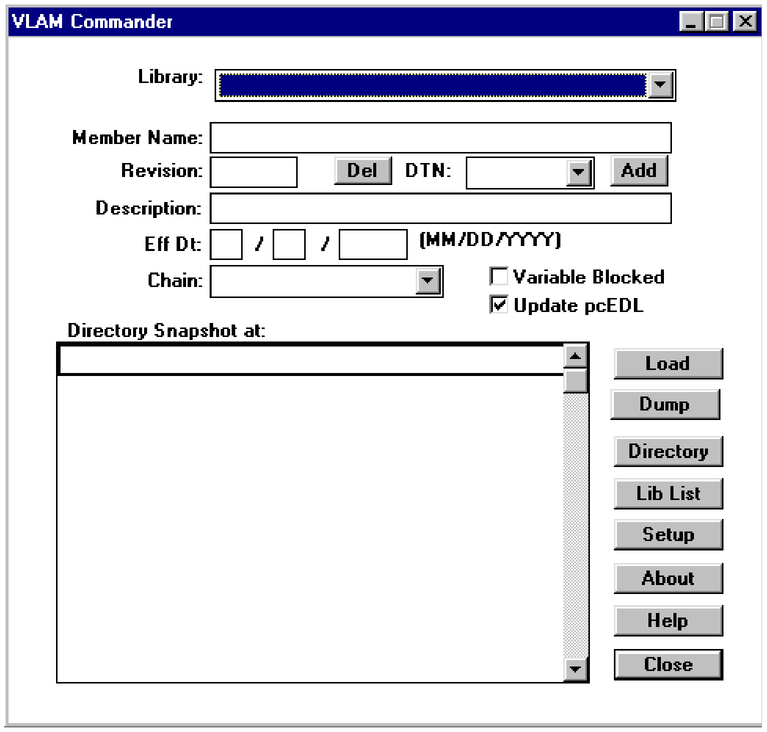
5. Specify the DOS file where you want to dump or copy the member chain. If you specify a file that already exists, VLAMcommander prompts you about overwriting that file. Once you've specified the file, click **OK**. VLAMcommander begins to dump the chain and informs you of its processing status via the dialog box's title bar. When the title bar returns to VLAMcommander, and when the cursor is no longer an hour glass, the dump process is complete.

Viewing Library Members (Directory)

The directory command updates the list of members displayed in the VLAMcommander dialog box. These members are or have been within the active library, the library displayed in the Library text box. Just above the list of members is the text *Directory Snapshot at mm/dd/yyyy hh:mm:ss*. The date and time indicate when the member list was made. If the library has changed since the indicated date and time, the member list might not be accurate.

To View Library Members

Click on the **Directory** button. Any time you make a new library using the Library drop down list box, a directory procedure is performed, just like clicking on the **Directory** button.



Note The name of this command is the same as its counterpart in VLMMAINT, the mainframe VLAM library utility.

Obtaining a Library List (Lib List)

To Obtain a Library List

Click on the **Lib List** button to update the list in the Library drop down list box.

In order to update this list, VLAMcommander compares these libraries with those entered in a mainframe data set called the Commcommander VLAM Driver/MVS Settings File. If there are any differences, the Library drop down list box is changed to reflect the libraries listed in the Commcommander VLAM Driver/MVS Settings file.

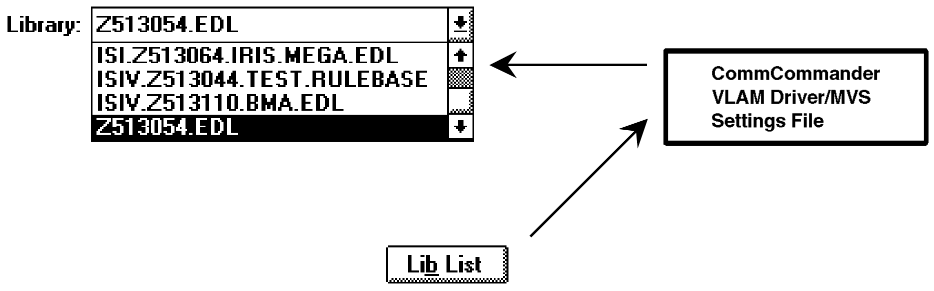

Library:
Z513054.EDL
ISI.Z513064.IRIS.MEGA.EDL
ISIV.Z513044.TEST.RULEBASE
ISIV.Z513110.BMA.EDL
Z513054.EDL
CommCommander VLAM Driver/MVS Settings File
Lib List

Figure 18: Obtaining a library listing

If a new VLAM library is created on the mainframe and referenced in the Commcommander VLAM Driver/MVS Settings File, the new library won't appear in the Library drop down list box until you click on the **Lib List** button. Conversely, if a mainframe Library is deleted, it's listed in the Library drop down list box until you click on the **Lib List** button. This is an example of a Commcommander VLAM Driver/MVS Settings File.

```
********************************* TOP OF DATA ******************************
* ID----VRM---DATE--    TYP-DDN-----DISP-DSN---------------------------
* ..+....1....+....2....+....3....+....4....+....5....+....6....+....7...+
DFCVLM  VvRrMmYYMMDD    DSN ddname1 SHR  data.set.name.1
DFCVLM  VvRrMmYYMMDD    DSN ddname2 SHR  data.set.name.2
DFCVLM  VvRrMmYYMMDD    DSN ddname3 SHR  data.set.name.3
***************************** BOTTOM OF DATA ****************************
```

An example of a settings file is shown above. Note how the drop down list box lists the data set names in the file (beginning in column 42). For comprehensive information about such files, see Settings File Information.

Setting Up VLAMcommander

For more information, see Setting Up VLAMcommander.

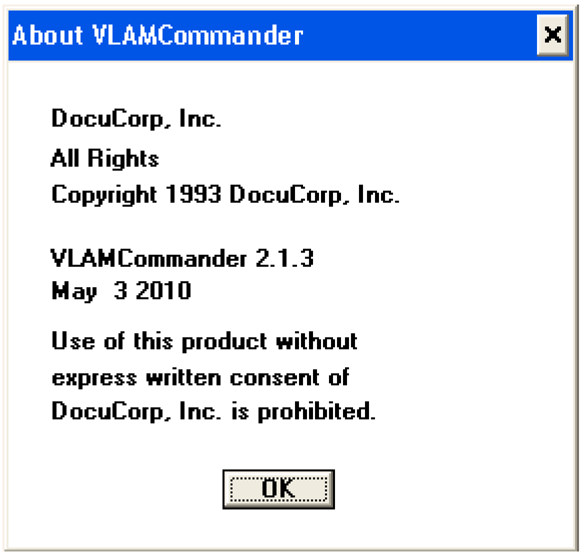
About VLAMcommander

Use the About icon to cisplay version information about VLAMcommander.

To Display the About Box

Click the **About** button to display a dialog box containing a variety of information about the VLAMcommander software you're running. If you should require technical assistance from Docucorp, this dialog box presents information that may be needed by our support team.

The date stamp for your version of VLAMcommander might differ from the one represented here. Please have this date ready when calling the Hotline for customer support.



DocuCorp, Inc.

All Rights

Copyright 1993 DocuCorp, Inc.

VLAMCommander 2.1.3

May 3 2010

Use of this product without express written consent of DocuCorp, Inc. is prohibited.

Help

Use the Help icon to display the contents of the help file.

To Invoke the Help System

Click the **Help** button to invoke the on-line help system for VLAMcommander. The help file contains helpful hints about the use and maintenance of VLAMcommander.

Closing VLAMcommander

Use the Close icon to close VLAMcommander and return to the Windows Desktop.

To Close VLAMcommander

Click on **Close** to close the VLAMcommander dialog box and end the VLAMcommander program.

VLAMcommander for Windows and Word Processor Macros

Many Windows applications, such as Microsoft Word and Corel WordPro, include a macro language that can invoke external Windows applications. VLAMcommander for Windows can be called from a macro. When the macro calls the **ShowVLAMDialog** subroutine, the VLAMcommander dialog is displayed and is active for use.

You may find it convenient to invoke VLAMcommander for Windows from a word processor macro: once you've created or updated a document, you can quickly call VLAMcommander and load the source file or print image to the Library.

Note You can invoke the VLAMcommander dialog box with any macro or program that can call routines in a DLL. This discussion is centered on word processor macros only because such macros are the most likely place from which you'll call VLAMcommander for Windows.

Information Needed by the Calling Macro

The table below shows the information any macro needs for calling VLAMcommander for Windows. Be aware that VLMCMDR.DLL is part of the VLAMcommander product and is in the directory where VLAMcommander is installed.

Parameter	Specification
DLL name:	VLMCMDR.DLL
Subroutine name:	ShowVLAMDialog

To invoke VLAMcommander for Windows from a macro, the directory containing the VLAMcommander .exe and .dll files must be specified in your DOS path. If not, then you must copy these files to your C:\WINDOWS sub-directory so that Microsoft Word can locate them.

Word for Windows Example

You can either customize one of your existing macros or create a new one. If you choose to create a new macro, you can choose any descriptive name (such as “VLMCMDR”). Regardless of your choice, add the following statements to your macro:

```
Declare Sub ShowVLAMDialog Lib "VLMCMDR.DLL"()
Sub MAIN
        Call ShowVLAMDialog()
End Sub
```

This macro is written in Word Basic, the macro language for Word for Windows.

The declaration statement identifies the subroutine and the DLL containing the subroutine. You might want to add your new macro to your customized Tool Bar for easy access to VLAMcommander for Windows. For reference information about Word Basic, see the Microsoft publication *Using Word Basic*.

Chapter 10

Using VLAM in a Multiple CPU Environment

Overview

In a multiple CPU environment, processing takes place on several CPUs which share common resources including DASD and data sets. The operating system ensures that these resources are allocated properly. However, in some instances additional communication between the application program and the operating system is required to control resource utilization. Docucorp software relies on the operating system to maintain information about the use of resources through system Enqueue/ Dequeue macros and specially defined Queue names.

In a multiple CPU environment, the global resource serialization package is responsible for communicating information about system Enqueues (ENQ) and Dequeues (DEQ) from one CPU to another. This is done through the use of an inclusion and exclusion list. If a Queue name is on the **inclusion list**, when an ENQ or DEQ is issued for that queue name the effects of the ENQ or DEQ will be communicated to all of the CPUs on the system.

If a Queue name is on the **exclusion list**, when an ENQ or DEQ is issued for that queue name, the effects of the ENQ or DEQ will take effect **only on the CPU under which processing is currently taking place**.

Some global resource serialization packages treat all queue names as part of the inclusion list unless they are explicitly placed in the exclusion list. Other packages treat all queue names as part of the exclusion list unless they are explicitly placed in the inclusion list. Since all serialization packages vary in the defaults used, **you must verify** that the queue names used by Docucorp will be treated as part of the **inclusion list**.

SYSTEM ENQUEUE/DEQUEUES

Tthe following queue names are used:

ISISGNON	Docuwise
DVSSGNON	Docuvise
HIAMEXTQ	HIAM, Docuwise, Docuvise
HIAMINTQ	HIAM, Docuwise, Docuvise
*** VLAM-MDB**	VLAM, Docuwise, Docuvise, Documerge,
	Commcommander (Inbound) MVS *Multi-Tasking Environment*

You must verify that these queue names will be treated as part of the inclusion list by your global resource serialization software.

* **VLAM-MDB** - Resource name **DFCSERV** in a Commcommander Multi-Tasking Environment is used to control the creation, and updating of the Message, Log, and Trace datasets. It's recommended that this resource name is added to the VLAM-MDB queue, ***exclusion list***, when logging and tracing Commcommander transaction throughput (for debug purposes.)

Logging and tracing should only be performed when gathering information for client support to analyze, and returned to normal as soon as the necessary data is captured.

Chapter 11

TCP/IP or ACF/VTAM Environments

TCP/IP OR ACF/VTAM ENVIRONMENTS

In a TCP/IP or ACF/VTAM environment, a Windows application calls Commcommander Server via, TCP/IP sockets, or Microsoft's Network Client.

The Commcommander Server then calls the outbound message handler and passes the commands and data to be sent to MVS. The outbound message handler initiates a conversation with the MVS parties. The messages sent from Commcommander Server are received by the inbound message handler (DFCCOMI), a started task running in mainframe memory. For more information, see Starting Commcommander in a TCP/IP or ACF/VTAM Environment.

The two message handlers remain open, passing commands and data between the PC and MVS programs until one of the programs signals processing is complete. The mainframe's inbound message handler remains running until terminated by the TERMINATE job. For more information, see To Stop Commcommander in a TCP/IP or ACF/VTAM Environment.

Chapter 12

APPC Environment Basics

WHAT IS APPC?

Advanced Program-to-Program Communication (APPC) is a set of program-to-program communication services that support cooperative processing. In an APPC environment, programs running under MVS and Windows NT or RS6000 AIX can communicate, transfer data back and forth, and activate processing tasks by programs within the same or different operating systems. If you aren't in an APPC environment, you should be using the ACF/VTAM version of Commcommander. For more information, see Different Versions of Commcommander.

One way VLAMcommander takes advantage of APPC is by downloading data streams and resources from the mainframe. APPC makes it possible for these data to seamlessly cross the MVS and Windows NT or RS6000 AIX operating systems.

Logical Unit and LU 6.2

APPC exists within IBM's **Systems Network Architecture (SNA)**. The SNA architecture provides formats and protocols that define a variety of physical and logical SNA components. One such logical component, called the **logical unit (LU)**, is responsible for handling communication between end users and for providing each end user with access to the SNA network. SNA defines different types of logical units to meet the needs of specific end users, whether the end user is an application program, a stand-alone terminal, or a terminal and an operator. **LU 6.2** is the type of logical unit that is specifically designed to handle communications between application programs.

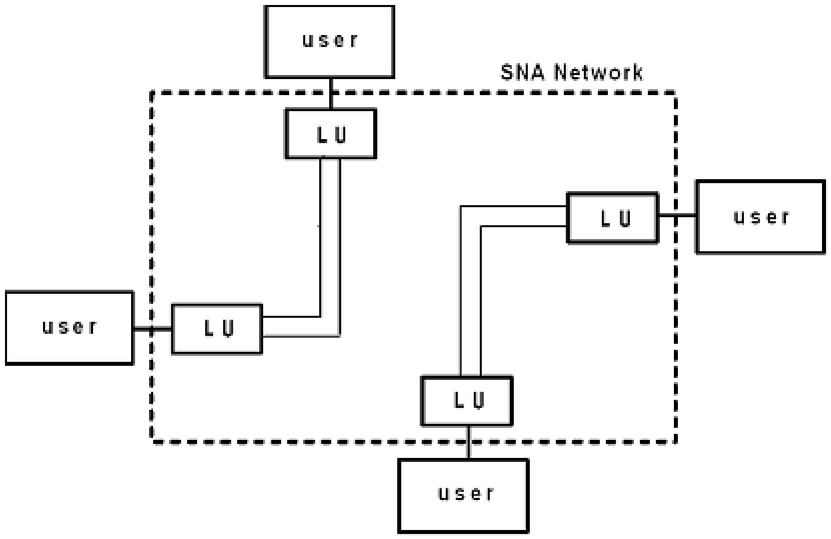


Figure 19: IBM's SNA Architecture

A typical SNA network consists of different types of processors or nodes. For example, there might be a System/370 or System/390 host node and dozens of Windows NT or RS6000 AIX workstation nodes. Using LU 6.2, an APPC application running on one of these processors can communicate with a remote APPC application running on another processor, regardless of the type of processor on which the remote application is running.

Communication between applications on diverse processors is made possible by **Virtual Telecommunications Access Method (VTAM)**. VTAM and APPC/MVS are implementations of SNA architecture that direct data between programs and devices.

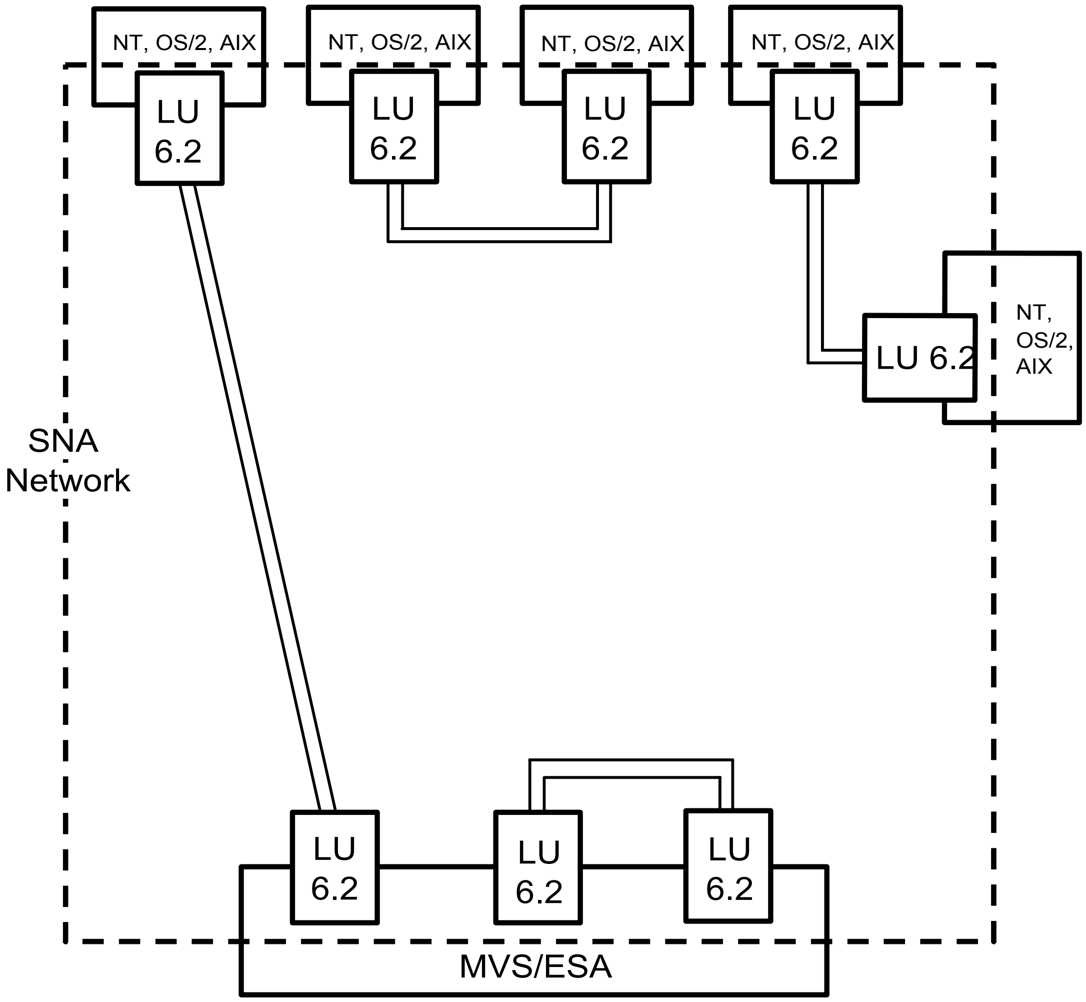


Figure 20: Different nodes within SNA

Each LU within an SNA network must be defined to VTAM. These definitions contain information about security, job scheduling (for MVS LUs), and the APPC programs associated with that LU.

Local and partner LUs are terms used in the context LU-to-LU communications. From an MVS system's point of view, LUs defined to the MVS system are local LUs and LUs defined to remote systems (e.g., NT or AIX systems) are partner LUs. From the remote system's point of view, the names are reversed: an NT- or AIX-defined LU is local and an MVS-defined LU is remote. If both LUs are on the same system, the LU that initiated communication is the local LU and the LU receiving the communication is the partner LU.

Sessions, Conversations, and Transaction

PROGRAMS

In an APPC context, the actual program-to-program communication is a **conversation**. The conversation is supported by a **session**, a logical connection established or bound between two LUs of the same type. A session acts as a conduit through which data moves between the pair of LUs.

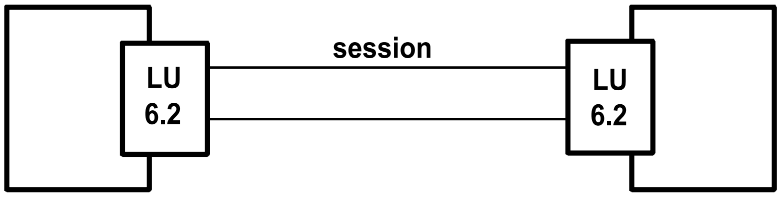


Figure 21: Session example

A session can support only one conversation at a time, but one session can support many conversations in sequence. Because sessions are reused by multiple conversations, a session is a long-lived connection compared to a conversation.

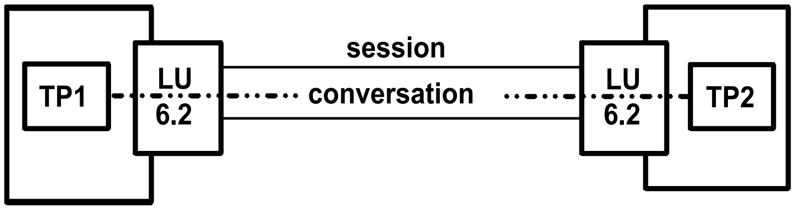


Figure 22: Sessions and Conversations

The session and its two LU 6.2 nodes transmit information for Transaction Programs. A **Transaction Program (TP)** is an application program that uses APPC communication calls. A TP on one system can communicate with a TP on another system to access resources on both systems. The data being transferred back and forth might be status queries, data records, or processing commands.

Conversations between TPs are analogous to telephone conversations: one TP calls the other and they *converse*, one TP *talking* at a time, until one TP ends the conversation. The TP that starts the conversation is sometimes referred to as the **outbound TP**. The one that responds is the **inbound TP**.

DOCUCORP TRANSACTION PROGRAMS

Docucorp products use a variety of transaction programs (TPs). Some of the TPs handle communications across operating systems while others have partner TPs within the same operating system.

Communication Between NT or UNIX and MVS

Docucorp products use four transaction programs to handle communication between processes running under NT or UNIX and MVS. The TPs can be characterized by the operating system under which they run and their outbound or inbound nature. An outbound TP initiates a conversation with and inbound TP. Once the conversation has been established, messages can flow both ways between the two.

	NT or UNIX	MVS
Outbound Message Handlers	`COMCMDRO`	`DFCCOMO`
Inbound Message Handlers	`COMCMDRI`	`DFCCOMI`

Since these TPs have been designed to communicate across operating systems, COMCMDRO communicates with DFCCOMI, exclusively. Similarly, DFCCOMO talks to no other TP than COMCMDRI.

Transaction Programs in Commcommander

First, a Windows application (e.g., VLAMcommander for Windows) calls Commcommander Server via Microsoft's Network Client, or TCP/IP sockets.

Commcommander Server then calls the outbound message handler and passes the commands and data to be sent to MVS. The outbound message handler initiates a conversation with the MVS parties. The messages sent from Windows NT or AIX are received by an APPC transaction scheduler running under MVS.

The messages from the server identify the desired MVS TP (the inbound message handler) to the scheduler, which starts the message handler. The inbound message handler, based on information sent from Commcommander Server, invokes Commcommander Server/MVS TP, passing all of the necessary parameters and data.

The two message handlers remain open, passing commands and data between the NT or and MVS programs until one of the programs signals processing is complete and the TPs are to terminate the conversation.

Chapter 13

System Messages

System Messages

APPC/MVS Error Messages

Consult your IBM documentation for explanations of APPC/MVS error messages:

“**Appendix B. Explanation of Return Codes**”
from *Application Development: Writing Transaction Programs for APPC/MVS*
IBM Doc: GC28-1121-2

ACF/VTAM Error Messages

Consult your IBM documentation for explanations of ACF/VTAM error messages:

“**Appendix A: Return Codes**”
from *Programming for LU 6.2*
IBM Doc: SC31-6410-1

VTAM Error Messages

Consult your IBM documentation for explanations of VTAM error messages:

“**Open Macro Return Codes**”
from *VTAM Programming*
IBM Doc: SC31-6409-1

TCP/IP Error Messages

Consult your IBM documentation for explanations of MVS TCP/IP error messages:

“**IBM TCP/IP for MVS: Messages and Codes**”
Volume I (EZA Messages)
IBM Doc: SC31-7132-04

Note UNIX TCP/IP errors can be obtained from the manual pages on your server machine. Also, you can use the Info Explorer Window Interface to get information on return codes. If you use Info Explorer, see “Sockets Overview and Understanding Socket Data Transfer” in “AIX Version 4 Communications Programming Concepts.”

Windows NT, UNIX, and IBM MVS Return Code tables are provided at the end of this section for your convenience.

VLAM Error Messages

Consult your Docucorp documentation for explanations of VLAM error messages:

“**Appendix A. Messages**”
from *VLAM User Guide and Reference*

Commcommander Error Messages

Message Format

All returned error messages appear in the following formats:

Mainframe:

```
SSSPPPnnnnc – hhhhhhhh hh:mm:ss mmmmmmmm
Message_Text
```

PC:

```
SSSPPPnnnnc Pid hh:mm:ss mmmmmmmm Std_Text
Tid Message_Text
```

where:

Code	Meaning	
SSS	The system identifier:	
	CCM	Commcommander Mainframe
	CCP	Commcommander PC
PPP	The application identifier:	
	SRV	Commcommander Server modules
	JES	Commcommander JES Driver modules
	VLM	Commcommander VLAM Driver modules
	VRF	Commcommander VRF Driver modules
	NOTE: The Module Identification space is not included in the error message number as contained within this appendix. It is included in the displayed message to identify the application issuing the message. As such, it is possible for multiple modules to issue the identical message, differing only in the module identification portion of the message identifier and the mmmmmmm portion of the message text (see definition for mmmmmmmm).	
nnnn	The unique message number	
c	The message severity of code. The severity associated with a given job is dependent upon the job's return code (RC). Commcommander server and its drivers will set the MVS system return code to the highest return code value the job.	

Code	Meaning	
	I	**Informational**. The message is for user information only. Generally, no user action is required. RC=0.
	W	**Warning**. An error has occurred, but the program will take corrective action. Processing continues. An error will be flagged as W when there area several possible corrective actions but only one is likely to be what the user intended. RC=4.
	C	**Conditional Error**. An error has occurred. Corrective action is taken but the action is probably not what the user intended. Processing continues. RC=8.
	E	**Error**. A non-correctable error has occurred. no corrective action may be assumed. The condition causing the error is bypassed. Processing continues, but processing might be terminated later as a result of this error. RC=12.
	F or T	**Fatal**. An unrecoverable error has occurred. Processing stops. RC=16.
	P	**Program**. A program logic error has occurred. Mainframe applications results in a job cancellation with the ABEND code U0020. RC=20.
x	The original output file id (L)og, (M)essage, (T)race.	
hhhhhhhh	The conversation id displayed in hexadecimal.	
Pid	Process Id displayed in hexadecimal	
hh:mm:ss	The time the message was written.	
mmmmmmmm	The name of the module issuing the message.	
Std_Text	Fixed portion of message containing Commcommander standard error text.	
Tid	Thread id displayed in hexadecimal	
Message_Text	Note that some messages contain variable text. Variable values are most often return/reason codes, ddnames, command names.	

MESSAGES BY MESSAGE NUMBER

Mainframe

(The meaning of the return code values are found in the books listed at the beginning of Appendix C.)

CCM1030EERROR SETTING SYSTEM TIMER. RETURN=return code.

Probable Cause: An error was encountered attempting to obtain the current date and time. This error may be issued if the system TIME macro fails, or an invalid parameter was passed to the module which fetches the current system date and time.

Action: Review the system console log at time of error for any system messages which may have displayed pertaining to the system time, if found, take corrective actions referencing the system console information. If there are no system log messages then contact Docucorp with the message number and associated return code.

CCM1040CALLOCATION FAILURE ON DDN: ddname, DSN: dsname.

Probable Cause: An error was encountered attempting to dynamically allocate the ddname. This error usually means that the ddname is missing or invalid.

Action: Verify that the requested ddname is present in the JCL, or that the dsname is a cataloged dataset.

CCM1050EERROR OPENING file description FILE. DDN: ddname, RETURN=return code.

Probable Cause: An error was encountered while trying to open the named file. The return code from the I/O module may provide additional information.

Action: Verify that the DCB information for the specified file was coded correctly. If the file is an input file, be sure that it exists and was allocated correctly.

CCM1060CERROR CLOSING file description FILE. DDN: ddname, RETURN=return code.

Probable Cause: An error was encountered while trying to close the named file. The return code from the I/O module may provide additional information.

Action: Verify the dataset definition in your JCL.

CCM1070EERROR READING FROM file description FILE. DDN: ddname, RETURN=return code.

Probable Cause: An error was encountered while trying to read from the named file. The return code from the I/O module may provide additional information.

Action: This error condition may be encountered if the DCB information of the file is incorrect. Verify that the correct file was specified on the DD card and in SYSIN.

CCM1080EERROR WRITING TO file description FILE. DDN: ddname, RETURN=return code.

Probable Cause: An error was encountered while trying to write to the named file. The return code from the I/O module may provide additional information.

Action: This error condition may be encountered if the DCB information of the file is incorrect. Verify that the correct file was specified on the DD card and in SYSIN.

CCM2010FERROR OPENING ACB. APPLID: vtam lu name, ERROR CODE=return code.

Probable Cause: An error was encountered while trying to open the specified ACB Application Id. The error code indicates the reason for the failure.

Action: Verify that the specified Application Id. is a valid VTAM LU name defined for use with this application. If valid, refer to “Open Macro Return Codes” in “VTAM Programming” IBM Doc: SC31-6409-1 for more information on the error code.

CCM2020FERROR CLOSING ACB. APPLID: vtam lu name, ERROR CODE=return code.

Probable Cause: An error was encountered while trying to open the specified ACB Application Id. The error code indicates the reason for the failure.

Action: This error condition may be encountered if the ACB should become invalid sometime during application processing. Refer to “VTAM Programming” IBM Doc: SC31-6409-1 for more information on the error code.

CCM2030CMESSAGE FROM APPC/VTAM COMMAND: command acronym, RETURN=return code.

Probable Cause: An error was encountered while trying to perform the specified command. The return code indicates the reason for the failure.

Action: Refer to “Application Development: Writing Transaction Programs for APPC/MVS” IBM Doc: GC28-1121-2 for more information on the error code.

CCM2035EMESSAGE FROM APPC/VTAM COMMAND: command acronym, RETURN=return code.

Probable Cause: An error was encountered while trying to perform the specified command. The return code indicates the reason for the failure.

Action: Refer to “Application Development: Writing Transaction Programs for APPC/MVS” IBM Doc: GC28-1121-2 for more information on the error code.

CCM2040CMESSAGE FROM ACF/VTAM COMMAND: command acronym additional information.

GENERAL RC=return code, RECOVERY ACTION=vtam recovery action code, SENSE CODE=vtam sense code.

RETURN=vtam rpl return code, FEEDBACK=vtam rpl feedback code, PRIMARY RC=appc primary return code, SECONDARY RC=appc secondary return code.

Probable Cause: An error was encountered while trying to perform the specified command. VTAM provides several codes to determine the reason for the failure.

- The general return code indicates whether or no the command was accepted.
- The recovery action code indicates the type of condition. The condition type will be displayed as additional information.
- The VTAM RPL return code indicates abnormal completion of the command, if not zero.
- The VTAM RPL feedback code indicates the type of condition resulting from a non-zero RPL return code.
- The sense code provides more information about the specific error. This code is displayed as a 4 byte hex field with byte 1 indicating the code category, byte 2 the modifier, and bytes 3 and 4 indicating specific information or user defined data.

Action: Using the information provided, refer to “VTAM Programming for LU 6.2” IBM Doc: SC31-6410-1 for more information on determining the actual cause of the error. Note: The official source for sense code Information is “SNA Format and Protocol Reference Manual: Architectural Logic” IBM Doc: SC30-3112-2. However, you can find the sense code information in “VTAM Messages and Codes” IBM Doc: SC31-6433-1.

CCM2045EMESSAGE FROM ACF/VTAM COMMAND: command acronym additional information.

GENERAL RC=return code, RECOVERY ACTION=vtam recovery action code, SENSE CODE=vtam sense code.

RETURN=vtam rpl return code, FEEDBACK=vtam rpl feedback code, PRIMARY RC=appc primary return code, SECONDARY RC=appc secondary return code.

Probable Cause: An error was encountered while trying to perform the specified command. VTAM provides several codes to determine the reason for the failure.

- The general return code indicates whether or no the command was accepted.
- The recovery action code indicates the type of condition. The condition type will be displayed as additional information.
- The VTAM RPL return code indicates abnormal completion of the command, if not zero.
- The VTAM RPL feedback code indicates the type of condition resulting from a non-zero RPL return code.
- The sense code provides more information about the specific error. This code is displayed as a 4 byte hex field with byte 1 indicating the code category, byte 2 the modifier, and bytes 3 and 4 indicating specific information or user defined data.

Action: Using the information provided, refer to “VTAM Programming for LU 6.2” IBM Doc: SC31-6410-1 for more information on determining the actual cause of the error. Note: The official source for sense code Information is “SNA Format and Protocol Reference Manual: Architectural Logic” IBM Doc: SC30-3112-2. However, you can find the sense code information in “VTAM Messages and Codes” IBM Doc: SC31-6433-1.

CCM2050IBASE LU USED FOR THIS TASK: vtam lu name.

Probable Cause: This message provides information about the Base LU being used for the current VTAM session.

Action: No action is required for this message.

CCM2060IPARTNER LU: vtam lu name, MODE: vtam mode name.

Probable Cause: This message provides information about the current conversation in-progress.

Action: No action is required for this message.

CCM2070EERROR RECEIVED FROM PARTNER LU vtam lu name. RETURN=return code, REASON=reason code.

Probable Cause: The partner lu has sent an error control block to the base lu indicating that an error has occurred, and processing should be terminated. The return and reason simply indicates that this condition will terminate the base lu's processing.

Action: Check the partner lu's message log for the possible cause of the error and take the appropriate actions.

CCM3000Iinformation_text.

Probable Cause: This message may be produced any time during a conversation. It's intended to provide additional information pertaining to the conversation. This message is informational only.

Action: If error messages precede this message, take corrective actions before establishing your next session.

CCM3010FPERMANENT PROCESSING ERROR RECEIVED FROM PROTOCOL SUB PROGRAM: module name. RETURN=return code.

Probable Cause: A deadly error was detected by the protocol API sub-program, processing will be terminated. The error causing the problem is reflected in the message log prior to this message. The return and reason code simply indicates that this error occurred.

Action: Use the error messages produced prior to this one to determine the cause of the error.

CCM3020EINVALID REQUEST FROM CALLING PROGRAM. REQUEST CODE: request code.

Probable Cause: An invalid request code was passed to the program issuing this message, resulting in a rejection of the request (request not processed).

Action: Contact Docucorp's Hotline support services.

CCM3030WERROR FROM PROGRAM: module name. RETURN=return code, REASON=reason code.

Probable Cause: The named module encountered an error during is processing. The return/reason codes indicates the cause of the error. Note: The named module produced messages on in the message log file prior to this one.

Action: Use messages produced via named module (prior to this message) to determine the cause of this error.

CCM3040EINCORRECT DFCCTL RECORD ID. RECORD ID: identifier.

Probable Cause: The record read in from the control file is not of the correct format. The record id: reflects the portion of the current record used for identification.

Action: Correct the control file record in error, and verify that all control cards are valid, before running the job again. Refer to "Setting Up Commcommander Server/ MVS" for more information on the communication initialization file.

CCM3050EINVALID DFCCTL RECORD COMMAND. COMMAND: command acronym.

Probable Cause: The command specified in the control file is invalid.

Action: Correct the command acronym in the dataset referenced via DFCCTL ddname in the JCL, and verify that all control cards are valid, before running the job again. Refer to “Setting Up Commcommander Server/MVS” for more information on the communication initialization file.

CCM3060CINVALID INPUT description CONTROL RECORD, BYPASSED AND PROCESSING CONTINUES WITH NEXT RECORD.

Probable Cause: The record read in from the control file is not of the correct format. The input description describes the type of control file being processed.

Action: Refer to the proper driver documentation for the type of control being processed, for more information on the format of the control file records.

CCM3070CFIRST INPUT RECORD MUST BE record type description.

Probable Cause: This is an application protocol type error. The application issuing the message was expecting a specific input record indicating the beginning of processing. The record type description indicates the type of record expected and the current executing application.

Action: Verify that the application was setup properly. Contact Docucorp's Hotline support services if the problem persist.

CCM3080CPREMATURE END OF FILE ON ddname. RETURN=return code.

Probable Cause: This error occurs when an input file is being processed and the end of the file is found before the end of file indicator is set. The ddname references the dataset being processing at the time of the error. The return code indicates the reason for the error.

Action: Be sure that the end of file code is part of the input data in the dataset being processed, refer to the documentation of the application issuing the message.

CCM6000CDDN: ddname IS ALREADY DEFINED IN THE VLAM TABLE.

Probable Cause: Attempting to add a duplicate ddname to the table of VLAM library names.

Action: Correct the entries in the dataset defined via the VLMLINNI ddname in the JCL, removing all duplicate entries.

CCM6010ETHE SPECIFIED DDN: ddname NOT DEFINED TO VLAM VIA IN-CORE TABLE.

Probable Cause: A VLAM library access request was specified for an non-existing library. All VLAM libraries need for a given session, should be placed in the dataset defined via VLMLIBNI ddname in the JCL, prior to running your job.

Action: Added the desired VLAM library entry to the VLMLIBNI dataset. For APPC/MVS protocol, simply rerun the JOB. For ACF/VTAM, stop and restart the DFCCOMI job step running on MVS.

CCM6020EVLAM IN-CORE TABLE IS FULL. MAX DDN TABLE ENTRIES

= 99. DDN: ddname, WAS NOT ADDED.

Probable Cause: The VLAM driver is limited to 99 VLAM libraries per session. An attempt was made to add a 100th entry.

Action: Remove any VLAM library entries not needed for this session, from the dataset defined via the VLMLIBNI ddname in your JCL.

PC Workstation or Server

(The meaning of the return code values are found in the books listed at the beginning of Appendix C.)

CCP7010EDrive/Directory/File access failure. Additional information

Probable Cause: An error was detected attempting to access a drive, directory, or file. The additional information area reflects the type of error that occurred.

Action: Examine the additional information to determine the necessary corrective action.

CCP8010FServer connection failure. Additional information

Probable Cause: An error was detected attempting to establish a connection to the specified server machine. This error indicates that the requested server can not be located.

Action: Examine the additional information to determine the necessary corrective action. Verify the validity of the server name specified in the control parameters (e.g.,.INI).

CCP8020FNetwork connection failure. Additional information

Probable Cause: An error was detected attempting to establish a connection to a partner node using the selected network protocol. This error indicates that the network connection is invalid.

Action: Examine the additional information to determine the necessary corrective action. Verify that all of the network parameters to establish a session with a partner node are correct (PC and Mainframe).

CCP8030FInvalid input record detected. Additional information

Probable Cause: An error was detected processing the specified input file (wrong record type). This error indicates that the specified input for the driver is not the type expected.

Action: Examine the additional information to determine the necessary corrective action. Verify that the correct input is supplied for the selected driver.

CCP8040FProtocol error detected. Additional information

Probable Cause: An error was detected processing data received from a network partner. The application receiving the data did not receive the proper header/control/data/trailer combination.

Possible additional information values and their corrective actions are listed below:

Conversation acceptance failure: Return Code nnnn.

nnnn	*is the return code from the protocol you selected during the application setup.*

Action: Verify that all of the network parameters to establish a session with a partner node are correct (PC and Mainframe)

Conversation connection failure: Return Code nnnn. Attempting connection to named pipe('aaaaaaaa ') at host('bbbbbbbb ').

nnnn	*is the return code from the protocol you selected during the application setup.*
aaaaaaaa	*is the name of the gate specified at startup to be used for connection to Commcommander Server Outbound.*
bbbbbbbb	*is the name of the host server machine where Commcommander Server Outbound is running.*

Action: Verify that all of the network parameters to establish a session with Commcommander Server Outbound are correct.

Conversation connection failure: Return Code nnnn. Attempting connection using Symbolic Destination ('aaaaaaaa ').

nnnn	*is the return code from the protocol you selected during the application setup.*
aaaaaaaa	*is the symbolic destination name, defined via SNA Services, where the LU definitions for the host server machine are defined.*

Action: Verify that the values in the symbolic destination (i.e. mode, partner lu, tpname) are valid for the desired connection.

Conversation connection failure: Return Code nnnn. Attempting connection to service('aaaaaaaa ') at host('bbbbbbbb ').

nnnn	*is the return code from the protocol you selected during the application setup.*
aaaaaaaa	*is the name of the port specified at startup to be used for connection to Commcommander Server (PC or Mainframe).*
bbbbbbbb	*is the name of the host server machine where Commcommander Server (PC or Mainframe) is running.*

CCP8040FProtocol error detected. Additional information (cont.)

Action: Verify that the partner transaction program is valid for the established conversation.

Conversation terminated by partner: Return Code nnnn.

nnnn	*is the return code from the protocol you selected during the application setup.*

Action:

CPIC environment exit failure: Return Code nnnn.

nnnn	*is the return code from the protocol you selected during the application setup.*

Action: Verify that the SNA Services is up and running.

Protocol error detected. CPIC environment initialization failure: Return Code nnnn.

nnnn	*is the return code from the protocol you selected during the application setup.*

Action: Verify that the SNA Services is up and running

Named pipe creation failure: Return Code nnnn.

nnnn	*is the return code from the protocol you selected during the application setup.*

Action: Verify that the named pipes protocol is enabled on your machine.

Named pipe disconnection failure: Return code: nnnn.

nnnn	*is the return code from the protocol you selected during the application setup.*

Action: Verify that the named pipes protocol is enabled on your machine.

Named pipe receive failure: Return Code nnnn.

nnnn	*is the return code from the protocol you selected during the application setup.*

Action: Verify that the partner application is up and running.

Named pipe send failure: Incomplete transmission: Transmitted('mmmm ')
of('nnnn ').

mmmm	*is the number of bytes transmitted during the last write pipe request.*
nnnn	*is the number of bytes requested for transmission.*

Action: Verify that the partner application is up and running.

Named pipe send failure: Return Code nnnn.

nnnn	*is the return code from the named pipe protocol.*

CCP8040FProtocol error detected. Additional information (cont.)

Action: Verify that the partner application is up and running.

Socket bind failure: Return Code nnnn.

nnnn	*is the return code from the named pipe protocol.*

Action: Verify that there is enough memory on your machine to process run in the TCP/IP environment

Socket creation failure: Return Code nnnn.

nnnn	*is the return code from the named pipe protocol.*

Action: Verify that there is enough memory on your machine to process run in the TCP/IP environment

Protocol error detected. Socket environment exit failure: Return Code nnnn.

nnnn	*is the return code from TCP/IP sockets.*

Action: Verify that there is enough memory on your machine to process run in the TCP/IP environment

Protocol error detected. Socket environment initialization failure: Return Code nnnn.

nnnn	*is the return code from the TCP/IP sockets.*

Action: Verify that there is enough memory on your machine to process run in the TCP/IP environment

Protocol error detected. Socket receive failure data(mmmm): Return Code nnnn.

mmmm	*is the number of bytes received.*
nnnn	*is the return code from the TCP/IP sockets.*

Action: Verify that the partner application is up and running.

Protocol error detected. Socket receive failure full record: totalBytesRead nnnn.

nnnn	*is the number of bytes received.*

Action: Verify that the partner application is up and running.

Protocol error detected. Socket receive failure length(mmmm): Return Code nnnn.

mmmm	*is the number of bytes received.*
nnnn	*is the return code from TCP/IP sockets.*

Action: Verify that the partner application is up and running.

Protocol error detected. Socket send failure data(mmmm): Return Code nnnn.

mmmm	*is the number of bytes sent.*
nnnn	*is the return code from TCP/IP sockets.*

CCP8040FProtocol error detected. Additional information (cont.)

Action: Verify that the partner application is up and running.

Protocol error detected. Socket send failure length(mmmm): Return Code nnnn.

mmmm	*is the number of bytes sent.*
nnnn	*is the return code from TCP/IP sockets.*

Action: Verify that the partner application is up and running

Protocol error detected. Socket send failure: Incomplete data transmission: Transmitted('mmmm ') of('nnnn ').

mmmm	*is the number of bytes sent.*
nnnn	*is the number of bytes requested.*

Action: Verify that the partner application is up and running.

Protocol error detected. Socket send failure: Incomplete length transmission: Transmitted('mmmm ') of('nnnn ').

mmmm	*is the number of bytes sent.*
nnnn	*is the number of bytes requested.*

Action: Verify that the partner application is up and running.

Protocol error detected. Unable to retrieve requested host('aaaaaaaa ') information: Return Code nnnn.

aaaaaaaa	*is the symbolic name of the host machine entry in the TCP/IP host file.*
nnnn	*is the return code from TCP/IP sockets.*

Action: Verify that the symbolic host name exist in the TCP/IP host file.

Protocol error detected. Unable to retrieve requested service('aaaaaaaa ') information: Return Code nnnn.

aaaaaaaa	*is the symbolic name of the service port entry in the TCP/IP services file.*
nnnn	*is the return code from TCP/IP sockets.*

Action: Verify that the symbolic service port name exist in the TCP/IP services file.

CCP8060FServer initialization failure. Additional information

Probable Cause: An error was detected attempting to initialize the server machine for Commcommander processing.

Action: Examine the additional information to determine the necessary corrective action. Verify that the server machine has named pipe support (sockets for UNIX), and the necessary memory requirements to activate this protocol. If you're running on a Windows NT platform and the additional information indicates a returned error number, refer to Windows NT messages in this documentation for the text associated with the message number.

Possible additional information values are listed below. The corrective action for each message is listed above:

failed to create mutually exclusive semaphore: returncode: nnnn

nnnn	*is the return code from the platform specific semaphore.*

Unable to open 'aaaaaaaa — (mode 'bb ')"...

aaaaaaaa	*is the name of the file where the requested action failed.*
bb	*is the type of file access attributes for the requested file.*

Server initialization failure. Failed to create mutually exclusive semaphore.

Server initialization failure. Invalid pipe name host('aaaaaaaa ') return from host.

aaaaaaaa	*is the invalid named pipe returned from the host.*

Server initialization failure. Security descriptor initialization failure: Return Code nnnn.

nnnn	*is the return code from the Windows NT security interface for named pipes.*

Server initialization failure. Security descriptor set failure: Return Code nnnn.

nnnn	*is the return code from the Windows NT security interface for named pipes.*

CCP8080EServer processing failure. Destination buffer size ('mmmm ') < packet size('nnnn ').

Probable Cause: An error was detected while processing a Commcommander server request. Client/Server connection has been interrupted before completion the current conversation. This problem is usually produced when a foreign application request services from a valid Commcommander partner.

Action: Verify that the network communications support (PC and mainframe) is still available. Restart the communication support before retrying the conversation.

mmmm	*is the size of the requester's data receive*

	buffer.
nnnn	*is the size of the sender's data send buffer.*

CCP8085WServer processing failure. Conversation terminated by partner

Probable Cause: The partner has terminated the conversation in an orderly fashion.

Action: None.

CCP8085WServer processing failure. UserID/password missing or invalid

Probable Cause: The UserID/Password combination is invalid or missing (UNIX only).

Action: Supply the necessary MVS userid and password in the COMCMDR.INI file.

CCP9010EApplication initialization failure. Additional information

Probable Cause: An error was detected attempting to initialize the current driver application.

Action: Examine the .LOG file and take correct actions on all messages prior to this one. If the additional information indicates return/reason, then use the reason code when looking for the cause of the failure.

CCP9020EApplication processing failure. Additional information

Probable Cause: An error was detected during the processing of a Commcommander driver.

Action: Examine the .LOG file and take correct actions on all messages prior to this one. If the additional information indicates return/reason, then use the reason code when looking for the cause of the failure.

CCP9030ECommcommander Gate name is missing or invalid. Additional information

Probable Cause: An error was detected while validating the specified gate name. Either the 'CommCommanderGate=' parameter is missing or contains a null value.

Action: Supply the CommCommanderGate= parameter with a valid value, and restart the driver application.

CCP9040WDelete after upload. Additional information

Probable Cause: A request to delete files after uploading was specified in the .INI file.

Action: Respond to the console prompt (Yes/No). Processing continues.

CCP9050EQueue Type is invalid. Additional information

Probable Cause: An error was detected while validating the specified queue type.

Action: Supply the CommCommanderGate= parameter with a valid value, and restart the driver application. See the associated driver documentation for the correct values to this parameter.

CCP9055EQueue is unavailable. Additional information

Probable Cause: An error was detected attempting to locate the specified queue on the selected server.

Action: Verify that the specified queue is running on the selected server.

CCP9060EQueue Server is unavailable. Additional information

Probable Cause: An error was detected attempting to connect to the specified queue server.

Action: Verify that the specified server machine is active.

CCP9070CSkipping reserved file extension. Additional information

Probable Cause: An error was detected while processing the specified input file. The extension of the selected file can not be processed.

Action: None.

CCP9080CInvalid argument detected. Additional information

Probable Cause: An error was detected while validating the initial arguments of the current driver. The required arguments for the current driver application are missing of invalid.

Action: Refer to the associated driver documentation for the correct argument types.

COMMCOMMANDER RETURN/REASON CODES CROSS REFERENCE

Mainframe

Return — Reason Codes	Message Number	Message Text
0 — 2050	2050	BASE LU USED FOR THIS TASK: vtam lu name.
0 — 2060	2060	PARTNER LU: vtam lu name, MODE: vtam mode name.
0 — 3000	3000	information_text.
4 — 2040	2040	MESSAGE FROM ACF/VTAM COMMAND: command acronym additional information. GENERAL RC=return code, RECOVERY ACTION=vtam recovery action code, SENSE CODE=vtam sense code. RETURN=vtam rpl return code, FEEDBACK=vtam rpl feedback code, PRIMARY RC=appc primary return code, SECONDARY RC=appc secondary return code.
4 — 3030	3030	ERROR FROM PROGRAM: module name. RETURN=return code, REASON=reason code.
8 — 1040	1040	ALLOCATION FAILURE ON DDN: ddname, DSN: dsname.

Return — Reason Codes	Message Number	Message Text
8 — 1060	1060	ERROR CLOSING file description FILE. DDN: ddname, RETURN=return code.
8 — 3060	3060	INVALID INPUT description CONTROL RECORD, BYPASSED AND PROCESSING CONTINUES WITH NEXT RECORD.
8 — 3070	3070	FIRST INPUT RECORD MUST BE record type description.
8 — 3080	3080	PREMATURE END OF FILE ON ddname. RETURN=return code.
8 — 6000	6000	DDN: ddname IS ALREADY DEFINED IN THE VLAM TABLE.
12 — 1030	1030	ERROR SETTING SYSTEM TIMER. RETURN=return code.
12 — 1050	1050	ERROR OPENING file description FILE. DDN: ddname, RETURN=return code.
12 — 1070	1070	ERROR READING FROM file description FILE. DDN: ddname, RETURN=return code.
12 — 1080	1080	ERROR WRITING TO file description FILE. DDN: ddname, RETURN=return code.
12 — 2030	2030	MESSAGE FROM PROTOCOL COMMAND: command acronym, RETURN=return code.
12 — 2035	2035	MESSAGE FROM PROTOCOL COMMAND: command acronym, RETURN=return code.
12 — 2040	2040	MESSAGE FROM ACF/VTAM COMMAND: command acronym additional information. GENERAL RC=return code, RECOVERY ACTION=vtam recovery action code, SENSE CODE=vtam sense code. RETURN=vtam rpl return code, FEEDBACK=vtam rpl feedback code, PRIMARY RC=appc primary return code, SECONDARY RC=appc secondary return code.
12 — 2045	2045	MESSAGE FROM ACF/VTAM COMMAND: command acronym additional information. GENERAL RC=return code, RECOVERY ACTION=vtam recovery action code, SENSE CODE=vtam sense code. RETURN=vtam rpl return code, FEEDBACK=vtam rpl feedback code, PRIMARY RC=appc primary return code, SECONDARY RC=appc secondary return code.
12 — 2070	2070	ERROR RECEIVED FROM PARTNER LU vtam lu name. RETURN=return code, REASON=reason code.
12 — 3020	3020	INVALID REQUEST FROM CALLING PROGRAM. REQUEST CODE: request code.
12 — 3040	3040	INCORRECT DFCCTL RECORD ID. RECORD ID: identifier.
12 — 3050	3050	INVALID DFCCTL RECORD COMMAND. COMMAND: command acronym.
12 — 6010	6010	THE SPECIFIED DDN: ddname NOT DEFINED TO VLAM VIA IN-CORE TABLE.
12 — 6020	6020	VLAM IN-CORE TABLE IS FULL. MAX DDN TABLE ENTRIES = 99. DDN: ddname, WAS NOT ADDED.
16 — 2010	2010	ERROR OPENING ACB. APPLID: vtam lu name, ERROR CODE=return code.
16 — 2020	2020	ERROR CLOSING ACB. APPLID: vtam lu name, ERROR CODE=return code.

Return — Reason Codes	Message Number	Message Text
16 — 3010	3010	PERMANENT PROCESSING ERROR RECEIVED FROM PROTOCOL SUB PROGRAM: module name. RETURN=return code.

PC Workstation or Server

Return — Reason Codes	Message Number	Message Text
4 — 8085	8085	Server processing failure. Additional information
4 — 9040	9040	Delete after upload. Additional information
8 — 9070	9070	Skipping reserved file extension. Additional information
8 — 9070	9080	Invalid argument detected. Additional information
12 — 7010	7010	Drive/Directory/File access failure. Additional information
12 — 8080	8080	Server processing failure. Additional information
12 — 9010	9010	Application initialization failure. Additional information
12 — 9020	9020	Application processing failure. Additional information
12 — 9030	9030	Commcommander Gate name is missing or invalid. Additional information
12 — 9050	9050	Queue Type is invalid. Additional information
12 — 9055	9055	Queue is unavailable. Additional information
12 — 9060	9060	Queue Server is unavailable. Additional information
16 — 8010	8010	Server connection failure. Additional information
16 — 8020	8020	Network connection failure. Additional information
16 — 8030	8030	Invalid input record detected. Additional information
16 — 8040	8040	Protocol error detected. Additional information
16 — 8060	8060	Server initialization failure. Additional information

UNIX TCP/IP Return Codes

Acronyms	Reason Codes	Description
EPERM	1	Operation not permitted
ENOENT	2	No such file or directory
ESRCH	3	No such process
EINTR	4	interrupted system call
EIO	5	I/O error
ENXIO	6	No such device or address
E2BIG	7	Arg list too long
ENOEXEC	8	Exec format error
EBADF	9	Bad file descriptor
ECHILD	10	No child processes
EAGAIN	11	Resource temporarily unavailable

Acronyms	Reason Codes	Description
ENOMEM	12	Not enough space
EACCES	13	Permission denied
EFAULT	14	Bad address
ENOTBLK	15	Block device required
EBUSY	16	Resource busy
EEXIST	17	File exists
EXDEV	18	Improper link
ENODEV	19	No such device
ENOTDIR	20	Not a directory
EISDIR	21	Is a directory
EINVAL	22	Invalid argument
ENFILE	23	Too many open files in system
EMFILE	24	Too many open files
ENOTTY	25	Inappropriate I/O control operation
ETXTBSY	26	Text file busy
EFBIG	27	File too large
ENOSPC	28	No space left on device
ESPIPE	29	Invalid seek
EROFS	30	Read only file system
EMLINK	31	Too many links
EPIPE	32	Broken pipe
EDOM	33	Domain error within math function
ERANGE	34	Result too large
ENOMSG	35	No message of desired type
EIDRM	36	Identifier removed
ECHRNG	37	Channel number out of range
EL2NSYNC	38	Level 2 not synchronized
EL3HLT	39	Level 3 halted
EL3RST	40	Level 3 reset
ELNRNG	41	Link number out of range
EUNATCH	42	Protocol driver not attached
ENOCSI	43	No CSI structure available
EL2HLT	44	Level 2 halted
EDEADLK	45	Resource deadlock avoided
ENOTREADY	46	Device not ready
EWRPROTECT	47	Write-protected media
EFORMAT	48	Unformatted media
ENOLCK	49	No locks available
ENOCONNECT	50	no connection

Acronyms	Reason Codes	Description
ESTALE	52	no filesystem
EDIST	53	old, currently unused UNIX errno
EWOULDBLOCK	EAGAIN	Operation would block
EWOULDBLOCK	54	Operation would block
EINPROGRESS	55	Operation now in progress
EALREADY	56	Operation already in progress
ENOTSOCK	57	Socket operation on non-socket
EDESTADDRREQ	58	Destination address required
EDESTADDREQ	EDESTADDRREQ	Destination address required
EMSGSIZE	59	Message too long
EPROTOTYPE	60	Protocol wrong type for socket
ENOPROTOOPT	61	Protocol not available
EPROTONOSUPPORT	62	Protocol not supported
ESOCKTNOSUPPORT	63	Socket type not supported
EOPNOTSUPP	64	Operation not supported on socket
EPFNOSUPPORT	65	Protocol family not supported
EAFNOSUPPORT	66	Address family not supported by protocol family
EADDRINUSE	67	Address already in use
EADDRNOTAVAIL	68	Can't assign requested address
ENETDOWN	69	Network is down
ENETUNREACH	70	Network is unreachable
ENETRESET	71	Network dropped connection on reset
ECONNABORTED	72	Software caused connection abort
ECONNRESET	73	Connection reset by peer
ENOBUFS	74	No buffer space available
EISCONN	75	Socket is already connected
ENOTCONN	76	Socket is not connected
ESHUTDOWN	77	Can't send after socket shutdown
ETIMEDOUT	78	Connection timed out (Unable to validate partner node)
ECONNREFUSED	79	Connection refused (Partner node not running)
EHOSTDOWN	80	Host is down
EHOSTUNREACH	81	No route to host
ERESTART	82	restart the system call
EPROCLIM	83	Too many processes
EUSERS	84	Too many users
ELOOP	85	Too many levels of symbolic links
ENAMETOOLONG	86	File name too long
ENOTEMPTY	EEXIST	Directory not empty

Acronyms	Reason Codes	Description
ENOTEMPTY	87	Directory not empty
EDQUOT	88	Disc quota exceeded
EREMOTE	93	Item is not local to host
ENOSYS	109	Function not implemented
EMEDIA	110	media surface error
ESOFT	111	I/O completed, but needs relocation
ENOATTR	112	no attribute found
ESAD	113	security authentication denied
ENOTRUST	114	not a trusted program
ETOOMANYREFS	115	Too many references: can't splice
EILSEQ	116	Invalid wide character
ECANCELED	117	asynchronous i/o canceled
ENOSR	118	temp out of streams resources
ETIME	119	I_STR ioctl timed out
EBADMSG	120	wrong message type at stream head
EPROTO	121	STREAMS protocol error
ENODATA	122	no message ready at stream head
ENOSTR	123	fd is not a stream
ECLONEME	ERESTART	this is the way we clone a stream...
ENOTSUP	124	POSIX threads unsupported value
EMULTIHOP	125	multihop is not allowed
ENOLINK	126	the link has been severed
EOVERFLOW	127	value too large to be stored in data type

MVS TCP/IP Return Codes

Return	Socket Codes	Type	Description
1	EPERM	All	Permission is denied. No owner exists.
2	ENOENT	All	The data set or directory was not found.
3	ESRCH	All	The process was not found.
4	EINTR	All	A system call was interrupted.
5	EIO	All	An I/O error occurred.
6	ENXIO	All	The device or driver was not found.
7	E2BIG	All	The argument list is too long.
8	ENOEXEC	All	An EXEC format error occurred.
9	EBADF	All	An incorrect socket descriptor was specified.
9	EBADF	Givesocket	The socket has already been given. The socket domain is not AF_INET.
9	EBADF	Select	One of the specified descriptor sets is an incorrect socket descriptor.
9	EBADF	Takesocket	The socket has already been taken.
10	ECHILD	All	There are no children.
11	EAGAIN	All	There are no more processes.
12	ENOMEM	All	There is not enough storage.
13	EACCES	Takesocket	The other application (listener) did not give the socket to your application.
13	EACCES	Socket	Access denied. The client's ID is not in the OBEY file.
14	EFAULT	All	An incorrect storage address or length was specified.
15	ENOTBLK	All	A block device is required.
16	EBUSY	Givesocket	Listen has already been called for this socket.
17	EEXIST	All	The data set exists.
18	EXDEV	All	This is a cross-device link.
19	ENODEV	All	The specified device does not exist.
20	ENOTDIR	All	The specified directory is not a directory.
21	EISDIR	All	The specified directory is a directory.
22	EINVAL	All	An incorrect argument was specified.
22	EINVAL	Accept	Listen was not called for this socket.
22	EINVAL	Bind	The socket is already bound to an address.
22	EINVAL	Connect	The specified name length is incorrect.
22	EINVAL	Fcntl	Incorrect flags were specified.
22	EINVAL	Givesocket	An incorrect client ID was entered.
22	EINVAL	Ioctl	The request is incorrect or not supported.
22	EINVAL	Select	One of the fields in the timeout structure is incorrect.

Return	Socket Codes	Type	Description
22	EINVAL	Sendto	The target address length is incorrect for the specified address family.
22	EINVAL	Shutdown	The shutdown condition is not 0, 1, or 2.
22	EINVAL	Takesocket	The specified client ID is incorrect.
23	ENFILE	All	Data set table overflow occurred.
24	EMFILE	Takesocket	The socket descriptor table is full.
25	ENOTTY	All	An incorrect device call was specified.
26	ETXTBSY	All	A text data set is busy.
27	EFBIG	All	The specified data set is too large.
28	ENOSPC	All	There is no space left on the device.
29	ESPIPE	All	An incorrect seek was attempted.
30	EROFS	All	The data set system is Read only.
31	EMLINK	All	There are too many links.
32	EPIPE	All	The connection is broken.
33	EDOM	All	The specified argument is too large.
34	ERANGE	All	The result is too large.
35	EWOULDBLOCK	Accept	The socket is in non-blocking mode and connections are not queued. This is not an error condition.
35	EWOULDBLOCK	Read	The socket is in Recvfrom non-blocking mode and read data is not available. This is not an error condition.
35	EWOULDBLOCK	Send	The socket is in Sendto non-blocking mode and Write buffers are not available.
36	EINPROGRESS	Connect	The socket is marked non-blocking and the connection cannot be completed immediately. This is not an error condition.
37	EALREADY	Connect	The socket is marked non-blocking and the previous connection has not been completed.
38	ENOTSOCK	All	A socket operation was requested on a non-socket connection.
39	EDESTADDRREQ	All	A destination address is required.
40	EMSGSIZE	Sendto	The message is too long. The default is 8192 and the maximum is 32,767. The LARGEENVELOPEPOOLSIZE statement in PROFILE.TCPIP may restrict this value.
41	EPROTOTYPE	All	The specified protocol type is incorrect for this socket.
42	ENOPROTOOPT	Getsockopt	The socket option specified Setsockopt is incorrect or the level is not SOL_SOCKET.
43	EPROTONOSUPPORT	Socket	The specified protocol is not supported.
44	ESOCKTNOSUPPORT	All	The specified socket type is not supported.
45	EOPNOTSUPP	Accept	The selected socket is not Givesocket a stream socket.
45	EOPNOTSUPP	Listen	The socket does not support the Listen call.

Return	Socket Codes	Type	Description
45	EOPNOTSUPP	Getibmopt	The socket does not support Setibmopt this function call. This command is not supported for this function.
45	EOPNOTSUPP	Connect	Either a previous Connect failed or Connect was issued after Listen. This command is not supported for this function.
46	EPFNOSUPPORT	All	The specified protocol especially family is not supported or Getclientid the specified domain for Takesocket. The client identifier is Socket not AF_INET=2.
47	EAFNOSUPPORT	Bind	The specified address Connect family is not supported by Socket this protocol family.
48	EADDRINUSE	Bind	The address is in a timed wait because a LINGER delay from a previous close or another process is using the address.
49	EADDRNOTAVAIL	Bind	The specified address is incorrect for this host.
49	EADDRNOTAVAIL	Connect	The calling host cannot reach the specified destination.
50	ENETDOWN	All	The network is down.
51	ENETUNREACH	Connect	The network cannot be reached.
52	ENETRESET	All	The network dropped a connection on a reset.
53	ECONNABORTED	All	The software caused a connection abend.
54	ECONNRESET	All	The connection to the destination host is not available.
55	ENOBUFS	All	No buffer space is available.
55	ENOBUFS	Accept	Not enough buffer space is available to create the new socket.
55	ENOBUFS	Send	Not enough buffer space is Sendto available to send the new Write message.
55	ENOBUFS	Takesocket	There is a socket control block (SCB) or socket interface control block (SKCB) shortage in the TCPIP address space.
56	EISCONN	Connect	The socket is already connected.
57	ENOTCONN	All	The socket is not connected.
58	ESHUTDOWN	All	A Send cannot be processed after socket shutdown.
59	ETOOMANYREFS	All	There are too many references. A splice cannot be completed.
60	ETIMEDOUT	Connect	The connection timed out before it was completed.
61	ECONNREFUSED	Connect	The requested connection was refused.
62	ELOOP	All	There are too many symbolic loop levels.
63	ENAMETOOLONG	All	The file name is too long.
64	EHOSTDOWN	All	The host is down.
65	EHOSTUNREACH	All	There is no route to the host.
66	ENOTEMPTY	All	The directory is not empty.

Return	Socket Codes	Type	Description
67	EPROCLIM	All	There are too many processes in the system.
68	EUSERS	All	There are too many users on the system.
69	EDQUOT	All	The disk quota has been exceeded.
70	ESTALE	All	An old NFS(**) data set handle was found.
71	EREMOTE	All	There are too many levels of remote in the path.
72	ENOSTR	All	The device is not a stream device.
73	ETIME	All	The timer has expired.
74	ENOSR	All	There are no more stream resources.
75	ENOMSG	All	There is no message of the desired type.
76	EBADMSG	All	The system cannot read the message.
77	EIDRM	All	The identifier has been removed.
78	EDEADLK	All	A deadlock condition has occurred.

Glossary

APPC

Advanced Program-to-Program Communication (APPC) is a set of program-to-program communication services that support cooperative processing. In an APPC environment, programs running under MVS and Windows NT or aIX can communicate, transfer data back and forth, and activate processing tasks by programs within the same or different operating systems.

application

A computer program used for a particular kind of work, such as word processing or database management.

chain

The data portion of a VLAM Library member. A single Library member can have up to 32 chains. Each chain within the member is identified by a four character chain name.

Most any kind of data can be stored in a chain. Typical chain contents include print data streams, graphics, word processor source code, list of documents, a set of document variables and values, Windows Metafiles, and so on.

Many Docucorp products automatically extract and store multiple chain data as part of their processing.

check box

A small, square box that appears in a dialog box and that can be selected or cleared. When the check box is selected, an X appears in the box. A check box represents an option that you can turn on or off.

command button

In a Windows dialog box, a button that carries out an action. A command button often has a label that describes the action it carries out (for example, Cancel, Help, or Install). Choosing a command button that is followed by an ellipsis (for example, Setup...) causes another dialog box to appear.

dialog box

A window that appears temporarily to request information. Many dialog boxes have options you must choose before Windows can carry out a command.

document

One or more recorded or printed pages forming a logical whole.

Documerge

A high-volume electronic publishing software system developed by Docucorp. It combines fixed text stored in a VLAM Library with variable data retrieved and formatted by a custom designed Documerge component. Production volume and sophisticated capabilities that organize, sort, and format document Document Packages prior to actual printing are Documerge's strengths.

download

To load information from the host into memory or disk storage.

DTN

A Document Type Number (DTN) is an integer, 0 - 99999, assigned to each VLAM Library member. DTNs are also incorporated in the output formatting rules used by Documerge. As Documerge gathers information about the Library members used to build the output, it compares the members' DTNs with the DTNs in the formatting rules. The results of these comparisons gives Documerge much of the information it needs to properly combine and format the Library members used to build a printable output data stream.

EDL

The Electronic Document Library is equivalent to a VLAM Library. It is a computer file used by Docucorp products to store forms, images, and/or formatting rules. Libraries are supported on the mainframe (MVS and VSE environments) and on the PC (Windows NT or UNIX environments).

effective date

An effective date is a value assigned to a VLAM library member. The value is typically in a *mm/dd/yy* format. Documerge uses Effective Date fields as selection criteria when choosing among dated versions of a given member.

host

Any computer that is accessed by users and serves as a source of high-speed data processing for workstations with less computer power. Commonly referred to as mainframe, as opposed to workstation or service. See also*, mainframe*.

library

See EDL.

list box

Within a Windows application window or dialog box, a type of box that lists available choices. A list box might display a list of all files in a directory. If all the choices don't fit in the list box, the box displays a scroll bar.

LU 6.2

LU is the abbreviation for logical unit. An LU is the component of IBM's Systems Network Architecture (SNA) responsible for providing each end user access to the SNA network as well as being responsible for handling communication between end users.

SNA defines different types of logical units to meet the needs of specific end users, whether the end user is an application program, a stand-alone terminal, or a terminal and an operator. **LU 6.2** is a type of logical unit that is specifically designed to handle communications between application programs.

mainframe

A high-level computer designed for intensive computational tasks. Mainframe computers are often shared by multiple users connected to the computer via terminals. They are typically capable of performing applications that access and process large amounts of data, e.g. a large scale payroll system. See also, *host*.

menu

Within the Windows environment, a menu is a list of available commands displayed near the top of an application window.

MS-DOS

Microsoft-Disk Operating System. An operating system designed for use on IBM, and IBM-compatible PCs. It's a single-tasking, single-user operating system with a command line interface.

MVS

Multiple Virtual Storage. An IBM operating system designed for use on System/370 processors. There is an MVS version of the host portions of PrintCommander.

operating system

Software that controls the execution of programs. An operating system may provide services such as resource allocation, scheduling, input/output control, and data management.

path

Within the MS-DOS operating system, the path refers to a listing of the directories that lead from the current directory to a file.

revision level

An integer that indicates the version level of a VLAM Library member. The lowest revision level is one. Higher revision levels indicate newer versions of a given member.

server

A shared computer that provides disk space, printers, or other services to computers over a network.

text box

In a dialog box, a box in which you type information needed to carry out a command. The text box may be blank or may contain text when the dialog box opens.

transaction program

A program used for cooperative transaction processing within an SNA network. For APPC/MVS, any program on MVS that issues APPC/MVS or CPI Communication calls, or is scheduled by the APPC/MVS transaction scheduler.

VTAM

Virtual Telecommunications Access Method (VTAM) is a feature of IBM's SNA architecture that directs data between programs and devices. VTAM enables communication between applications running on different processors, such as an application on an AIX platform and an application on a mainframe running MVS.

INDEX