AS/400e

Communications Management

Version 4

SC41-5406-02

AS/400e

Communications Management

Version 4

SC41-5406-02

Note

Before using this information and the product it supports, be sure to read the information in “Notices” on page 249.

Third Edition (May 1999)

This edition applies to version 4, release 4, modification 0 of Operating System/400 licensed program (product number 5769-SS1) and to all subsequent releases and modifications until otherwise indicated in new editions. This edition applies only to reduced instruction set computer (RISC) systems.

| This edition replaces SC41-5406-01.

Contents

Figures

Tables

About Communications Management (SC41-5406)

In this book, the term ‘user’ refers to the application user. The term ‘operator’ refers to the system operator.

This book contains information that is needed for managing communications on the AS/400 system. The *Communications Configuration* book describes the objects, commands, and parameters that are used to configure OS/400 communications. Once that task is complete, you may require additional information on how to manage the work your AS/400 system performs. This book provides information on how to do the following:

- Determine the status of your communications objects
- Trace and diagnose communications problems
- Handle and recover from communications errors
- Improve performance
- Determine the aggregate line speed and subsystem storage needs of your AS/400 system

This book is divided into separate chapters to address the following topics:

- Work management in a communications environment

  This topic discusses the use of subsystems to control your communications jobs. Also included are discussions of communications entries, routing information, and how to handle program start request failures.
- Communications status and configurations

  This topic discusses how to do the following:
  - Determine the status of communications sessions and conversations
  - Work with communications configurations
  - Vary on or off a communications object
  - Retrieve configuration status
  - Display connection status
  - Display inbound routing information
  - Display mode status
  - Change the maximum number of sessions
- Tracing and diagnosing communications problems

  This topic discusses:
  - How to trace communications lines
  - Common Programming Interface Communications
  - Intersystem communications function (ICF) operations and functions used by a user program

  It also provides information on how to isolate problems in an Advanced Peer-to-Peer Networking (APPN) environment.
- Handling communications errors

  This topic provides error recovery information for the following:
  - Communications
  - Application programs
  - Operating system
  - Remote work station loss of power
  - Subsystems

Communications configuration flow charts for error recovery procedures are provided.

Also included in this chapter is a discussion of communications problem analysis covering Link Problem Determination Aid (LPDA) tests, system service tools, and automatic communications recovery.

- Communications threshold process

  This topic discusses the threshold process and error checks that are provided by the system.
- Performance

  This topic includes information on line speed, X.21 short-hold mode port sharing, line disconnection, modems, data link protocols, printers, and pacing.
- Aggregate line speed and subsystem storage

  This topic discusses maximum aggregate line speed on the various AS/400 models in addition to how to calculate subsystem storage for each of the AS/400 models.
- Appendixes

  The appendixes cover virtual telecommunications access method (VTAM) node support, data and security coexistence, and an overview of communications functions and data link protocol considerations.

You may need to refer to other IBM books for more specific information about a particular topic. The *Publications Reference* provides information on all the books in the AS/400 library.

Who Should Read This Book

This book supplies programmers and system administrators with the information that is needed to develop, maintain, and support communications on the AS/400 system. It provides an overview of communications management and how the system works.

You should be familiar with the general communications concepts and communications configuration on the AS/400 system. For more information on general communications concepts, refer to Discover/Education Introduction to Data Communications course, which you may order separately.

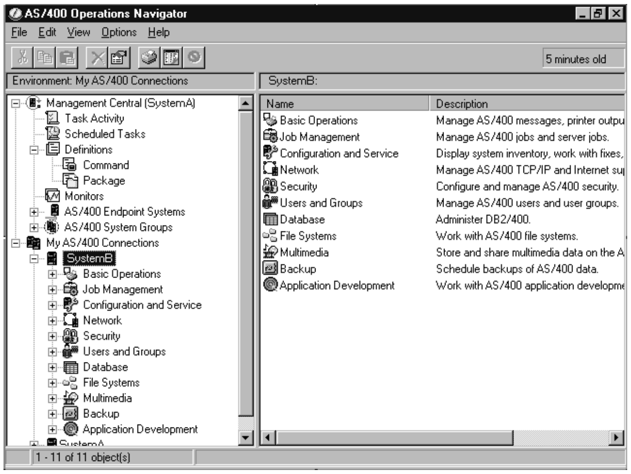
You should have read the *System Operation* book and the *Basic System Operation, Administration, and Problem Handling* book or have the equivalent knowledge.

AS/400 Operations Navigator

AS/400 Operations Navigator is a powerful graphical interface for Windows clients. With AS/400 Operations Navigator, you can manage and administer your AS/400 systems from your Windows desktop.

You can use Operations Navigator to manage communications, printing, database, security, and other system operations. Operations Navigator includes Management Central for managing multiple AS/400 systems centrally.

Figure 1 on page xiii shows an example of the Operations Navigator display:



SystemB:

Name	Description
Basic Operations	Manage AS/400 messages, printer outpu
Job Management	Manage AS/400 jobs and server jobs.
Configuration and Service	Display system inventory, work with fixes,
Network	Manage AS/400 TCP/IP and Internet su
Security	Configure and manage AS/400 security.
Users and Groups	Manage AS/400 users and user groups.
Database	Administer DB2/400.
File Systems	Work with AS/400 file systems.
Multimedia	Store and share multimedia data on the A
Backup	Schedule backups of AS/400 data.
Application Development	Work with AS/400 application developme

Figure 1. AS/400 Operations Navigator Display

This new interface has been designed to make you more productive and is the only user interface to new, advanced features of OS/400. Therefore, IBM recommends that you use AS/400 Operations Navigator, which has online help to guide you. While this interface is being developed, you may still need to use a traditional emulator such as PC5250 to do some of your tasks.

Installing Operations Navigator

To use AS/400 Operations Navigator, you must have Client Access installed on your Windows PC. For help in connecting your Windows PC to your AS/400 system, consult *Client Access Express for Windows - Setup*, SC41-5507-00.

AS/400 Operations Navigator is a separately installable component of Client Access that contains many subcomponents. If you are installing for the first time and you use the **Typical** installation option, the following options are installed by default:

- Operations Navigator base support
- Basic operations (messages, printer output, and printers)

To select the subcomponents that you want to install, select the **Custom** installation option. (After Operations Navigator has been installed, you can add subcomponents by using Client Access Selective Setup.)

1. Display the list of currently installed subcomponents in the **Component Selection** window of **Custom** installation or Selective Setup.
2. Select AS/400 Operations Navigator.
3. Select any additional subcomponents that you want to install and continue with **Custom** installation or Selective Setup.

After you install Client Access, double-click the **AS400 Operations Navigator** icon on your desktop to access Operations Navigator and create an AS/400 connection.

Prerequisite and related information

Use the AS/400 Information Center as your starting point for looking up AS/400 technical information. You can access the Information Center from the AS/400e Information Center CD-ROM (English version: *SK3T-2027*) or from one of these Web sites:

```
http://www.as400.ibm.com/infocenter
http://publib.boulder.ibm.com/pubs/html/as400/infocenter.htm
```

The AS/400 Information Center contains important topics such as logical partitioning, clustering, Java, TCP/IP, Web serving, and secured networks. It also contains Internet links to Web sites such as the AS/400 Online Library and the AS/400 Technical Studio. Included in the Information Center is a link that describes at a high level the differences in information between the Information Center and the Online Library.

For a list of related publications, see the “AS/400 Books” on page 253.

How to send your comments

Your feedback is important in helping to provide the most accurate and high-quality information. If you have any comments about this book or any other AS/400 documentation, fill out the readers' comment form at the back of this book.

- If you prefer to send comments by mail, use the readers' comment form with the address that is printed on the back. If you are mailing a readers' comment form from a country other than the United States, you can give the form to the local IBM branch office or IBM representative for postage-paid mailing.
- If you prefer to send comments by FAX, use either of the following numbers:
  - United States and Canada: 1-800-937-3430
  - Other countries: 1-507-253-5192
- If you prefer to send comments electronically, use one of these e-mail addresses:
  - Comments on books:

    RCHCLERK@us.ibm.com

    IBMMAIL, to IBMMAIL(USIB56RZ)
  - Comments on the AS/400 Information Center:

    RCHINFOC@us.ibm.com

Be sure to include the following:

- The name of the book.
- The publication number of the book.
- The page number or topic to which your comment applies.

Summary of Changes for Communications Management

Subsystem Configuration Considerations
An expanded discussion on subsystem configuration considerations. See "Subsystem Configuration Considerations" on page 3.

Varying a Configuration On or Off
Additional information on varying a configuration on and off, including some examples. See "Varying a Configuration On or Off" on page 24.

Managing Communication Messages
Additional information about the primary places where communications messages are logged, including a new message queue, QCFGMSGQ. See "Managing Communication Messages" on page 76.

System Value Considerations
New information about QCMNARB system values. See "System Value Considerations" on page 109.

Handling Communications Errors
Expanded chapter on handling communications errors , with updated information on configuration considerations, job considerations, and communications error recovery problem analysis. See "Chapter 4. Handling Communications Errors" on page 107.

Miscellaneous Changes:
Updated AS/400 displays, systems, and network device names.

A vertical line (|) to the left of the text indicates a change or addition.

Chapter 1. Work Management

All of the work that is done is submitted through the work management function. You can design specialized operating environments to handle different types of work to satisfy the requirements of your system. However, when the operating system is installed, it includes a work management environment that supports interactive and batch processing, communications, and spool processing. This chapter discusses the communications aspects of work management.

The operating system allows you to tailor this support or to create your own work management environment. To do this, you need an understanding of the work management concepts. Following is an overview of the work management concepts and the objects that are supplied by IBM. See the *Work Management* book for more information about work management.

Concepts

On the AS/400 system, all user jobs operate in an environment that is called a **subsystem.** This is where the system coordinates processing and resources. A subsystem description defines a subsystem. Placing a group of jobs with common characteristics in a subsystem enables users to control the jobs independently of others. Users can easily start and end subsystems as needed to support the work being done, and to maintain desired performance characteristics. One such subsystem, called a **controlling subsystem**, automatically starts when the system loads. (For information on how to load the system, see the *Basic System Operation, Administration, and Problem Handling* book.)

IBM supplies two subsystem configurations, and may be used without charge. The first configuration includes the following subsystems:

- QBASE, the controlling subsystem, supports interactive, batch, and communications jobs.
- QSPL supports processing of spooling readers and writers.
- QSYSWRK supports system functions.
- QSERVER supports file serving.
- QUSRSYS supports server jobs that performs user functions.

QBASE and QSYSWRK are automatically started when the system starts. An automatically started job in QBASE starts QSPL, QSERVER, and QUSRSYS.

The second subsystem configuration supplied by IBM is more complex. This configuration consists of the following subsystems:

- QCTL, the controlling subsystem, supports interactive jobs that are started at the console.
- QINTER supports interactive jobs that are started at other work stations.
- QCMN supports communications jobs.
- QBATCH supports batch jobs.
- QSPL supports processing of spooling readers and writers.
- QSYSWRK supports system functions.
- QSERVER supports file serving.
- QUSRSYS supports server jobs that performs user functions.

Note: Job queues with names identical to their corresponding IBM subsystems are also supplied. Jobs can start from these queues after their corresponding subsystems have started.

| The QBASE configuration provides the capability to run all the same functions
| possible with the QCTL configuration. Because QBASE consists of fewer
| subsystems, it is easier to manage.

If you change your configuration to use the QCTL controlling subsystem, it starts automatically when the system starts. An automatically started job in QCTL starts the other subsystems.

You can change your subsystem configuration from QBASE to QCTL by changing the system value QCTLSBSD (controlling subsystem) to 'QCTL QSYS' on the Change System Value (CHGSYSVAL) command. Then, perform an initial program load (IPL) of the system.

For more information about the controlling subsystems that are shipped by IBM, and for additional subsystem descriptions, see the *Work Management* book.

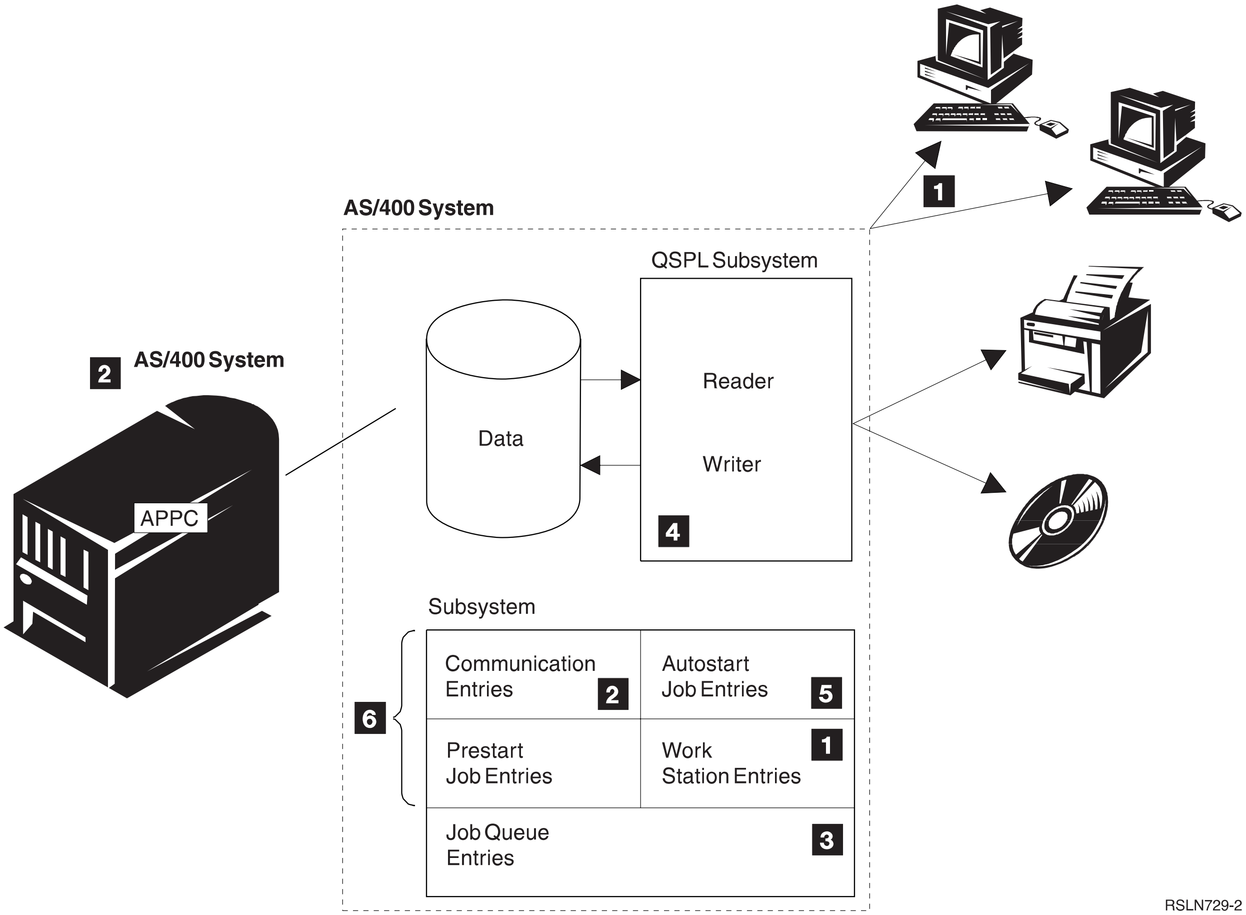


Figure 2. Work Management Environment

Five basic types of jobs run on the system:

- Interactive jobs
- Batch jobs

- Spooling jobs
- Autostart jobs
- Prestart jobs

Figure 2 on page 2 shows these jobs and their relationship to each other.

1 An interactive job starts when you sign on a work station and ends when you sign off.

2 A communications batch job is a job that is started from a program start request from another system.

3 A non-communications batch job is started from a job queue. Job queues are not used when starting a communications batch job.

4 Spooling functions are available for both input and output. For input spooling, a system program, called a reader, transfers job instructions and data from an input device (diskette or database file) to a job queue. For output spooling, the system places output records that are produced by a program in an output queue.

5 Autostart jobs perform repetitive work or one-time initialization work. Autostart jobs are associated with a particular subsystem. Each time the subsystem is started, the associated autostart jobs are started.

6 Prestart jobs are jobs that begin running before the remote program sends a program start request. To run a prestart job, you first must define both communications and prestart job entries in the same subsystem description. Then, make specific programming changes to the prestart job target program with which your program communicates. For information about defining communications and prestart job entries and managing prestart jobs, see the *Work Management* book. For information on designing prestart jobs to use the intersystem communications function (ICF), see the *ICF Programming* book. For information about designing prestart jobs to use CPI Communications, see the *APPC Programming* book.

Subsystem Configuration Considerations

If you have any or all of the following, you need to consider how you configure your subsystems for your users:

- Local work stations
- Remote work stations
- Advanced program-to-program communications (APPC)-based 5250 display sessions
- Telnet sessions
- General APPC communications activity
- Any other interactive job

Placing too many users (or too much work load) on a subsystem can adversely affect the users on that subsystem. Subsystem monitor jobs provide function for initiating and ending jobs. In addition, subsystems perform device recovery when a job is ended for some reason. There are various reasons why device recovery may be needed; some examples include powering off a display, ending an emulator connection from the client, or a network failure. This device recovery is done on one device at a time, and is a synchronous operation.

When a subsystem has many jobs that start or end at one point in time, or many devices to recover at one time, it can become very busy, and negatively impact the work of others also running in that subsystem. For example, users may not be able to sign on to the system if the interactive subsystem in which they run is busy.

To maintain good system performance, you should limit the number of devices allocated to an interactive or communications subsystem to between 200 and 300. Spreading the work across multiple subsystems provides multiple processes to handle the work. This provide better error isolation. In addition, multiple processes can lead to greater parallelism on multi-processor systems.

To minimize this effect, separate your system's work load into multiple subsystems.

In summary, consider these four points when configuring your subsystems:

1. **The number of users and jobs**

   Limit the number of users and jobs that are serviced by any single subsystem to 300.
2. **The connectivity used to access the system**

   Subsystems perform best when remote users and jobs are isolated from local users and jobs. Correspondingly, you should isolate local area network (LAN) users and jobs.

   A good way to perform the subsystem configuration is to define one QINTER/QCMN subsystem pair for any communications line that can support many users. For example, place all users on a token-ring line into their own QINTER/QCMN subsystem pair, and place users connected by a 5494 remote controller in another subsystem pair.
3. **The type of work the subsystem has to perform**

   In this case, consider the following:
   - Are there many jobs starting and ending throughout the day?
   - Does the subsystem handle many communications evokes?
   - Does the subsystem have frequent device allocation and deallocation?
   - Do the devices in the subsystem frequently go into error recovery?

     Multiple subsystems provide multiple processes to perform cleanup and recovery when an error condition occurs. This may result in improved performance.
   - Is one set of users more time-sensitive than another?
   - Is the subsystem affected by vary-on and vary-off processing?
4. **The geographic location of the users**

Use the above considerations to determine the best configuration for your interactive subsystems, as well as your communications subsystems.

Interactive Subsystem Configuration Considerations

To configure interactive subsystems, do the following:

1. Identify how you want the interactive users separated, and create the appropriate subsystem descriptions.
2. For each subsystem you define, use the Add Work Station Entry (ADDWSE) command to identify which users will run in which subsystem.

   Each subsystem requires an appropriate work station entry for the users that will run in it. You can set up work station entries that identify the devices a subsystem allocates, and the devices a subsystem does not allocate.

You can use generic names for work station name entries on the work station entry commands. These commands include, Add Work Station Entry (ADDWSE), Change Work Station Entry (CHGWSE), and Remove Work Station Entry (RMVWSE). These generic names for work station entries provide easier addition of work stations to an already active subsystem.

To specify which devices a particular subsystem should allocate, use the following command:

- **ADDWSE SBSD(libname/sbsname) WRKSTN(devname*) AT(*SIGNON)**

To specify which devices a particular subsystem should not allocate, use the following command. This workstation entry will allow a job at that device to transfer into that subsystem.

- **ADDWSE SBSD(libname/sbsname) WRKSTN(devname*) AT(*ENTER)**

Note: If the subsystem description does not have a workstation entry for a particular device, the subsystem will not allocate it. However, you cannot transfer an interactive job into that subsystem with that unallocate device.

See *Work Management* for more information on configuring interactive subsystems.

The QDEVRCYACN system value specifies the device recovery action than is taken when an I/O error occurs. The device recovery action can make a significant difference in the performance of your subsystem, and the system as a whole, when device I/O errors occur. This especially applies when many users experience the same error simultaneously (for example, as a result of a network failure). See “Device Recovery Action (DEVRCYACN)” on page 117, and the *Work Management* book for more information about QDEVRCYACN.

Communications Subsystem Configuration Considerations

You may benefit by configuring multiple communications subsystems. However, the work that is done in a communications subsystem generally is less than the work that is performed in a corresponding interactive subsystem. Consider configuring fewer communication subsystems than interactive subsystems.

Some of the work performed in the communications subsystem is done only when users connect and disconnect, and for error recovery. This consideration is important in the configuration of the communications subsystem.

To configure multiple communications subsystems, use the Add Communications Entry (ADDCMNE) and Remove Communications Entry (RMVCMNE) commands to set up the communications entries.

To specify which devices a communications subsystem should allocate, use the following command:

ADDCMNE SBSD(libname/sbsname) DEV(devname*)

To specify which devices a communications subsystem should not allocate, use the following command:

ADDCMNE SBSD(libname/sbsname) DEV(devname*) MAXACT(0)

See “Communications Device Allocation” on page 6 and the *Work Management* book for additional information.

Switched Line Considerations

When a call is successful, the remote system may begin a session with a correctly configured subsystem monitor. Before program start requests are accepted by an AS/400 system, a subsystem that supports communications must start.

Subsystem monitors support the answer function from remote systems. If the call function is desired, a user program must make the call manually. Or, you can cause the connection to be established by opening a file or acquiring a device on an associated controller that specifies INLCNN(*DIAL).

IBM-supplied QBASE and QCMN Subsystem Descriptions

Before program start requests are accepted by an AS/400 system, a subsystem that supports communications must be started. Two IBM-supplied subsystems, QBASE and QCMN, accept program start requests for all communications types. QBASE is the default controlling subsystem. QCMN is the communications subsystem that is used when QCTL is the controlling subsystem. Having the QBASE or QCMN subsystem active allows program start requests to be accepted for all communications types.

The QCMN and QBASE subsystems have device-type entries of *ALL and mode(*ANY). Both subsystems have the appropriate routing entries, using CMPVAL(PGMEVOKE 29), so all program start requests received by the AS/400 system are accepted. If you have either of these subsystems and then start your own communications subsystem or other subsystems, such as DSNX(QDSNX) or SNADS(QSNADS), read “Communications Device Allocation”. Both subsystems attempt to allocate the same communications devices.

Communications Device Allocation

When subsystems start, they request allocation of all communications devices for communications entries in the subsystem description. The requests are sent to the QLUS (LU services) system job that handles device allocation for all communications devices.

QLUS is notified when a communications device is available for program start request processing. This notification occurs when the connection between the local and remote system is established for that device. When QLUS receives this notification, it attempts to allocate the communications device to a subsystem based on communication entry definitions. If there is no subsystem active that wants to use the device, QLUS maintains allocation of the device until the device is varied off, or until a subsystem starts that wants to use the device.

Rules for Device Allocation

When more than one subsystem contains communications entries for a communications device, QLUS uses the following rules to determine which subsystem uses the device when the device is available:

- Communications entries with the highest level of detail for the device are processed first. The order (from highest to lowest) of detail is: Device name entry, remote location name entry, and device-type entry.

- Mode names are only used for APPC devices. Each mode on each device is allocated to a subsystem. A specific mode name takes priority over the generic *ANY mode name.
- The time that the subsystem requested the device (when the subsystem is started) is used to break ties when two or more subsystems have the same level of detail for the device and mode.

When a communications device is allocated to a subsystem, it remains in that subsystem until the subsystem deallocates the device. When a subsystem deallocates a device (and deallocation is not due to a device error or varying off the device), QLUS attempts to allocate the device to another subsystem.

If a subsystem has a communications device allocated, and you start a second subsystem that should use the same communications device allocated to it (based on the device allocation rules), you can force the original subsystem to deallocate the device. Here are some ways to cause a subsystem to deallocate a communications device:

- Varying the device off, and then on again causes QLUS to attempt to allocate the device to a subsystem.
- Issuing the Allocate Object (ALCOBJ) command against the device works for BSCEL, SNUF, retail, finance, and asynchronous communications types (request an *EXCLRD lock). Issuing the Deallocate Object (DLCOBJ) command causes QLUS to attempt to allocate the device to a subsystem.
- Issuing the End Subsystem (ENDSBS) command to end the first subsystem causes QLUS to automatically attempt to allocate the device to another subsystem.

This deallocation causes QLUS to go through the device allocation algorithm again, which eventually causes the device to be allocated to the second subsystem.

Describing a Subsystem

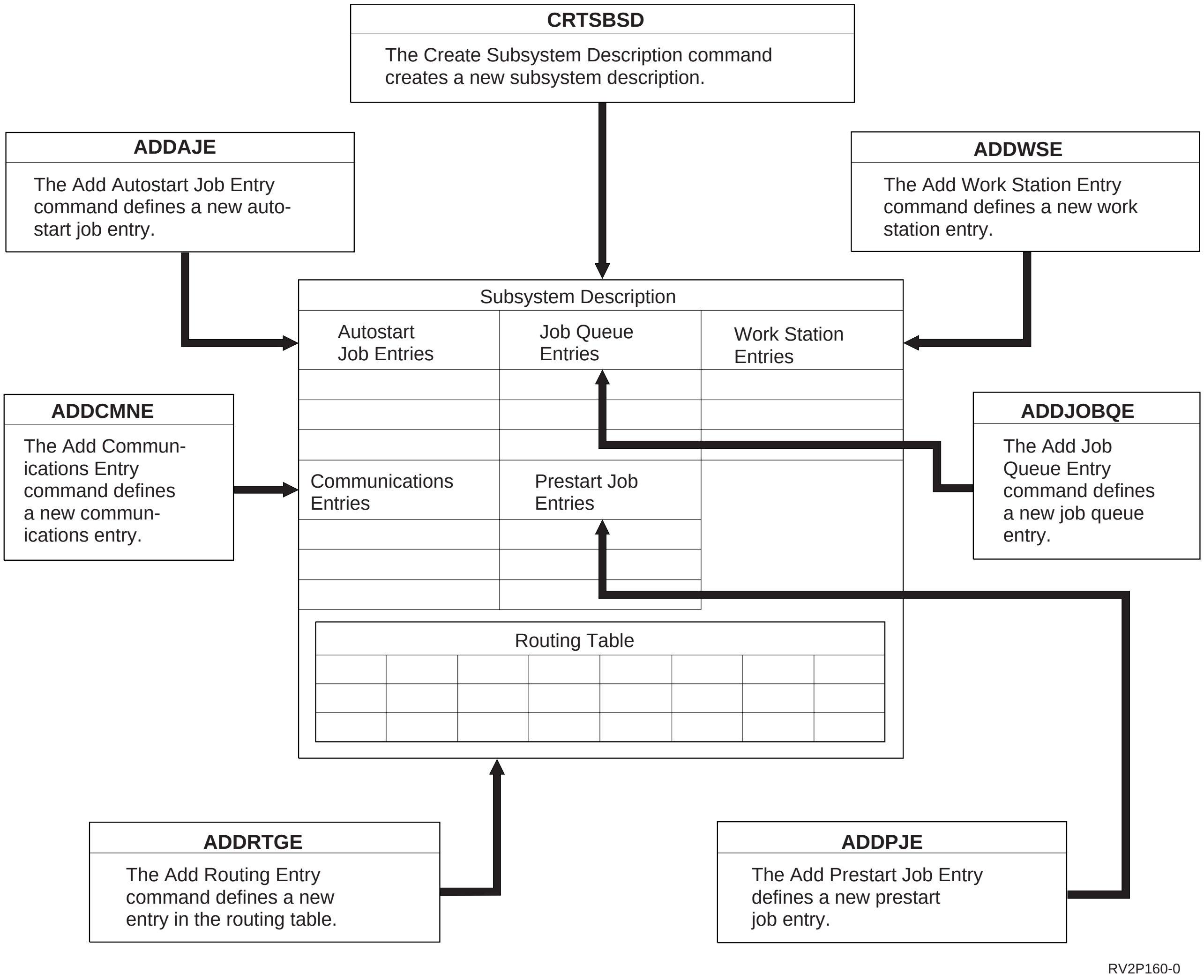
A subsystem description consists of these parts:

- Subsystem attributes (overall subsystem characteristics).
- Work entries (sources of work). Only the communications entry is discussed in this book. The communications job is processed when the subsystem receives a program start request from a remote system. See the topic “Adding a Communications Entry” on page 9.
- Routing entries (define how a job’s routing step is started). See the topic “Adding Routing Information” on page 11.

One subsystem attribute is storage pool definitions. Storage pools are logical allocations of main storage. The same storage pool can be shared by multiple subsystems. You can display and change your storage pool definitions with the Work with Shared Pools (WRKSHRPOOL) command. The WRKSHRPOOL command allows you to work with shared pools only. A subsystem might also have private pools, which need the Change Subsystem Description (CHGSBSD) command for changes. If a subsystem is active, the Work with System Status (WRKSYSSTS) command provides an interface to change both shared and private pools.

You can change the IBM-supplied subsystem descriptions or any user-created subsystem descriptions by using the Change Subsystem Description (CHGSBSD) command. You can use this command to change the storage pool size, storage pool activity level, and the maximum number of jobs for the subsystem description of an active subsystem.

Figure 3 shows the relationship of the communications entries and routing entries with the subsystem description.



CRTSBSD
The Create Subsystem Description command creates a new subsystem description.

ADDAJE
The Add Autostart Job Entry command defines a new autostart job entry.

ADDWSE
The Add Work Station Entry command defines a new work station entry.

ADDCMNE
The Add Communications Entry command defines a new communications entry.

ADDJOBQE
The Add Job Queue Entry command defines a new job queue entry.

ADDRTGE
The Add Routing Entry command defines a new entry in the routing table.

ADDPJE
The Add Prestart Job Entry defines a new prestart job entry.

Figure 3. Subsystem Descriptions and Work within the System

Communications Entries in Subsystem Descriptions

The AS/400 system considers communications devices to be another source of work for a subsystem. Therefore, a communications entry must be defined within the subsystem description to identify the devices from which work (program start requests) can be received by the subsystem. Subsystem descriptions are created using the Create Subsystem Description (CRTSBSD) command. Communications entries are added to a subsystem description using the Add Communications Entry (ADDCMNE) command.

Adding a Communications Entry

You can use the Add Communications Entry (ADDCMNE) command to add a communications entry to an existing subsystem description.

Each communications entry describes one or more devices or remote locations that are controlled by the subsystem. The devices identified in the communications entries are allocated by the subsystem for receiving program start requests to start communications batch jobs.

Table 1 shows the parameters for the ADDCMNE command.

Table 1. ADDCMNE Parameters

Parameter Name	Keyword	Description	Valid Values
Subsystem description	SBSD	Name and library of the subsystem description.	*LIBL, *CURLIB, or a user-specified library name; if no library is named, *LIBL is assumed
Device[1]	DEV	Name of the device description or the type of device being used.	*device-description-name*, *ALL, *APPC, *ASYNC, *BSCEL, *FINANCE, *INTRA, *RETAIL, or *SNUF
Remote location[1]	RMTLOCNAME	Name of the remote location.	*remote-location-name*.
Job description	JOBD	Name and library of the job description. If the job description does not exist when the communications entry is added, a library qualifier must be specified.	*USRPRF, *SBSD, or *job-description-name*. Possible library values are *LIBL, *CURLIB, or a user-specified library name
Default user profile[2]	DFTUSR	Default user ID (user profile).	*user-profile-name*, *NONE, or *SYS
Mode[3]	MODE	If the communications type being used supports modes, this is the name used by both ends of the data link to refer to this group of sessions.	*ANY, BLANK, or *mode-name*
Maximum active jobs	MAXACT	Maximum number of jobs (program start requests) that can be active at the same time.	*NOMAX or *maximum-active-jobs*

Notes:

[1] You must specify either the *Device* prompt or the *Remote location* prompt, but not both.

[2] The names QSECOFR, QSPL, QDOC, QDBSHR, QRJE, and QSYS are not valid entries for this parameter.

[3] Specific mode name entries are processed before *ANY entries are processed.

The following command adds a communications entry for the remote location CHICAGO to the subsystem description SBS1, which resides in the ALIB library. The DFTUSR parameter defaults to *NONE, meaning that no jobs may enter the system through this entry unless valid security information is supplied on the program start request.

```
ADDCMNE   SBSD(ALIB/SBS1)  RMTLOCNAME(CHICAGO)
```

Changing a Communications Entry

You can use the Change Communications Entry (CHGCMNE) command to change the attributes of a communications entry in an existing subsystem description. The parameters used on this command are the same as the parameters on the Add Communications Entry command. For a description of these parameters, refer to “Adding a Communications Entry” on page 9. To determine the communications entry to change, use the SBSD, DEV, RMTLOCNAME, and MODE parameters.

The following command changes the communications entry (in the subsystem description QGPL/BAKER) for the remote location CHICAGO.

```
CHGCMNE   SBSD(QGPL/BAKER)  RMTLOCNAME(CHICAGO)
MAXACT(*NOMAX)
```

The maximum activity level is changed to *NOMAX. This means that the communications entry puts no restrictions on the number of program start requests that may be active at the same time. However, the MAXJOBS value in the subsystem description BAKER limits the total number of jobs that can be active in the subsystem. This limit includes those created by a program start request. Also, a user can specify a limit on the number of active jobs that can be routed through any particular routing entry (MAXACT). The limit specified in the routing entry may control the number of jobs using a particular pool or the number of calls of a particular program. In all cases, none of these limits can be exceeded as a result of processing a program start request.

Removing a Communications Entry

You can use the Remove Communications Entry (RMVCMNE) command to remove a communications entry from an existing subsystem description. The SBSD, DEV, RMTLOCNAME, and MODE parameters are used to determine which communications entry to remove. If a communications entry is to be removed from an active subsystem, jobs that used this entry cannot be active. Refer to “Adding a Communications Entry” on page 9 for a description of the parameters.

Note: MODE(*ANY) only removes an entry of *ANY, not specific mode entries.

The following command removes the communications entry for the remote location CHICAGO from the subsystem description SBS1 in library LIB2.

```
RMVCMNE   SBSD(LIB2/SBS1)  RMTLOCNAME(CHICAGO)
```

Routing Information for Communications Entries

When the system processes a program start request, it creates a fixed-length data string that is used to route data. The data used to build the routing data string comes from the program start request. Program start requests insert the character string PGMEVOKE (Program Evoke) into position 29 of the routing data string. This ensures a match to a routing entry with a compare value of PGMEVOKE in position 29. When you use the Add Routing Entry command to define a routing entry for a subsystem description, you specify a value for the Program to Call (PGM) parameter. That value specifies the name of the program that is to be run for the routing entry. If you specify *RTGDTA (Routing Data) for this parameter, the program name that is specified in the program start request is used.

Note: The PGMEVOKE value must be typed in uppercase.

The routing data created by an incoming program start request contains the character string PGMEVOKE beginning in position 29. This character string may be used to route program start requests differently than interactive or batch jobs.

Note: To add, change, or remove routing entries, you must have object operational and object management authorities for the subsystem description.

The following shows the format of the system-generated routing data for a program start request:

```
Mode Name   Device Name  User ID Name
|       |   |         |  |          |
1       8   9        18  19        28


PGMEVOKE  Program Name  Library Name
|      |  |          |  |          |
29    36  37        46  47        56
```

Adding Routing Information

The Add Routing Entry (ADDRTGE) command adds a routing entry to the specified subsystem description. Each routing entry specifies the parameters used to start a routing step for a job. For example, the routing entry specifies the name of the program to run when the routing data that matches the compare value in this routing entry is received.

Table 2 shows the parameters for the ADDRTGE command.

Table 2. ADDRTGE Parameters

Parameter Name	Keyword	Description	Valid Values
Subsystem description	SBSD	Name and library of the subsystem description.	Subsystem description name, *LIBL, *CURLIB, or a user-specified library name; if no library is named, *LIBL is assumed.
Sequence number	SEQNBR	Sequence number of the routing entry.	1 through 9999
Compare value	CMPVAL	A value that is compared with the routing data to determine whether this is the routing entry used for starting a routing step. If the routing data matches the routing entry compare value, that routing entry is used. Optionally, a starting position in the routing data character string can be specified for the comparison.	*ANY, *compare-value:*, 1, or *starting-position*
Program	PGM	Name and library of the program called as the first program run in the routing step.	*RTGDTA or *program-name* Possible library values are: *LIBL, *CURLIB, or a user-specified library name.

Table 2. ADDRTGE Parameters (continued)

Parameter Name	Keyword	Description	Valid Values
Class	CLS	Name and library of the class used for the routing steps.	*SBSD or *class-name* Possible library values are: *LIBL, *CURLIB, or a user-specified library name.
Maximum active routing steps	MAXACT	Maximum number of jobs that can be active at the same time.	*NOMAX or *maximum-active-jobs*
Storage pool ID	POOLID	Pool identifier of the storage pool in which the program runs.	1 or *pool-identifier*

The following example shows how the sequence number and the compare value work together. When you add a routing entry to a subsystem description, you assign a sequence number to the entry. This sequence number tells the subsystem the order in which routing entries are searched for a routing data match. For example, you have a subsystem description that contains the following five routing entries:

Sequence Number	Compare Value
10	'ABC'
20	'AB'
30	'A'
40	'E'
50	'D'

The routing entries are searched in sequence number order. If the routing data is 'A', the search ends with routing entry 30. If the routing data is 'AB', the search ends with routing entry 20. If the routing data is 'ABC', the search ends with routing entry 10. Because routing data can be longer than the compare value of the routing entry, the comparison, which is done in left-to-right order, stops when it reaches the end of the compare value. Therefore, if the routing data is 'ABCD', the search ends with routing entry 10.

If routing entry 20 is changed to have a compare value of 'ABCD', the results of the routing search are different. In this case, the routing entries are:

Sequence Number	Compare Value
10	'ABC'
20	'ABCD'
30	'A'
40	'E'
50	'D'

If the routing data is 'A', the search ends with routing entry 30. If the routing data is 'AB', the search ends with routing entry 30. If the routing data is 'ABC', the search ends with routing entry 10. If the routing data is 'ABCD', the search ends with routing entry 10.

In this case, it is no longer possible to match routing entry 20. This is because any routing data that matches the compare value for routing entry 20 matches the routing entry 10 first. When a routing entry is changed or added to a subsystem description with a compare value that causes this situation, the system sends a diagnostic message identifying the situation.

When you add routing entries to a subsystem description, you should order them so that the entries likely to be compared oftenest are first. This reduces the search time.

You can specify a comparison value of *ANY on the highest numbered routing entry. *ANY means that a match is forced regardless of the routing data. Only one routing entry can contain the comparison value of *ANY, and it must be the last (highest sequence number) entry in the subsystem description.

The program named in the routing entry is given control when the routing step for the job is started. Parameters to control the running of the routing step for the job are taken from the class specified in the routing entry.

This following example adds routing entry 500 to the routing portion of the subsystem description CMNSBS. To use this routing entry, the routing data must contain the characters PGMEVOKE starting in position 29. This is always true for jobs that are being started as a result of a program start request being received. Any number of jobs can be active through this routing entry at any one time. The program that was sent on the program start request runs in the job because PGM(*RTGDTA) was specified.

```
ADDRTGE   SBSD(CMNSBS)  SEQNBR(500)
          CMPVAL(PGMEVOKE 29)   PGM(*RTGDTA)
          CLS(QGPL/QBATCH)  POOLID(2)
```

Note: The PGMEVOKE value must be typed in uppercase.

If a library was sent on the program start request, the subsystem searches that library for the program. Otherwise, the subsystem searches the subsystem's library list for the program. The job runs in storage pool 2 using class QBATCH in library QGPL.

The following command adds routing entry 210 to the routing portion of the subsystem description CMNSBS. To use this routing entry, the routing data must contain the characters PROGRAMA that starts in position 37. For jobs being started as a result of a program start request being received, position 37 always contains the name of the program sent on the program start request. This allows special handling for all jobs that run PROGRAMA. For example, the job can run with a different class, or it can run in a different pool.

```
ADDRTGE   SBSD(CMNSBS)  SEQNBR(210)
          CMPVAL(PROGRAMA 37)  PGM(*RTGDTA)
          CLS(QGPL/CMNCLASS)  POOLID(3)
```

Note: The compare value (CMPVAL) is case sensitive. If a program start request was received for program `PROGRAMA`, for example, this routing entry would not be selected.

When using the program name as a compare value (CMPVAL), that routing entry should have a lower sequence number than the routing entry with PGMEVOKE. The PGMEVOKE routing entry matches the routing data that is started as a result of a program start request being received.

Any number of jobs can be active through this routing entry at any one time. The program that was sent on the program start request runs in the job because PGM(*RTGDTA) was specified. The job runs in storage pool 3 using class CMNCLASS in library QGPL.

Changing Routing Information

You can use the Change Routing Entry (CHGRTGE) command to make changes to a routing entry in the specified subsystem description. The parameters used on this command are the same as the parameters on the Add Routing Entry command. Refer to “Adding Routing Information” on page 11 for a description of these parameters. The SBSD and SEQNBR parameters are used to determine the routing entry to change.

The following command changes routing entry 1478 in the subsystem description ORDER that is found in library LIB5. The same program is used, but now it runs in storage pool 3 by using class SOFAST in library LIB6.

```
CHGRTGE   SBSD(LIB5/ORDER)  SEQNBR(1478)
  CLS(LIB6/SOFAST)  POOLID(3)
```

The following command changes routing entry 157 in the subsystem description PGMR found in library T7. The program INTDEV in library T7 is now called whenever this routing entry is selected. The other routing entry parameters are not changed.

```
CHGRTGE   SBSD(T7/PGMR)  SEQNBR(157)
  PGM(T7/INTDEV)
```

Removing Routing Information

You can use the Remove Routing Entry (RMVRTGE) command to remove a routing entry from the specified subsystem description. If a routing entry is to be removed from an active subsystem, jobs that used this entry cannot be active. The parameters used on this command tell the system which routing entry to remove. The subsystem description and library (SBSD) and the routing entry sequence number (SEQNBR) are the parameters that are used on this command. Refer to “Adding Routing Information” on page 11 for a description of the parameters.

The following command removes the routing entry 9912 from subsystem description PERT in library OR.

```
RMVRTGE   SBSD(OR/PERT)  SEQNBR(9912)
```

Handling Program Start Request Failures

When a program start request is received by an Operating System/400 (OS/400) subsystem, it attempts to start a job based on information that is sent with the program start request. The user’s authority to the system, existence of the requested program, and many other items are checked.

If the subsystem determines that it cannot start the job, the subsystem sends a message (CPF1269) to the QSYSMSG message queue or to QSYSOPR, or the configured message queue, when QSYSMSG does not exist. An example of this is when the requested program is not found. The CPF1269 message contains two reason codes (you can ignore reason codes of zero).

Note: The subsystem does not send the CPF1269 message with reason code 401 in all the instances in which it was sent in previous OS/400 versions. The following describes subsystem conditions, and their resulting CPF1269 message statuses.

No subsystem available
CPF1269 is issued once for each mode description on APPC device descriptions when there is no subsystem to accept the incoming request. Repetitive CPF1269 messages (which previously occurred in restricted state, for example) have been eliminated.

Incorrectly configured device or subsystem
One CPF1269 message is sent if a device or subsystem is incorrectly configured. Besides attempting to fix the configuration error, the user should vary off and vary on the device. This will permit another CPF1269 message to be received if the configuration error is not properly resolved. If the device is not varied off and varied on, a persistent configuration error may result in a failure for which no message is issued.

Inactive QSERVER subsystem
CPF1269 with reason code 413 is sent when the QSERVER subsystem is not active.

The *ICF Programming* book includes a description of reason codes and their meanings.

If only one nonzero reason code appears, that code is the reason the program start request was rejected.

At times, two nonzero reason codes may appear. This occurs when the OS/400 subsystem cannot determine whether the program start request is supposed to start a job in the System/36 environment or under the OS/400 subsystem.

One of the reason codes explains why the System/36 environment rejected the program start request. The other reason code explains why the AS/400 system rejected the program start request. When you receive two reason codes, you should determine where the job runs, and correct the problem.

Chapter 2. Working with Communications Configurations and Status

This chapter documents how to display the communications status of your applications programs, and how to display which communications configurations the application programs are using.

This chapter also contains information about managing configurations, including:

- Methods of saving and restoring configuration descriptions
- Commands for changing, activating and deactivating, and displaying the status of configuration descriptions
- Managing your messages

For information about TCP/IP network status, see the *TCP/IP Configuration and Reference* book.

Obtaining Application Communications Status Information

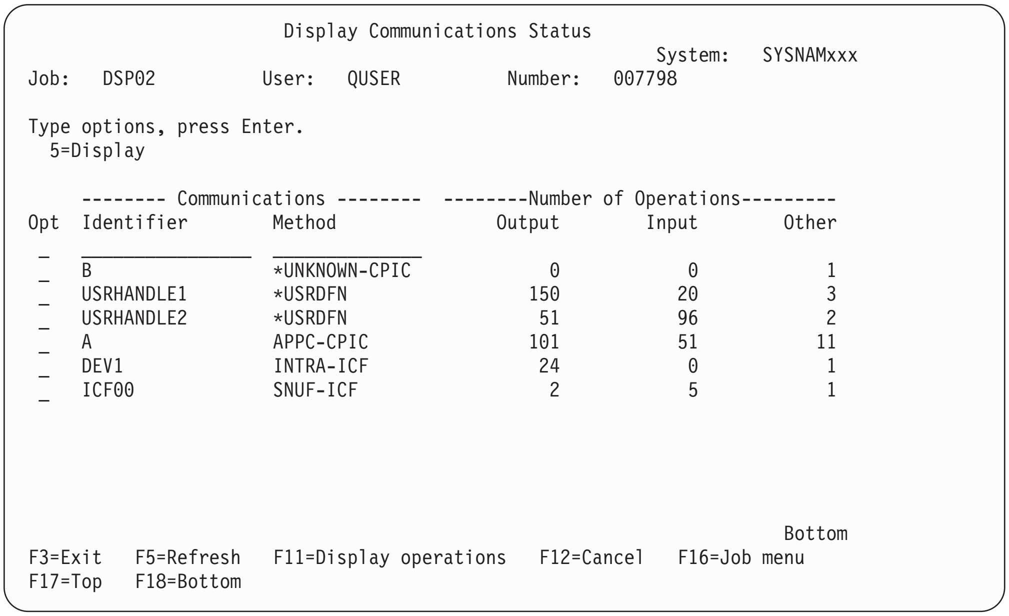
You can obtain communications status information about your applications by using the Display Job (DSPJOB) or the Work with Job (WRKJOB) commands. Communications status information can be obtained for:

- Active intersystem communications function (ICF) sessions
- Common Programming Interface (CPI) Communications conversations
- User-defined communications

Information that can be obtained includes the number of input, output, and other operations, as well as the current operation and most recently issued operation.

You can obtain the communications status information, by using the DSPJOB, or WRKJOB commands with OPTION(*CMNSTS) specified. You can also select option 17 from the Display Job or Work with Job displays. To print communications status information, specify OUTPUT(*PRINT). The Work with Configuration Status, Work with Active Jobs, and Work with APPN Status can access the Work with Job display.

Selecting option 17 from the Display Job or the Work with Job displays when communications identifiers are active brings up the following display:



```
                        Display Communications Status
                                                           System:   SYSNAMxxx
Job:   DSP02         User:   QUSER         Number:   007798

Type options, press Enter.
  5=Display

     -------- Communications --------  --------Number of Operations---------
Opt  Identifier        Method               Output        Input        Other
 _   ________________  ______________
 _   B                 *UNKNOWN-CPIC             0            0            1
 _   USRHANDLE1        *USRDFN                 150           20            3
 _   USRHANDLE2        *USRDFN                  51           96            2
 _   A                 APPC-CPIC               101           51           11
 _   DEV1              INTRA-ICF                24            0            1
 _   ICF00             SNUF-ICF                  2            5            1






                                                                         Bottom
F3=Exit   F5=Refresh   F11=Display operations   F12=Cancel   F16=Job menu
F17=Top   F18=Bottom
```

Opt

Type in the number of the appropriate option next to one or more entries. **5=Display** is the only valid option, and is used to display detailed information about the session. For ICF entries, the following kinds of displays can appear:

- Display CPI Communications
- Display APPC ICF Session
- Display Asynchronous ICF Session
- Display BSCEL ICF Session
- Display Finance ICF Session
- Display Intrasystem ICF Session
- Display Retail ICF Session
- Display SNUF ICF Session

For CPI Communications entries, the Display CPI Communications display appears. No options are supported for user-defined communications.

Communications Identifier

This is the identifier that is used by the application program.

CPI Communications conversations

The *conversation_ID* returned by the Initialize or Accept_Conversation calls and specified on all other calls.

ICF sessions

The program device name specified in the application for an active (acquired) ICF session.

User-defined communications

The communications handle that the application program is using.

Communications Method

This is the communications method being used.

ICF sessions

The ICF communications type used. Possible values are:

APPC-ICF

Advanced program-to-program communications

ASYNC-ICF
Asynchronous communications

BSCEL-ICF
Binary synchronous communications equivalence link

FINANCE-ICF
Finance communications

INTRA-ICF
Intrasystem communications

RETAIL-ICF
Retail communications

SNUF-ICF
SNA upline facility communications

CPI Communications conversations
The communications type that is used for the CPI Communications conversation. Possible values are:

***UNKNOWN-CPIC**
CPI Communications conversations in the **Initialize** *conversation_state*.

APPC-CPIC
CPI Communications conversations in any *conversation_state* other than **Reset** or **Initialize**.

User-defined communications
This field always shows *USRDFN for communications identifiers that are using user-defined communications support.

Output
This is the number of output operations that are performed on the communications identifier.

ICF sessions
The number of successful ICF write operations. It does not count write operations in which another function, such as invite, is specified and the length is 0. Cancel, cancel-invite, fail, negative-response, and request-write operations are not counted. Successful combined write/read operations are counted.

CPI Communications conversations
This count will be increased when a Send_Data completes successfully. However, CPI Communications will not increase this count for a Send_Data call with a *send_length* of zero and another function that is requested such as a *send_type* of CM_SEND_AND_CONFIRM. This count also will be increased when an Allocate call completes successfully.

User-defined communications
The number of calls to Send Data (QOLSEND).

Input
This is the number of input operations that are performed on the communications identifier.

ICF sessions
The number of successful ICF read operations that received data.

CPI Communications conversations

The number of Receive_Data calls that have completed successfully.

User-defined communications

The number of calls to Receive Data (QOLRECV).

Other

This field contains operations that are counted and are not included under output or input operations.

ICF sessions

The number of all other high-level language operations such as open/acquire, acquire, release, and close.

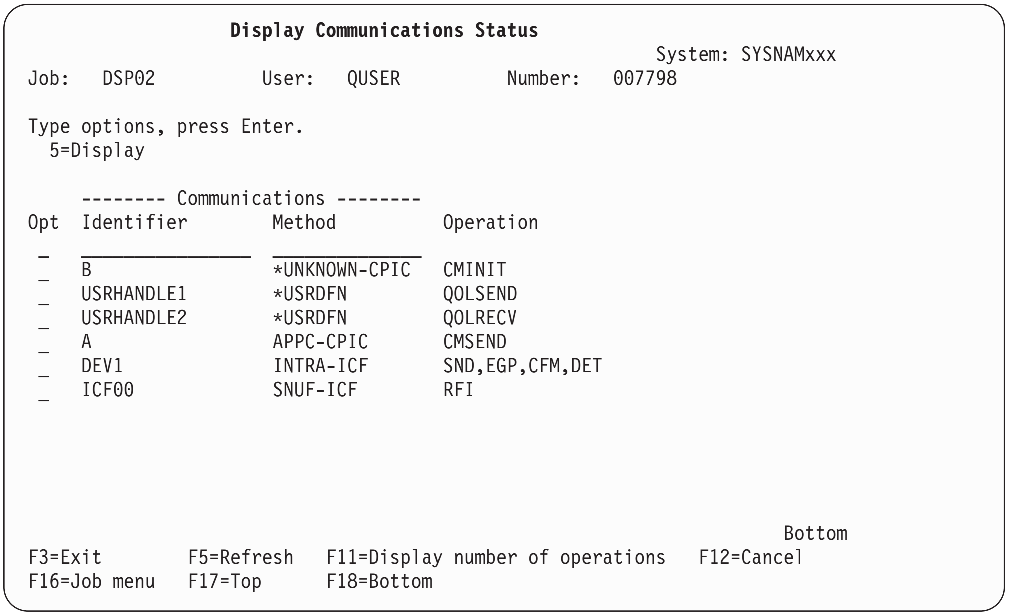
CPI Communications conversations

The number of all other successful CPI Communications calls that are not counted under output or input.

User-defined communications

The number of calls to Enable Link (QOLELINK) and Set Filter (QOLSETF).

When F11 is pressed on the first Display Communications Status display, a second Display Communications Status display appears. This display lists the current or last operation issued for the communications identifier.



```
                   Display Communications Status
                                                       System: SYSNAMxxx
Job:   DSP02         User:   QUSER         Number:   007798

Type options, press Enter.
  5=Display

     -------- Communications --------
Opt  Identifier       Method          Operation

 _   ________________ ______________
 _   B                *UNKNOWN-CPIC   CMINIT
 _   USRHANDLE1       *USRDFN         QOLSEND
 _   USRHANDLE2       *USRDFN         QOLRECV
 _   A                APPC-CPIC       CMSEND
 _   DEV1             INTRA-ICF       SND,EGP,CFM,DET
 _   ICF00            SNUF-ICF        RFI





                                                                   Bottom
F3=Exit        F5=Refresh   F11=Display number of operations   F12=Cancel
F16=Job menu   F17=Top      F18=Bottom
```

Operation

This is the current or last operation that is issued by the program. The operation displayed is dependent on the communications method that is used for the communications identifier.

ICF sessions

The current or last ICF operation that is issued by the program. The ICF function codes are the same as the function codes used by the Trace ICF function, and are described as follows:

ACQ Acquire

AWT Allow-Write

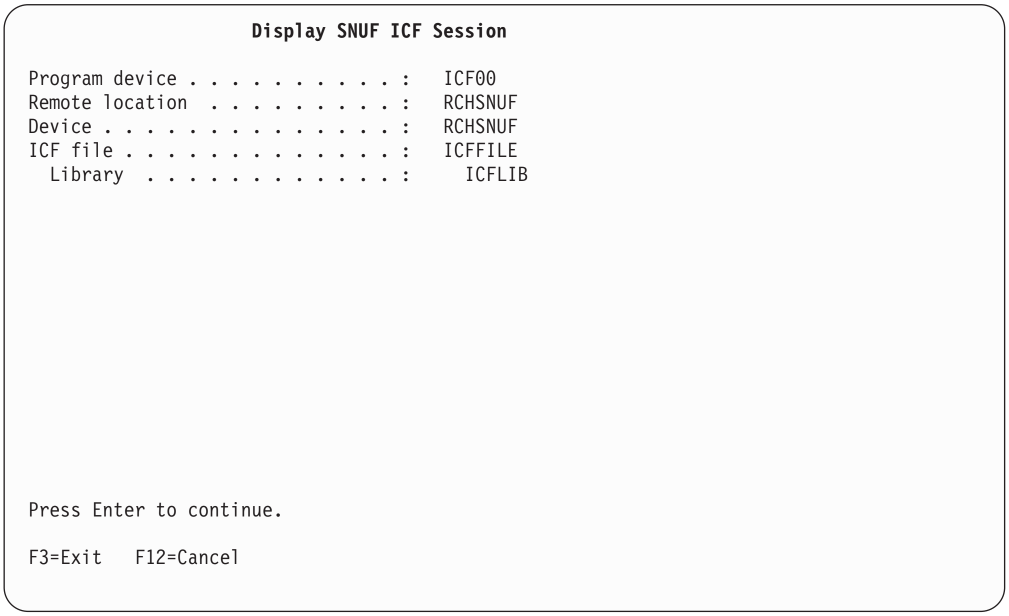
CFM Confirm

CLS	Close
CNI	Cancel-Invite
CNL	Cancel
DET	Detach
EGP	End-of-Group
EOA	End-of-Session-Abnormal
EOS	End-of-Session
EVK	Evoke
FAL	Fail
FMH	Function-Management-Header
FRC	Force-Data
GTA	Get-Attributes
INV	Invite
NRP	Negative-Response
OPN	Open with Acquire-Program-Device
RCF	Respond-to-Confirm
RCV	Receive
REL	Release
RFI	Read-From-Invited-Program-Devices
RST	Restore
RWT	Request-to-Write
SDV	Subdevice-Selection
SND	Send
SPD	Suspend
TMR	Timer

CPI Communications conversations
The current or last CPI Communications call that was issued, for example, CMINIT (Initialize_Conversation). Refer to the *CPI Communications Reference* book for more information on CPI Communications calls.

User-defined communications
The current or last user-defined communications call that was issued, for example, QOLELINK, QOLSETF, QOLSEND, or QOLRECV.

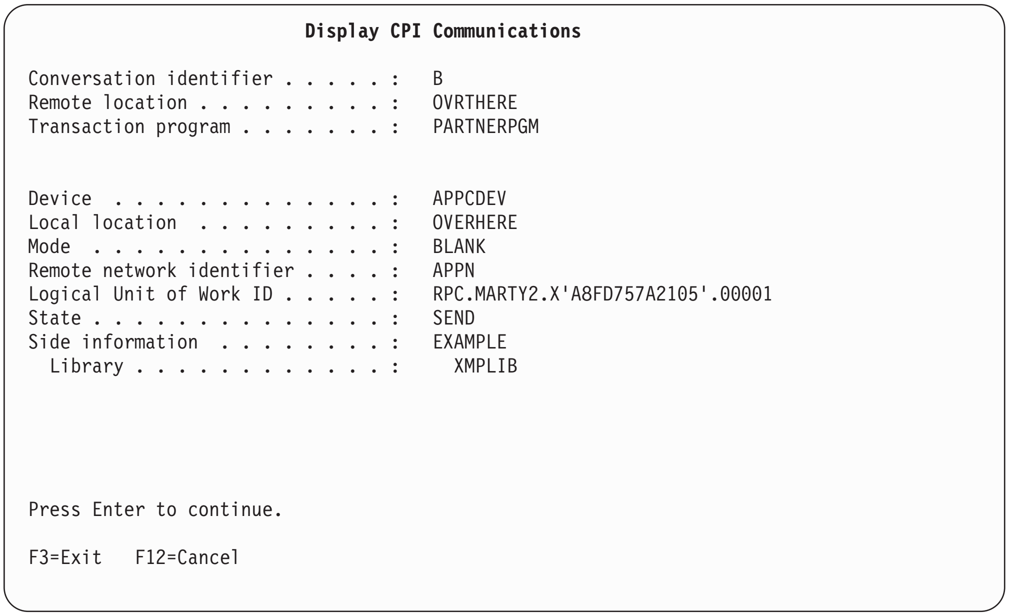
When the **5=Display** option is entered next to the ICF00 identifier in the first or second Display Communications Status display, the following display appears:



```
                     Display SNUF ICF Session

Program device . . . . . . . . . . . :   ICF00
Remote location  . . . . . . . . . . :   RCHSNUF
Device . . . . . . . . . . . . . . . :   RCHSNUF
ICF file . . . . . . . . . . . . . . :   ICFFILE
  Library  . . . . . . . . . . . . . :     ICFLIB










Press Enter to continue.

F3=Exit   F12=Cancel
```

The same information is displayed for the other ICF communication types. For APPC communications the local location name, mode, remote network identifier, logical unit of work ID, and state are also shown.

When the **5=Display** option is entered next to the B identifier in the first or second Display Communications Status display, the following display appears:



```
                          Display CPI Communications

Conversation identifier . . . . . :   B
Remote location . . . . . . . . . :   OVRTHERE
Transaction program . . . . . . . :   PARTNERPGM


Device  . . . . . . . . . . . . . :   APPCDEV
Local location  . . . . . . . . . :   OVERHERE
Mode  . . . . . . . . . . . . . . :   BLANK
Remote network identifier . . . . :   APPN
Logical Unit of Work ID . . . . . :   RPC.MARTY2.X'A8FD757A2105'.00001
State . . . . . . . . . . . . . . :   SEND
Side information  . . . . . . . . :   EXAMPLE
  Library . . . . . . . . . . . . :     XMPLIB





Press Enter to continue.

F3=Exit   F12=Cancel
```

Working with Communications Configurations

You describe your communications environment to the AS/400 system by creating a set of configuration descriptions. These configuration descriptions identify and describe the communications devices and services being used. The configuration descriptions available for AS/400 communications are as follows:

Line descriptions
Describe the physical line and the line protocol that are used for communications.

Controller descriptions
Describe physical remote controllers or provide logical representations of remote systems.

Device descriptions
Describe the characteristics of physical or logical remote devices.

Mode descriptions
Describe session limits and characteristics that are used for advanced program-to-program communications (APPC), Advanced Peer-to-Peer Networking (APPN), and high-performance routing (HPR).

Class-of-service descriptions
Describe node and transmission group characteristics that are used for APPN route selection.

Configuration lists
Contain entries that describe local and remote locations, pass-through information, and addresses that are used by a configuration.

Network interface descriptions
Describe the characteristics or protocol for communications with an Integrated Services Digital Network (ISDN), frame-relay network, or asynchronous transfer mode (ATM).

Connection lists
Contain entries describing local and remote locations in an ISDN network.

Network server descriptions
Describe the characteristics of an integrated PC server (IPCS).

NetBIOS descriptions
Describe the characteristics of a NetBIOS network that are connected to an IPCS.

Internetwork Packet Exchange (IPX) Descriptions
Describe the characteristics of IPX support. IPX descriptions are used for AS/400 IPX support and IPCSs using NetWare.

Communications Configuration Commands

You create and maintain communications configuration descriptions with commands from the system menus or from a command line. If you enter a “Work with...” command on the command line of any display, you are shown a Work with Configuration Description display. From here you can create, change, copy, rename, delete, display, print, or retrieve the CL source for a configuration description. You can also work with the configuration status. The available Work with Configuration Description commands are as follows:

- Work with Line Descriptions (WRKLIND)
- Work with Controller Descriptions (WRKCTLD)
- Work with Device Descriptions (WRKDEVD)
- Work with Mode Descriptions (WRKMODD)
- Work with Class-of-Service Descriptions (WRKCOSD)
- Work with Configuration Lists (WRKCFGL)
- Work with Network Interface Descriptions (WRKNWID)

- Work with Connection Lists (WRKCNNL)
- Work with Network Server Descriptions (WRKNWSD)
- Work with NetBIOS Descriptions (WRKNTBD)
- Work with IPX Descriptions (WRKIPXD)

Note: For more information on configuration descriptions and the Work with Configuration Description commands, refer to the *Communications Configuration* book.

You can also use the following communications configuration commands to maintain configuration descriptions:

Retrieve Configuration Source (RTVCFGSRC)
Retrieves the configuration source for one or more configuration descriptions

Save Configuration (SAVCFG)
Saves configuration descriptions onto a save media

Restore Configuration (RSTCFG)
Restores previously saved configuration descriptions from a save media back onto the system

Note: More information on saving and restoring configurations is in the *Communications Configuration* book, or the *Backup and Recovery* book.

In addition, you can access the status of configuration descriptions with the following two commands:

Work with Configuration Status (WRKCFGSTS)
You use the WRKCFGSTS command to work with the status of configuration descriptions in an interactive environment (see “Working with Configuration Status”).

Retrieve Configuration Status (RTVCFGSTS)
You use the RTVCFGSTS command in a CL program to retrieve the status of a configuration description (see “Retrieving Configuration Status” on page 32).

Working with Configuration Status

You can tell the system when to use the communications descriptions you have configured by varying the descriptions on or off. You can also display the status of the communications descriptions, which tells you the progress the system is making in performing the operations you have requested. This section explains these operations.

Varying a Configuration On or Off

After configuring your communications descriptions, you can vary these descriptions on or off using the Vary Configuration (VRYCFG) command. You can also use the Work with Configuration Status (WRKCFGSTS) display to vary on or off one or more configuration objects, including a network server, network interface, line, controller, and device.

Rules for varying on or off a specified object type:

- A line cannot be varied on:
  - Nonswitched ISDN data link control (IDLC) and PPP lines, until the network interface description is varied on.
  - If it is attached to a network server. Vary on the network server description, which varies on the attached lines.
- A line cannot be varied off:
  - Until all the attached controllers and devices are varied off.
  - If it is attached to a network server. Vary off the network server to vary off the attached lines.
- A controller cannot be varied on:
  - Leased (nonswitched) lines, if the line to which it is attached is varied off.
- A controller cannot be varied off:
  - If it is being used or allocated for use.
  - Until all the attached devices are varied off.
- A device cannot be varied on:
  - If the controller to which it is attached is varied off. This does not apply to tape and diskette devices because they are not attached to a controller.
- A device cannot be varied off:
  - If it is being used or allocated for use.

    Note: If you are using the FRCVRYOFF parameter, you can vary the APPC or SNPT device off even if it is in use.
- A network server cannot be varied off:
  - Until all attached devices and controllers are varied off. Varying off the server also varies off the attached line descriptions.
  - If any AS/400 clients have files open on the server.

  Note: Use the Work with Network Server Status (WRKNWSST) command (available from Work with Configuration Status display) to determine the status of network server session with other clients.
- A network interface description cannot be varied off:
  - Until all attached lines, controllers, and devices are varied off.

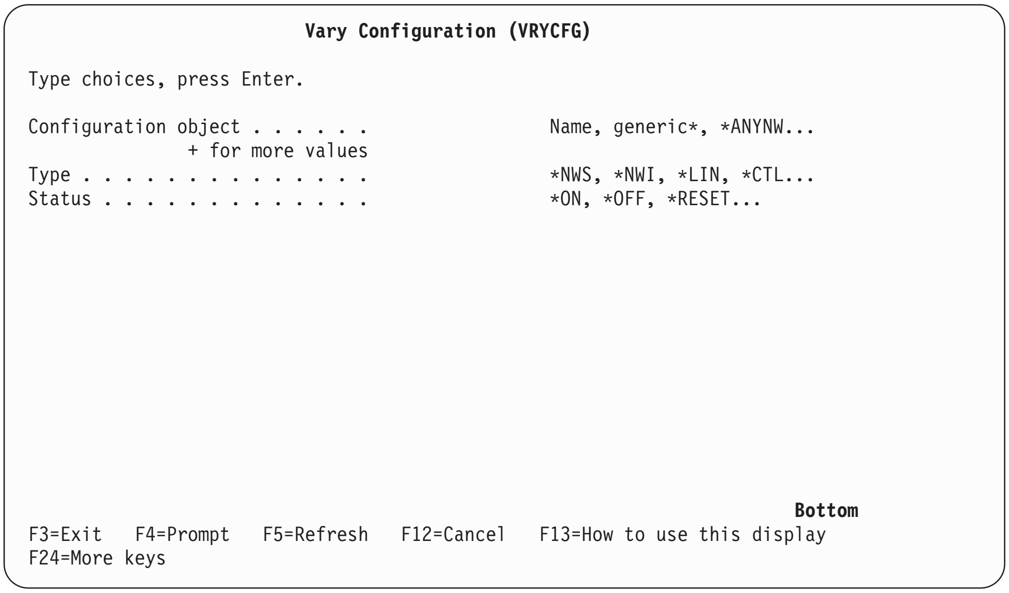
The VRYCFG also optionally resets the input/output processor (IOP) associated with the specified objects. An IOP can be a communications controller, local work station, or magnetic media controller. An IOP reset is valid only when varying on network interface descriptions, lines (except twinaxial data link controller (TDLC) lines), local work station controllers, tapes, and diskettes.

Note: When varying on a network server, an IOP reset is always done. The RESET parameter is ignored for network servers.

When configuring line descriptions, you can have more than one line description that describes the same physical resource. For example, you can have a switched line that you use for synchronous data link control (SDLC) communications during the daytime hours. You achieve this by varying on an SDLC line description created with the CRTLINSDLC command. In the evening hours, you can use the same line for binary synchronous (BSC) communications. You do this by varying off the SDLC line and varying on the BSC line description created with the CRTLINBSC command.

Only one of several line descriptions with the same resource name can be varied on at one time. By varying the objects on, you are instructing the system to use the objects for communications. If you vary the objects off, you are instructing the system not to use the objects for communications.

The following is an example of the Vary Configuration display:



```
                         Vary Configuration (VRYCFG)

Type choices, press Enter.

Configuration object . . . . . .                Name, generic*, *ANYNW...
             + for more values
Type . . . . . . . . . . . . . .                *NWS, *NWI, *LIN, *CTL...
Status . . . . . . . . . . . . .                *ON, *OFF, *RESET...




















                                                                     Bottom
F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display
F24=More keys
```

To use the prompt display for the VRYCFG command, you can either type the command and the associated parameters, or type the command and press F4 (Prompt). On the prompt display, you enter the following information:

Required Parameters

Configuration object (CFGOBJ)

The name of the network interface, network server, line, controller, or device description to be varied on or off, or a list of names of configuration objects of the same description type. For example, a list of the names of network interfaces, network servers, lines, devices, or controllers.

You can type the names of up to 256 configuration objects. Type a + (plus sign) on the second line for additional names and press the Enter key. A second display appears on which you can type many object names.

***APPN**

All objects that use Advanced Peer-to-Peer Networking (APPN) will be varied on or off. This value is only valid if CFGTYPE is *CTL or *DEV.

***ANYNW**

This value varies all AnyNet controllers. It is only valid if CFGTYPE is *CTL.

***PRVCFGTYPE**

Process all objects that were processed the last time this command was run in the job for the specified configuration object type.

generic-configuration-description-name:*

Specify the generic name of the configuration description name. A generic name is character string of one or more characters that are followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters.

Notes:

1. If no generic names are specified, objects are processed in the order the names were entered for this parameter.
2. If one or more generic names are specified, all objects matching the Name that is specified will be processed in alphabetical order.
3. You cannot specify generic names if the CFGTYPE parameter is *MLBRSC.
4. You must take care when varying objects that use generic names or special values so that unintentional processing of objects does not occur. For example, using the generic name CTL* will cause all objects beginning with the letters CTL to be processed. This occurs even though the user may have only intended some of these objects to be processed.

Configuration type (CFGTYPE)
The type of configuration to be varied on or off.

***NWS** Network server configuration

***NWI** Network interface configuration

***LIN** Line configuration

***CTL** Controller configuration

***DEV** Device configuration

***MLBRSC**
Tape media library (RISC only)

Note: This value is valid for tape media library devices only.

Status (STATUS)
The status of the configuration object such as vary on (*ON) or vary off (*OFF). When STATUS (*OFF) is used, all devices must be varied off before the attached controller can be varied off. All controllers must be varied off before the associated line can be varied off (this can be done by using the RANGE parameter). All lines must be varied off before the associated network interface can be varied off (this can be done by using the RANGE parameter). A device can be varied off only when it is not allocated to an active job. Values for the STATUS parameter are:

***ON** The object is varied on.

***OFF** The object is varied off.

For RISC only, the STATUS parameter can be set to the following values once the media library device is varied on:

***RESET**
The drive resources of the tape media library device are reset.

Note: The drive resources must be specified on the RSRCNAME parameter.

***ALLOCATE**
The drive resources of the tape media library device are allocated for use only by this system. If the library device is shared by multiple systems, other systems cannot use these drives while this device description is varied on.

Note: The drive resources must be specified on the RSRCNAME parameter.

***UNPROTECTED**
The drive resources of the tape media library device can be used by all systems that share this library device.

Note: This value is not recommended. When the drive resources are in unprotected mode, each system can access the resource at the same time. Unpredictable results can occur.

***DEALLOCATE**
The drive resources of the tape media library device are deallocated for this system. If the tape media library is shared by multiple systems, this system can not use the drives, but can be used by other systems.

Note: The drive resources must be specified on the RSRCNAME parameter.

Note: If an APPC device is allocated to an active job when a vary off is requested, message (CPA2610) is sent to the QSYSOPR message queue or the configured message queue. The operator then can do one of the following:

- Reply C (Cancel the vary off request) and allow the job to continue
- Reply G (Go) to continue the vary off request immediately

UNBIND requests occur for all sessions associated with the device prior to allowing the jobs associated with the sessions to complete processing. This only applies to APPC device descriptions (which includes APPC over TCP/IP). You can eliminate the inquiry message by using the Force Vary Off (FRCVRYOFF) parameter. However, that FRCVRYOFF should be used with care, and only when absolutely necessary. FRCVRYOFF processing bypasses normal communications take down, and is intended only for cases where the jobs must end.

Range (RANGE) (optional parameter)
Specifies which configuration elements are varied on or off.

Notes:

1. Device descriptions: Because devices do not have downline attached objects, value *NET is not valid
2. Switched line descriptions: When varying on elements, value *NET is not valid. When varying off elements and specifying value *NET, the line, and its downline attached objects are varied off.
3. Network interface descriptions: Specifying value *NET varies on all nonswitched attachments and varies off all nonswitched attachments.
4. Network server descriptions: The parameter is ignored in this instance. Varying on or off a network server also varies the attached lines.

***NET** All downline configuration objects are varied on or off. Downline objects are the following:

- Devices that are attached to a controller
- Controllers that are attached to a communications line
- Lines that are attached to a network interface in a communications configuration

This is the default value.

Note: *NET is always specified for network server description.

***OBJ** Only the specified object is varied on or off. This value is ignored for network server configurations.

Vary on wait (VRYWAIT) (optional parameter)

Specifies synchronous or asynchronous vary on of the network. You should specify a wait time (synchronous vary on) when an application program requests a session that uses the communications descriptions after varying on the communications descriptions. For example, specify a wait time when an open or acquire operation to the ICF file follows a batch program varying on a network server, network interface, line, controller, or device description. Values for the VRYWAIT parameter are:

Notes:

1. If the VRYWAIT parameter is specified on the VRYCFG command for a line description that is not Ethernet, token-ring, DI, X.25, or switched SDLC, BSC, or Async, the parameter is accepted, but ignored.
2. If the VRYWAIT parameter is specified on the VRYCFG command for a server description, the parameter is accepted, but ignored.

***CFGOBJ**
The VRYWAIT value specified in the line, network, or server interface description. This is the default value.

Note: Use *CFGOBJ when varying on Network Server Descriptions.

***NOWAIT**
The system does not wait for vary on completion. The line, network, or server interface description interface varies on asynchronously.

vary-on-wait
Specify a value from 15 to 180 seconds in one-second intervals. The system waits until the line is varied on, or until the specified time passes, before ending the VRYCFG command.

Notes:

1. If ONLINE(*YES) is specified, specifying a wait time in the line description will increase the system IPL time. It is increased by the amount of time it takes to synchronously vary on the line or reach the wait-time value.
2. The time required to vary on a line is the time it takes to:
   - Put tasks in place to manage the line
   - Activate the communications I/O processor (IOP) (including downloading the IOP model-unique Licensed Internal Code)
   - Establish the communication tasks and processes

   Normal vary-on time ranges from 5 to 45 seconds or longer depending on factors such as the system and line protocol.

Asynchronous Vary Off (ASCVRYOFF) (optional parameter)

Specifies whether the vary off process is synchronous or asynchronous. When you specify STATUS (*ON), this parameter is not allowed.

***NO** The vary off process is synchronous. This is the default value.

***YES** The vary off process is asynchronous.

Reset (RESET) (optional parameter)

An optional parameter to reset the communications controller associated with a network interface or line description. A reset places the communications controller in a usable state and can be used to recover from communications controller program failures. *YES is always specified for network server description.

Note: Any multiple function input/output processor that has DASD on it cannot be reset with this parameter. For example, the multiple function input/output processor on a 9404 System Unit may not be reset because the input/output processor controls the service processor, disk devices, tape, diskette, and communications. An initial program load (IPL) of the system must be done to reset this input/output processor.

A reset can be done *only* when varying on an object. All of the network interfaces, network servers, and line descriptions associated with the communications controller must be in a varied off state before doing a reset for a communications controller associated with a network interface, network server, or line. If all of the network interfaces, network servers, and line descriptions associated with a communications controller are not varied off, and RESET(*YES) is specified, the system rejects the Vary Configuration (VRYCFG) command.

***NO** The associated communications controller is not reset. This is the default value.

***YES** The associated communications controller is reset. *YES is always specified for network servers.

Resource name (RSRCNAME) (optional parameter)

Specifies the resource name of the drive within the media library to be reset or reallocated.

Reset configuration file (RESETCFGF) (optional parameter)

Specifies whether to reset the configuration file that is associated with a *BASE or *NETWARE network server description. If there is no configuration file that is associated with *BASE or *NETWARE network server description, this parameter is ignored. This parameter is valid only when CFGTYPE is *NWS.

***NO** The configuration file is not reset.

***YES** The configuration file is reset.

Force Vary Off (FRCVRYOFF) (optional parameter)

Specifies whether inquiry messages for active jobs will be suppressed or whether special processing regarding device locks will be performed. When you specify STATUS(*ON), this parameter is not allowed. FRCVRYOFF should be used with care, and only when absolutely necessary. FRCVRYOFF processing bypasses normal communications take down, and you should only use it for situations where the job must end.

***NO** Inquiry messages will be presented for active jobs.

***YES** Inquiry messages will be suppressed for active jobs, and the jobs will end.

***LOCK**

For devices other than APPC and Intra, an attempt will be made to get a lock on the device description no matter what its current status might be. If the lock is successfully obtained, it will transfer to the system job that is assigned to hold the device description lock. This occurs when the device is in a varied off state. If the device is in a state other than varied off, an attempt to vary off the device description will also be made.

The following are a list of ways you can use the VRYCFG command .

Example 1: Varying On the Network Interface and Downline Attachments.

```
VRYCFG  CFGOBJ(NWI1)  OBJTYPE(*NWI)  STATUS(*ON)
```

Example 2: Varying Off the Line and Attached Downline Objects.

```
VRYCFG  CFGOBJ(LINE1)  OJBTYPE(*LIN)  STATUS(*OFF)
```

Example 3: Varying on the Controller.

```
VRYCFG  CFGOBJ(CONTROLLER1)  CFGTYPE(*CTL)  STSTUS(*ON)
  RANGE(*OBJ)
```

Example 4: Varying on the Device.

```
VRYCFG  CFGOBJ(DEVICE1)  CFGTYPE(*DEV)  STATUS(*ON)
  RANGE(*NET)
```

This command only varies on the device.

Note: The RANGE parameter value has no effect on devices.

Example 5: Varying on the Line and Resetting the IOP.

```
VRYCFG  CFGOBJ(LINE1)  CFGTYPE(*LIN)  STATUS(*ON)
  RANGE(*OBJ)  RESET(*YES)
```

Example 6: Using Line Description Value for Wait Time.

```
VRYCFG  CFGOBJ(LINE1)  OJBTYPE(*LIN)  STATUS(*ON)
  RANGE(*OBJ)  VRYWAIT(*CFGOBJ)
```

This command varies on only the line and uses the vary wait time value specified in the line description for LINE1.

Example 7: Using 80 seconds as Vary Wait Time.

```
VRYCFG  CFGOBJ(LINW1)  CFGTYPE(*LIN)  STATUS(*ON)
  RANGE(*OBJ)  VRYWAIT(80)
```

Example 8: Varying on a Network Server.

```
VRFCFG  CFGOBJ(NWS1)  CFGTYPE(*NWS)  STATUS(*ON)
```

This command varies on the network server description named SERVER1 and its attached line descriptions. The vary on wait value specified in the network server description is used.

Note: The RANGE and RESET parameters are ignored for network servers if specified.

Example 9: Resetting Drives Within a Media Library.

```
VRYCFG  CFGOBJ(MYLIBRARY)  CFGTYPE(*MLBRSC)
  STATUS(*RESET)  RSRCNAME(TAP01 TAP02)
```

This command resets the drives TAP01 and TAP02 within the media library device MYLIBRARY. You must vary on MYLIBRARY to perform this action.

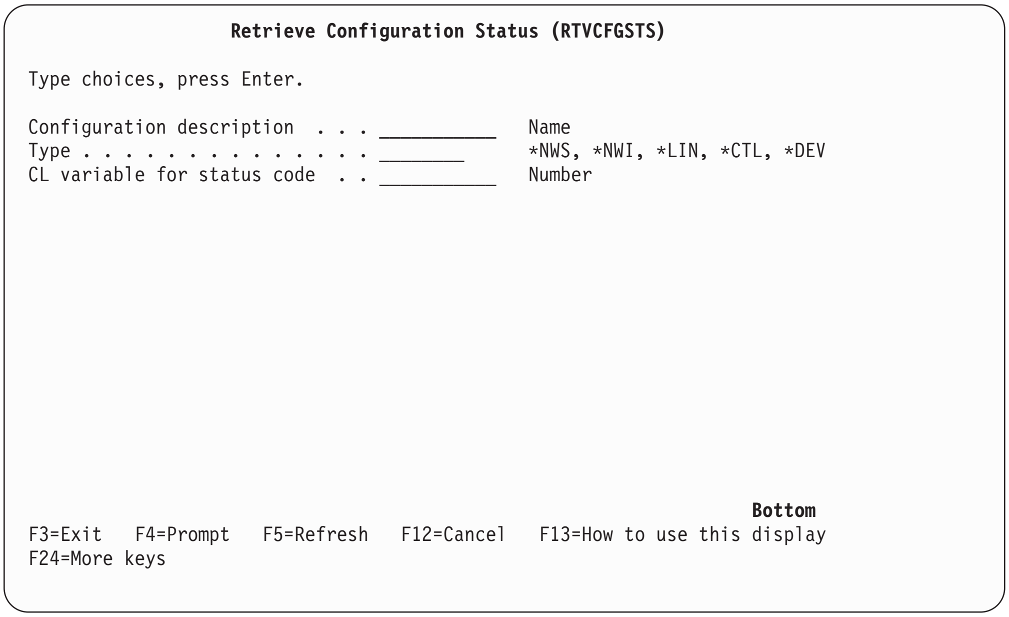
Retrieving Configuration Status

The Retrieve Configuration Status (RTVCFGSTS) command allows you to retrieve configuration status from object types (network interface, network server, line, controller, and device), and place the configuration status into a CL program variable. CL programs can only use this command. You cannot use it from the AS/400 command line.

Note: You may use the QDCRCFGS API within application programs to retrieve the configuration status. See the *System API Reference: Configuration APIs* book for more information.

Communications applications can react quickly and easily to a configuration status with direct access to the objects. For example, a user can retrieve the status of various objects to determine whether they are varied on rather than varying on the objects to determine their status.

If you use the RTVCFGSTS command, you can either type the command and the associated parameters, or you can type the command and press F4 (Prompt) to use the prompt display for this command. The following is an example of the Retrieve Configuration Status display:



```
                 Retrieve Configuration Status (RTVCFGSTS)

 Type choices, press Enter.

 Configuration description  . . . __________   Name
 Type . . . . . . . . . . . . . . _______      *NWS, *NWI, *LIN, *CTL, *DEV
 CL variable for status code  . . __________   Number

















                                                                     Bottom
 F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display
 F24=More keys
```

On the prompt display, you enter the following information:

Configuration description (CFGD)
The name of the configuration description.

Type (CFGTYPE)
The type of configuration to be retrieved. This is a required parameter.

***NWS** Network server configuration

***NWI** Network interface configuration

***LIN** Line configuration

***CTL** Controller configuration

***DEV** Device configuration

Status code (STSCDE)

The name of the control language (CL) variable receiving the status code. This CL variable must be declared as a five-position decimal variable without decimal positions.

The following program example uses these parameters:

```
DCL VAR(&STSCODE)  TYPE(*DEC)    LEN(5 0)

RTVCFGSTS CFGD(ND01) CFGTYPE(*LIN)
          STSCDE(&STSCODE)
```

Table 3 shows the possible status codes. See “Status Description of Network Servers, Network Interfaces, Lines, Controllers, and Devices” on page 41 for more information.

Table 3. Status Codes for Network Server, Network Interfaces, Lines, Controllers, and Devices

Status Code Number	Status Name	Status Code Description
0	Varied Off (VARIED OFF)	The system is not using the description.
10	Vary Off Pending (VARY OFF PENDING)	The description is being varied off. During this time, the system ends the operations that manage the resource or communicate with data circuit-terminating equipment (DCE).
20	Vary On Pending (VARY ON PENDING)	The description is being varied on. During this time, the system begins the operations that manage the resource, download Licensed Internal Code to an input/output processor, and communicate with the DCE.
30	Varied On (VARIED ON)	The functions that manage the network interface, line, controller, or device have been put into place by the system. This status code does not apply to network server.
32	Vary On/Connect Pending (VARYON/CNN PENDING)	The first of a pair of OptiConnect controllers is varied on but its attached device is not yet in a varied on state.
40	Connect Pending (CONNECT PENDING)	Valid only for switched SDLC, bisynchronous, X.25, IDLC, PPP, or asynchronous lines. The line is in this status while waiting for the switched connection to be established.
50	Sign On Display (SIGN ON DISPLAY)	Valid only for display devices. Either the system is preparing the device to receive the sign-on display, sending the sign-on display, or the sign-on display is at the display station.
51	Active/Connect Pending (ACTIVE/CONNECT PENDING)	The first of a pair of OptiConnect controllers and its attached device are varied on and waiting for the OptiConnect path to be established.
60	Active (ACTIVE)	The object is successfully placed in VARIED ON status. In addition, one of the following is true: • For network interfaces, one or more attached lines is in a VARY ON PENDING or higher status. • For lines, one or more attached controllers is in a VARY ON PENDING or higher status. • For controllers, one or more attached devices is in a VARY ON PENDING or higher status. • For devices, active status varies, depending on the type of device. For network servers, the functions that manage the network server have been put into place by the system. The network server is automatically in ACTIVE status.

Table 3. Status Codes for Network Server, Network Interfaces, Lines, Controllers, and Devices (continued)

Status Code Number	Status Name	Status Code Description
63	Active Reader (ACTIVE READER)	The device is in use by a spool reader
66	Active Writer (ACTIVE WRITER)	The device is in use by spool writer.
70	Held (HELD)	Valid only for device descriptions. The user or system held the communications device to prevent it from communicating. Use the Release Communications Device (RLSCMNDEV) command to release the device.
80	Recovery Pending (RCYPND)	Error recovery is pending for the network interface, line, controller, or device. A message indicating what error occurred appears on the QSYSOPR message queue or the configured message queue.
90	Recovery Cancel (RCYCNL)	Error recovery is canceled for the network interface, line, controller, or device. An error occurred and the operator gave a C (cancel error recovery) reply to a message, or the operator used one of the following commands to end the error recovery process: End Network Interface Recovery (ENDNWIRCY), End Line Recovery (ENDLINRCY), End Controller Recovery (ENDCTLRCY), or End Device Recovery (ENDDEVRCY).
95	System Request (SYSTEM REQUEST)	The display device has been requested by the system, and its associated job has been suspended. This occurs as a result of a user pressing the System Request key.
100	Failed (FAILED)	An error occurred for the network interface, network server, line, controller, or device that can be recovered only by varying off and on again.
103	Failed Reader (FAILED READER)	An error occurred for the device while in use by a spool reader.
106	Failed Writer (FAILED WRITER)	An error occurred for the device while in use by a spool writer.
107	Shutdown (SHUTDOWN)	Shutdown is pending for a network server description.
110	Diagnostic Mode (DIAGNOSTIC MODE)	The network interface, network server, line, controller, or device is being used by problem analysis procedures to diagnose problems. The resource cannot be used by other users.
111	Damaged (*DAMAGED)	The network interface, network server, line, controller, or device description is damaged. This is a system error condition. Information showing when this damage occurred exists in the history log (QHST) or in the Vertical Licensed Internal Code (VLIC) log. You must delete the description and create it again before it can be used. The VLIC log information can be used when reporting a problem to IBM service.
112	Locked (*LOCKED)	The actual status of the resource cannot be determined because another job has an exclusive lock on the description. Try again at a later time or use the Work with Object Lock (WRKOBJLCK) command to determine which job has the lock on the description.
113	Unknown (*UNKNOWN)	The status indicator of the description cannot be determined. This is a system error condition. Use the Dump Object (DMPOBJ) command to get the attributes and contents of the description to a spooled printer file and contact your IBM representative or your IBM-approved remarketer.

The following exceptions can be signaled by the RTVCFGSTS command:

- `CPF9801`: Object not found

- `CPF9802`: Not authorized to object

Using the Work with Configuration Status Command

The Work with Configuration Status (WRKCFGSTS) command provides menus for configuration status procedures. When you use this command, the Work with Configuration Status display appears (refer to the Work with Configuration Status displays shown on page 39).

To use this command, you must specify the information for the following parameters:

Configuration type (CFGTYPE)
The type of configuration description for which you want the status:

***NWS** Network server status

***NWI** Network interface status

***LIN** Line status

***CTL** Controller status

***DEV** Device status

Configuration descriptions (CFGD)
The configuration descriptions you want displayed on the Work with Configuration Status display.

***ALL** This is the default. All valid descriptions.

***APPC**
Advanced program-to-program communications controller and devices.

***ASYNC**
The asynchronous lines.

***BSC** The bisynchronous lines.

***CMN** The communications controllers and devices. *CMN does not include remote work station controllers, virtual controllers, and devices.

***DDI** The distributed data interface (DDI) lines.

***DKT** The diskette devices.

***DSP** The local, remote, and virtual display station devices.

***ELAN**
Ethernet local area network (LAN) lines.

***FAX** The facsimile (FAX) communication lines.

***FNC** Finance controllers and devices.

***FR** frame-relay (FR) network interfaces and lines.

***HOST**
SNA host controllers and devices.

***IDLC** ISDN data link control (IDLC) lines.

***INTRA**
Intrasystem devices.

***ISDN** Integrated services digital network interfaces.

***LANPRT**
The local area network (LAN) printer devices.

***LCLDSP**
The local display station devices.

***LCLPRT**
The local printer devices.

***LOC** The communications devices whose remote location name matches the name specified for the RMTLOCNAME parameter. A value other than *NONE must be specified for the RMTLOCNAME parameter if you specify *LOC for this parameter.

***LWS** The local work station controllers.

***MLB** Both optical and tape media library devices (RISC only).

***NET** The network lines.

***OPT** Optical devices (RISC only).

***OPTMLB**
Optical media library devices (RISC only).

***PPP** Point-to-point protocol lines.

***PRT** The local, remote, and virtual printer devices.

***RMTDSP**
The remote display station devices.

***RMTPRT**
The remote printer devices.

***RTL** Retail controllers and devices.

***RWS** The remote work station controllers.

***SDLC**
The synchronous data link control (SDLC) lines.

***SNPT**
The SNA Pass-through devices.

***TAP** The tape devices.

***TAPMLB**
Tape media library devices (RISC only).

***TDLC**
The twinaxial data link control (TDLC) lines.

***TRLAN**
The token-ring lines.

***VRTDSP**
The virtual display station devices.

***VRTPRT**
The virtual (pass-through) printer devices.

***VWS** The virtual (5250 display station pass-through) work station controllers.

***WLS** The wireless lines.

***WS** The local, remote, and virtual work station controllers.

***X25** The X.25 lines.

generic-description-name
The descriptions of the names starting with a certain character string are displayed.

description-name
The named descriptions and any upline or downline attachments.

Remote location name (RMTLOCNAME)
The remote location name of those communications device descriptions that you want. This parameter shows only the device descriptions with the specified remote location names. The remote location name that you enter here is the same as the name that you specify as the remote location name parameter for one of the following devices:

- Advanced program-to-program communications (APPC)
- Asynchronous communications
- Binary synchronous communications (BSC)
- Finance
- Host
- Intrasystem
- Retail
- SNA upline facility (SNUF)

Note: This parameter is required if CFGD(*LOC) is specified. It is not a valid parameter for any other value of the CFGD parameter.

***NONE**
The descriptions displayed are not for communications devices with a specific location name. *NONE must be specified if the CFGD parameter is anything other than *LOC. This is the default value.

generic-remote-location-name
Specify the generic location name of the communications devices for which the status is displayed.

remote-location-name
Specify the remote location name of the communications devices for which the status is displayed.

Note: When the System Request (Sys Req) key is used on a display to get a second job, WRKCFGSTS shows SYSREQ as the status of the first job and SIGNON DISPLAY or ACTIVE as the status of the second job. The Retrieve Configuration Status (RTVCFGSTS) command returns an ACTIVE status for the device in this case. This happens because the WRKCFGSTS shows the status of the job, while RTVCFGSTS returns the status of the device.

RANGE
Specifies whether downline or upline attached configuration descriptions are shown.

***NET:** If a single object name is specified for the configuration description (CFGD) parameter, both downline and upline descriptions are shown. If a special value or generic name is specified for the CFGD parameter, downline descriptions are shown.

***OBJ:** Only objects of the type that are specified by the configuration description type (CFGTYPE) parameter are shown.

STATUS

Specifies the status values for the list descriptions that are shown.

***ALL:** All descriptions are included in the list regardless of their status.

***ACTIVE:** All descriptions with an active status are shown.

***FAILED:** All descriptions with a failed, recovery, damaged, or unknown status are shown.

***VARYOFF:** All descriptions with a varied off, or vary off pending status are shown.

***VARYON:** All descriptions that do not have a varied off or vary off pending status are shown.

Using the Work with Configuration Status Display

You can reach the Work with Configuration Status display through the following methods:

- Work with Configuration Status (WRKCFGSTS) command
- Work with Hardware Resources (WRKHDWRSC) command
- Work with APPN Status (WRKAPPNSTS) command
- Menus (for example, the Configure Devices and Communications menu, option 4, Work with Configuration Status)
- Displays:
  - Work with Network Server Descriptions (WRKNWSD)
  - Work with Network Interface Descriptions (WRKNWID)
  - Work with Line Descriptions (WRKLIND)
  - Work with Controller Descriptions (WRKCTLD)
  - Work with Device Descriptions (WRKDEVD)

The Work with Configuration Status display shows status information for network servers, network interfaces, lines, controllers, and devices, and for jobs associated with devices. The display can be for a remote location or for one or more network servers, network interfaces, lines, controllers, or devices. Configuration descriptions are displayed for each network server, network interface, line, controller, or device description selected. Attached configuration descriptions are indented under the object to which they are attached (refer to the Work with Configuration Status display that follows).

If the status is displayed for a specific controller, upline line descriptions as well as downline device descriptions are displayed. Status displays for any collection of controllers or devices. Controllers whose names start with a specified (generic) character string, retail controller, or finance controller, show only downline attachments. The status for a remote location shows devices and modes for the specified location.

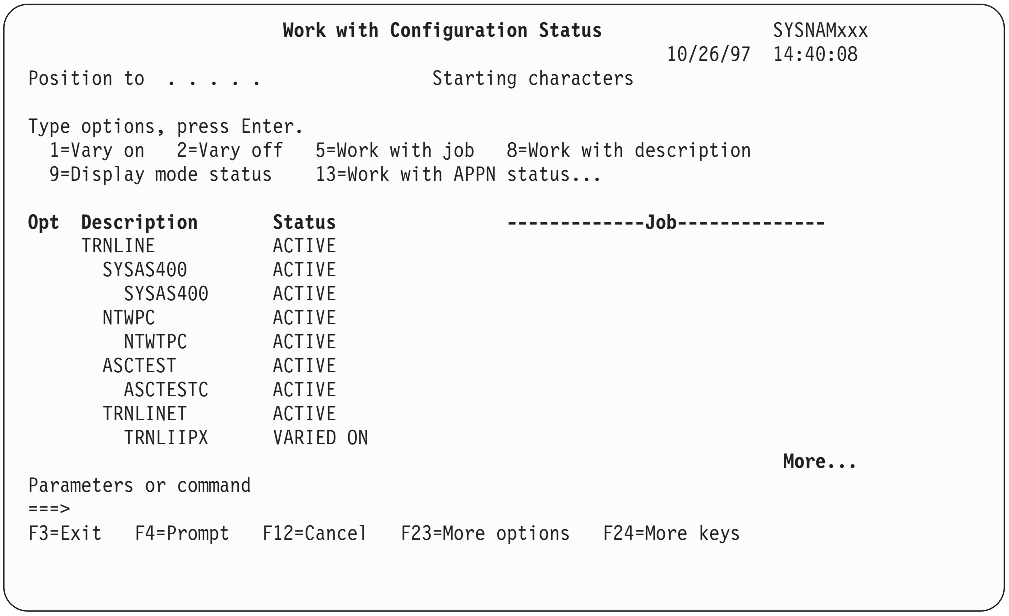
Use the F15 key from this display to work with network server status. The Work with Network Server Status display allows you to do the following:

- Display users
- Work with aliases

- Work with network server sessions
- Display statistics associated with each server
- Restart a server

See the *OS/2 Warp Server for AS/400 Administration* book more information about network servers.

The following is an example of the Work with Configuration Status display and an explanation of the options on the display:



```
                        Work with Configuration Status               SYSNAMxxx
                                                           10/26/97  14:40:08
Position to  . . . . .               Starting characters

Type options, press Enter.
  1=Vary on   2=Vary off   5=Work with job   8=Work with description
  9=Display mode status    13=Work with APPN status...

Opt  Description       Status                -------------Job--------------
     TRNLINE           ACTIVE
       SYSAS400        ACTIVE
         SYSAS400      ACTIVE
       NTWPC           ACTIVE
         NTWTPC        ACTIVE
       ASCTEST         ACTIVE
         ASCTESTC      ACTIVE
       TRNLINET        ACTIVE
         TRNLIIPX      VARIED ON
                                                                       More...
Parameters or command
===>
F3=Exit   F4=Prompt   F12=Cancel   F23=More options   F24=More keys
```

The options are:

Position to
Type the starting characters (or full name) of the description name to which you want the list positioned. The list can be positioned only on the highest level of name for which the status is displayed. For example, when you want to look at the status for the lines, the list can be positioned to start with names that are in the list.

Options
To select an option, enter the associated number in the Option column to the left of the description name. (For example, to Vary On description AAAA, enter a 1 in the Option column next to description AAAA.) Not all of the options are shown on the display at one time. Press F23 (More options) to display additional options. A list of the options you can select follows:

1. Vary on: Varies on the network server, network interface, line, controller, or device and all of the attached lines, controllers, and devices. This is the same as using the Vary Configuration (VRYCFG) command with STATUS(*ON).

2. Vary off: Varies off the network server, network interface, line, controller, or device and all of the attached lines, controllers, and devices. This is the same as using the Vary Configuration (VRYCFG) command with STATUS(*OFF). You may vary off devices only if they are not allocated to an active job. Jobs can be canceled if you need to vary off a device by using the Work with job option.

3. Hold device: This option allows you to stop the communications device from receiving or transmitting any data.

4. End recovery: This option allows you to cancel automatic error recovery for the object.

5. Work with job: This option shows another display on which you can perform operations related to this job, such as canceling or displaying the job associated with the device. This is the same as using the Work with Job (WRKJOB) command.

6. Release device: This option allows you to release the communications device for receiving or transmitting data.

7. Resume recovery: This option allows you to start automatic error recovery again for the object.

8. Work with description: This option runs the Work with Network Server Descriptions (WRKNWSD), Work with Network Interface Descriptions (WRKNWID), Work with Line Descriptions (WRKLIND), Work with Controller Descriptions (WRKCTLD), Work with Device Descriptions (WRKDEVD), or Work with Mode Descriptions (WRKMODD) commands depending on the type of object selected.

9. Display mode status: This option runs the Display Mode Status (DSPMODSTS) command for the device or mode.

11. Display connection status: This option shows the connection status for the network device. For more information on this option, see“Displaying Connection Status” on page 61.

12. Work with LAN adapter: This option allows you to work with LAN adapters located on the token ring. For more information on this option, see the *OS/2 Warp Server for AS/400 Administration* book.

13. Work with APPN status: This option allows you to work with the status of APPN and HPR network sessions, and with the RTP connections on your local system. See “Work with APPN Status” on page 60 for additional information.

Description
The names of existing network interface, line, controller, device, and mode descriptions.

Type
The type of description.

***NWS** Network server description

***NWI** Network interface description

***LIN** Line description

***CTL** Controller description

***DEV** Device description

Status
The status of the specified network servers, network interfaces, lines, controllers, devices, or modes. See “Status Description of Network Servers, Network Interfaces, Lines, Controllers, and Devices” on page 41 for an explanation of the type of status information that appears for the network servers, network interfaces, lines, controllers, and devices.

Job

Consists of three parts: The name of the job using the device (or mode if advanced program-to-program communications (APPC)), the user profile under which the job is running, and the system-unique number of the named job. This field is blank if no job is using the device or mode, and it is always blank for line and controller entries. Only up to 255 user jobs per APPC device are shown.

Pass-through device

The SNA pass-through device that is associated with the device description.

Additional functions are available through function keys that do not appear on the initial Work with Configuration Status display. Press F24 (More keys) to display additional function keys. F10 and F11 enable you to scroll left and right to view different segments of the configuration status information, such as the type of description, job information, and associated device information.

You can use F14 (Work with ...) to go to the Work with ... Description displays, as in the following list:

- If you are working with network server status, the display shows F14 (Work with network server descriptions).
- If you are working with network interface status, the display shows F14 (Work with network interface descriptions).
- If you are working with line status, the display shows F14 (Work with lines).
- If you are working with controller status, the display shows F14 (Work with controllers).
- If you are working with device status, the display shows F14 (Work with devices).

Status Description of Network Servers, Network Interfaces, Lines, Controllers, and Devices

Each network server, network interface, line, controller, or device description listed on the Work with Configuration Status display has a status associated with it. The following lists provide a status description for each object type.

Status Description of Network Server Objects

The following list provides status descriptions for network server objects.

VARIED OFF

The system is not using the network server for communications. The network server description tells how the communications interface can be used. There may be other network server descriptions that describe the same communications interface (the network server descriptions all have the same resource name (RSRCNAME)). However, only one of several network server descriptions with the same resource name can have a status other than VARIED OFF; the unused network server descriptions have a VARIED OFF status.

VARY OFF PENDING

The network server is being varied off. During this time, the system ends the server functions and communication through the attached lines. When these operations have completed, the network server changes to VARIED OFF status. The system also ensures that the object changes from VARY OFF PENDING to VARIED OFF when a VARY OFF operation completes in error or fails to complete within a set time.

VARY ON PENDING

The network server is being varied on. During this time the system resets the IOP, initializes server functions, and starts communications.

ACTIVE

The network server is automatically in ACTIVE status when VARY ON PENDING is complete.

FAILED

An error occurred for the network server. You can recover this by varying the object off and on again. Information indicating what error caused the object to go into this status appears in the history log (QHST). Similar information also may be in the QSYSOPR message queue or the configured message queue.

Do not confuse this status with the term `Failed` in messages at the QSYSOPR message queue or the configured message queue. Often, these messages use the term `Failed` or `Failure` when describing communications problems. For example, these problems can result from an unreliable link, incorrect configuration of the local or remote system, or failures in the remote system. A status of FAILED for the object indicates that the only recovery for the error is to vary off and on again. This is in contrast to the use of the term in these messages. You can recover many of the messages using the term `Failed` without varying off the object.

DIAGNOSTIC MODE

Problem analysis procedures use the network server to diagnose problems with the network connection. Other users can not use the network server. When the problem analysis procedures are finished, the object shows a VARIED OFF status, and other operations are allowed on the object. You can enter problem analysis procedures by using F14 (Run problem analysis) when you display a message at the QSYSOPR message queue or the configured message queue. You can also use the Analyze Problem (ANZPRB) or Work with Problem (WRKPRB) commands.

***DAMAGED**

The network server description is damaged. This is a system error condition. Information indicating when this damage occurred appears in the history log (QHST). Further information may be in the Vertical Licensed Internal Code (VLIC) logs. When the object shows this status, it must be deleted and then restored. You can use the VLIC log information when reporting a problem to your IBM service representative.

***LOCKED**

The actual status of the network server cannot be determined because another job has an exclusive lock on the network server description. Make another attempt to display the status of the network server. If the status of *LOCKED continues, use the Work with Object Lock (WRKOBJLCK) command to determine which job has the lock on the object description.

***UNKNOWN**

The status indicator of the network server cannot be determined. This is a system error condition. Use the Dump Object (DMPOBJ) command to dump the contents and attributes of the object description into a spooled printer file. If the status of the object description becomes *UNKNOWN, contact your IBM service representative.

Status Description of Network Interface Objects

The following list provides status descriptions for network interface objects.

VARIED OFF

The system is not using the network interface for communications. The network interface description tells how the communications interface can be used. There may be other network interface descriptions that describe the same communications interface (the network interface descriptions all have the same resource name (RSRCNAME)). However, only one of several network interface descriptions with the same resource name can have a status other than VARIED OFF; the unused network interface descriptions have a VARIED OFF status.

VARY OFF PENDING

The network interface is being varied off. During this time the system ends the connection between the local system and the network. When these operations have completed, the network interface changes to VARIED OFF status. The system also ensures that the object changes from VARY OFF PENDING to VARIED OFF when a VARY OFF operation completes in error or fails to complete within a set time.

VARY ON PENDING

The network interface is being varied on. During this time the system begins the operations necessary to communicate with the network. For ISDN connections, the system establishes the connection to the ISDN network termination (NT) and begins communications to the ISDN on the D-channel. For frame-relay and ATM connections, the system establishes the connection to the network.

VARIED ON

The network interface completed the operations described for the VARY ON PENDING status. The network interface is communicating with the network.

ACTIVE

The network interface is successfully placed in VARIED ON status. In addition, the network interface has one or more attached lines that are in a VARY ON PENDING status or higher.

RCYPND

Error recovery is pending for the network interface. If the object shows this status, a message indicating which error occurred on the object appears on the QSYSOPR message queue or the configured message queue. The operator can reply with a G (Go), R (Retry), or C (to cancel the recovery) to the message. Responding with a G or R places the object in VARY ON PENDING status again, and instructs the system to attempt recovery from the error. See the discussion about VARY ON PENDING in this list for successful recovery. If the operator replies to the message with a C (to cancel the recovery), the object goes to RCYCNL status.

RCYCNL

Error recovery is canceled for the network interface. An error occurred on this object and the operator replied with a C (to cancel error recovery for the object) to a message, or the operator used the End Network Interface Recovery (ENDNWIRCY) command to end object recovery when the next second-level error occurs. Refer to “Chapter 4. Handling Communications Errors” on page 107 for more information about error recovery.

You can use the Resume Network Interface Recovery (RSMNWIRCY) command to place the object in a VARY ON PENDING status and resume recovery of the object.

FAILED

An error occurred for the network interface that varying the object off and on again can only recover. For certain errors, it may be necessary to vary off all of the other objects that are running over this same communications controller. Then, reset the first object that you vary on again. You can do this by specifying RESET(*YES) on the Vary Configuration (VRYCFG) command. See the *Communications Configuration* book for more information about this process. Information indicating what error caused the object to go into this status appears in the history log (QHST). Similar information also may be in the QSYSOPR message queue or the configured message queue.

This status should not be confused with the term `Failed` in messages at the QSYSOPR message queue or the configured message queue. Often, these messages use the term `Failed` or `Failure` when describing communications problems. For example, these problems can result from an unreliable link, incorrect configuration of the local or remote system, or failures in the remote system. A status of FAILED for the object indicates that the only recovery for the error is to vary off and on again. This is in contrast to the use of the term in these messages. You can recover many of the messages that use the term `Failed` without varying off the object.

DIAGNOSTIC MODE
Problem analysis procedures use the network interface to diagnose problems with the network connection. Other users can not use the network interface. When the problem analysis procedures are finished, the object shows a VARIED OFF status, and other operations are allowed on the object. You can enter problem analysis procedures by using F14 (Run problem analysis) when you display a message at the QSYSOPR message queue or the configured message queue. You can also use the Analyze Problem (ANZPRB) or Work with Problem (WRKPRB) commands.

***DAMAGED**
The network interface description is damaged. This is a system error condition. Information indicating when this damage occurred appears in the history log (QHST). Further information may be in the Vertical Licensed Internal Code (VLIC) logs. When the object shows this status, it must be deleted and created once more before it can be used again. You can use the VLIC log information when reporting a problem to your IBM service representative.

***LOCKED**
The actual status of the network interface cannot be determined because another job has an exclusive lock on the network interface description. Make another attempt to display the status of the network interface. If the status of *LOCKED continues, use the Work with Object Lock (WRKOBJLCK) command to determine which job has the lock on the object description.

***UNKNOWN**
The status indicator of the network interface cannot be determined. This is a system error condition. Use the Dump Object (DMPOBJ) command to dump the contents and attributes of the object description into a spooled printer file. If the status of the object description becomes *UNKNOWN, contact your IBM service representative.

Status Description of Line Objects

The following list provides status descriptions for line objects.

VARIED OFF

The system is not using the line description for communications. The line description describes the physical line and line protocol that are used for communications. If several line descriptions are configured for the same physical resource, only one can be varied on at a time; all the rest must have a VARIED OFF status. A line description uses ISDN, frame relay, or ATM by referring to network interface descriptions, which in turn refer to physical resources.

Each ISDN network interface description contains two ISDN B-channels, each of which can be switched or nonswitched. Many switched line descriptions can refer to the same network interface description if the following conditions exist:

- The network interface description contains at least one switched B-channel
- All of the line descriptions can be varied on at the same time

However, only one nonswitched line description can be *configured* for each nonswitched B-channel in one network interface description.

VARY OFF PENDING

The line is in the process of being varied off. During this time the system ends the operations that manage communications over the line. When these operations have completed, the line changes to VARIED OFF status. In addition, the system may communicate with data circuit-terminating equipment (DCE). The system also ensures that the object changes from VARY OFF PENDING to VARIED OFF when a VARY OFF operation completes in error or fails to complete within a set time.

VARY ON PENDING

The line is being varied on. During this time the system begins the operations that manage communications over the line. In addition, the system may communicate with the data circuit-terminating equipment (DCE). The operations per communications type vary, depending on the line:

- Nonswitched SDLC and BSC lines: The system raises the data terminal ready signal, and expects the modem to raise the data set ready signal.
- X.25 line: For nonswitched lines, the system attempts to communicate with the X.25 network (by going into asynchronous balanced mode at the HDLC LAPB level). For switched lines, the system prepares the line to go to the connect pending state.
- For switched lines configured to use an ISDN interface (IDLC, PPP, or X.25 over ISDN (interface *X.31)), the VARY ON PENDING status indicates that none of the network interface descriptions and the associated channels, if any, configured in the Switched NWI LIST (SWTNWILST) parameter of the line description are available for switched communications. For nonswitched lines, the system enables the associated ISDN channel in the network interface descriptions for communications.
- Token-ring network line: The system places itself in the local area network (LAN) by participating in exchanging tokens on the network.
- Lines configured to use a frame-relay network interface: The system is establishing a connection over a data link connection identifier (DLCI) or the frame-relay network.
- Lines attached to an ATM network: The system is establishing connections to appropriate servers in the network.

- PPP line: For nonswitched lines, the system raises the data terminal ready signal, and expects the modem to raise the data set ready signal. For switched lines, the system establishes a path to the modem, but does not raise the data set ready signal.

  For both nonswitched and switched PPP lines, when the above operations complete successfully, the line changes to VARIED ON status.

  A PPP analog switched line does not change to CONNECT PENDING status. It is ready and it is waiting for a switched connection to be established while in VARIED ON status.

When these operations have successfully finished, only the following lines change to VARIED ON status:

- Nonswitched BSC, asynchronous, SDLC, X.25, and IDLC
- DDI network
- Ethernet network
- Frame-relay network
- SDLC X.21 short-hold mode
- Token-ring network
- Wireless network
- FAX

A message is placed in QSYSOPR or the configured message queue, indicating that the line was varied on successfully. This information is also placed in the history log (QHST).

The same message is generated for switched BSC, asynchronous, SDLC, IDLC, and X.25 lines that change to VARIED ON status. When the system has determined that the line is ready for a switched connection, the switched lines change to a status of CONNECT PENDING to indicate they are waiting for a switched connection to be established.

If there are errors during these operations, a message describing the error is placed on the QSYSOPR message queue or the configured message queue. Information about the error is placed in the history log (QHST). The status of the line becomes RCYPND if the system requires the operator to reply to a message before attempting to recover from the error. If the system tries the operation again, without having the operator reply to a message by using the system value, communications recovery limit (QCMNRCYLMT), the line continues to be in VARY ON PENDING status. See Chapter 4. Handling Communications Errors for more information about error recovery and how to use QCMNRCYLMT.

VARIED ON

The tasks that communicate over the communications line have been put in place by the system. In addition, the system has the capability to communicate with the data-circuit terminating equipment (DCE) for nonswitched lines because the activity described for the VARY ON PENDING status has successfully finished. Switched lines change to a CONNECT PENDING status, waiting for the switched connection to be established.

CONNECT PENDING

This status is shown only for switched SDLC, BSC, X.25, IDLC, or asynchronous lines. The line is in this status while waiting for the switched connection to be established; this can be either a dial or an answer connection.

ACTIVE
The line is successfully placed in VARIED ON status. In addition, the line has one or more attached controllers that are in a VARY ON PENDING status or higher.

RCYPND
Error recovery is pending for the line. If the line shows this status, a message indicating what error occurred on the line appears on the QSYSOPR message queue or the configured message queue. The operator can reply with a G (Go), R (Retry), or C (to cancel the recovery) to the message. Responding with a G or R places the line in VARY ON PENDING status again, and instructs the system to attempt recovery from the error (see the discussion about VARY ON PENDING in this list for successful recovery). If the operator replies with a C (to cancel the recovery) to the message, the line goes to the RCYCNL status.

RCYCNL
Error recovery is canceled for the line. An error occurred on this line, and the operator replied with a C (to cancel error recovery for the line) to a message, or the operator used the End Line Recovery (ENDLINRCY) command to end line recovery when the next second-level error occurs. See Chapter 4. Handling Communications Errors for more information about error recovery. Information about the error appears in the history log (QHST).

You can use the Resume Line Recovery (RSMLINRCY) command to place the line in a VARY ON PENDING status and resume recovery of the line.

The operator should cancel error recovery if the error condition cannot be corrected to avoid an inefficient use of the system's resources.

FAILED
An error occurred for the line that varying the line off and on again can only recover. For certain errors, it may be necessary to vary off all of the other lines that are running over this same communications controller. Then reset the first line that you vary on again. Specifying RESET(*YES) on the Vary Configuration (VRYCFG) command can do this. See the *Communications Configuration* book for more information about this process. Information indicating what error caused the line to go into this status appears in the history log (QHST). Similar information may also be at the QSYSOPR message queue, or the configured message queue.

This status should not be confused with the term `Failed` in messages at the QSYSOPR message queue or the configured message queue. Often, these messages use the term `Failed` or `Failure` when describing communications problems. For example, these problems can result from an unreliable link, incorrect configuration of the local or remote system, or failures in the remote system. A status of FAILED for the line, in contrast to the use of the term in these messages, indicates that the only recovery for the error is to vary off and on again. Many of the messages using the term `Failed` can be recovered without varying off the line.

If the line description's status becomes FAILED, contact your IBM service representative.

DIAGNOSTIC MODE

The line is being used by problem analysis procedures to diagnose problems with the line, and cannot be used by other users. When the problem analysis procedures are finished, the line shows a VARIED OFF status, and other operations are allowed on the line. You can enter problem analysis procedures by using F14 (Run problem analysis) when you display a message at the QSYSOPR message queue or the configured message queue. You can also use the Analyze Problem (ANZPRB) command.

***DAMAGED**

The line description is damaged. This is a system error condition. Information indicating when this damage occurred appears in the history log (QHST). Further information may be in the Vertical Licensed Internal Code (VLIC) logs. When the line shows this status, it must be deleted and created once more before it can be used again. The VLIC log information can be used when reporting a problem to your IBM service representative.

***LOCKED**

The actual status of the line cannot be determined because another job has an exclusive lock on the line description. Make another attempt to display the status of the line. If the status of *LOCKED continues, use the Work with Object Lock (WRKOBJLCK) command to determine which job has the lock on the line description.

***UNKNOWN**

The status indicator of the line cannot be determined. This is a system error condition. Use the Dump Object (DMPOBJ) command to dump the contents and attributes of the line description to a spooled printer file. If the line description's status becomes *UNKNOWN, contact your IBM service representative.

Status Description of Controller Objects

The following list provides status descriptions for controller objects.

VARIED OFF

The system is not using the controller description for communications.

VARY OFF PENDING

The controller is in the process of being varied off. During this time the system ends the connection between the controller and the line, or ends the operations that control the resources associated with the controller. When these operations have completed, the controller status is changed to VARIED OFF. The system also ensures the status changes from VARY OFF PENDING to VARIED OFF when an operation completes in error or fails to complete within a set time.

VARY ON PENDING

The controller is in the process of being varied on. Controllers on nonswitched lines have this status after you use the Vary Configuration (VRYCFG) command with STATUS(*ON), but the lines to which they are attached have not yet reached the VARIED ON or ACTIVE status. The further significance of this status differs, depending on the protocol used and if the controller is switched or nonswitched:

- Systems Network Architecture (SNA) nonswitched SDLC, IDLC, or X.25 controller: The controller has this status after you use the Vary Configuration (VRYCFG) command, and the controller is exchanging identifiers (XIDs). After successful identification exchange, the controller changes to normal response mode at the data link level (or asynchronous balanced mode at the logical link level, if the controller is an X.25 permanent virtual circuit).

The controller may stay in this status for a long time for certain configurations. For example, if the local system is secondary, the system shows this status and waits for the remote system to begin sending exchange identifiers (XIDs). If the local system is primary or negotiable, and you configure the controller with the connect poll retry parameter without a maximum (CNNPOLLRTY(*NOMAX)), this exchange identifier/set normal response mode (XID/SNRM) polling can continue for an indefinite period of time.

- APPC controller that specifies MDLCTL(*YES): The controller displays this status if the line associated with the model controller is not currently available for use. Some examples of this are when the line is not varied on or active, when no connections are available for use, or the line is specified AUTOCRTCTL(*NO).

  APPC controllers with a link type of *ILAN stay in vary on pending status until the partner Advanced System/36 configuration for ILAN is enabled.
- Some types of 3274 work station controllers, or if the controller describes a host system: The activate physical unit (ACTPU) request is sent by the system when the controller shows this status. For host controller descriptions, an ACTPU request is received from the remote host system; for some 3x74 descriptions, the AS/400 system sends an ACTPU to the remote controller.
- Switched SNA controllers: The controller shows this status while waiting for a switched connection to be established. The switched connection can be physical (such as an SDLC) or logical (such as a local area network or an X.25 switched virtual circuit), after which the controller goes through the steps needed for controllers on nonswitched lines.
- Asynchronous controller on a nonswitched line: The system raises the data terminal ready signal and expects the modem to raise the data set ready signal. An asynchronous controller on a switched line shows this status while waiting for the switched connection to be established.
- Base binary synchronous communication (BSC) (APPTYPE(*PGM)) on a nonswitched line: No line activity occurs while the controller's VARY ON PENDING status is shown. The controller shows this status for base BSC on a switched line while waiting for a switched connection to be established. The local and remote IDs are then exchanged.
- 3270 emulation for BSC (APPTYPE(*EML)) on a nonswitched line: This status is shown if the system begins to monitor the line for polls and selects for its station address, which was specified in the STNADR parameter of the CRTLINBSC command. The system responds with the reverse-interrupt character (RVI) to a select sequence, INTERVENTION REQUIRED status to specific polls, and end-of-transmission character (EOT) to general polls. This activity continues after the controller's status becomes VARIED ON until the device description becomes ACTIVE.
- Multileaving remote job entry (MRJE BSC) (APPTYPE(*RJE)) on a switched line: This status is shown while waiting for a switched connection to be established. After the switched connection is established, the activity is the same as for a nonswitched connection; the AS/400 system sends the host sign-on.

If these operations finish successfully, the controller goes to VARIED ON status. A message indicating that the controller was contacted is also placed on the QSYSOPR message queue or the configured message queue. This same information is also placed in the history log (QHST).

If errors occur during these operations, a message describing the error is placed on the QSYSOPR message queue or the configured message queue, and information about the error is placed in the history log (QHST). If the system requires the operator to reply to a message before attempting to recover from the error, the status of the controller becomes RCYPND. If the system tries the operation again, without having the operator reply to a message (by using the system value QCMNRCYLMT), the status of the controller continues to be VARY ON PENDING. See Chapter 4. Handling Communications Errors for more information about error recovery and how to use QCMNRCYLMT.

VARIED ON

The controller completed the operations that are described for the VARY ON PENDING status. The controller is now ready to perform activity on behalf of the devices that are attached to it. If it is an Advanced Peer-to-Peer Networking (APPN) controller, it may perform intermediate routing. Depending on the values for the network attributes ALWVRTAPPN and ALWHPRTWR, the APPN controller may also perform the following:

- APPN/HPR boundary function
- HPR and APPN endpoint communications, in conjunction with one or more APPN virtual controllers

Note: The APPC controller will remain in VARIED ON status, and will not go to ACTIVE status, in the following instances:

- When the APPC controller description specifies APPN(*YES), and the ALWVRTAPPN network attribute is *YES
- When the ALWHPRTWR attribute is *YES, and the APPC controller description specifies APPN/HPR(*YES)

Any APPC device that was attached to the APPC controller prior to specifying these settings cannot be varied on while these settings are in affect. See the *AS/400e Information Center* for more information.

If this is an APPC controller specified as MDLCTL(*YES), a VARIED ON status indicates the following:

- The line associated with this controller is varied on or active
- It has available connections
- The line supports automatic creation of APPC controllers

Note: The model controllers will not be displayed under a line because the model is not actually using an available connection on the line.

ACTIVE

The controller was successfully placed in VARIED ON status. In addition, the controller has one or more attached devices with a VARY ON PENDING status or higher.

RCYPND

Error recovery is pending for the controller. If the controller shows this status, a message indicating what type of error occurred on the controller appears on the QSYSOPR message queue or the configured message queue. If the controller is on a nonswitched line and an error has occurred on the line, the controller stays in this status while the system waits for the system operator to reply to the message for the line error.

The operator can reply with a G (Go), R (Retry), or C (to cancel the recovery) to the message. Responding with a G or R places the controller in VARY ON PENDING status again, and instructs the system to attempt recovery from the error (see the description of VARY ON PENDING in this list for successful recovery). If the operator replies with a C (to cancel the recovery) to the message, the controller goes to the RCYCNL status.

Note: If an error message has not been sent to the QSYSOPR message queue or the configured message queue, for an APPC controller using a TDLC line, a status of RCYPND is equivalent to the VARY ON PENDING state for APPC controllers attached to other line types. In this case, no error recovery is in progress, and when the remote system (a personal computer) is contacted, the controller description goes to a VARIED ON status.

RCYCNL

Error recovery is canceled for the controller. An error occurred on this controller or associated line, and the operator replied with a C (to cancel error recovery for the controller) to a message, or the operator used the End Controller Recovery (ENDCTLRCY) command to end the controller recovery when the next second-level error occurs. Information about the error appears in the history log (QHST).

You can use the Resume Controller Recovery (RSMCTLRCY) command to place the controller in a VARY ON PENDING status and resume recovery of the controller.

The operator should cancel error recovery if the error condition cannot be corrected to avoid an inefficient use of the system's resources.

FAILED

An error occurred for the controller that varying the controller off and on again can only recover. Information indicating what error caused the controller to go into this status appears in the history log (QHST). Similar information may also be on the QSYSOPR message queue or the configured message queue.

Do not confuse this status with the term `Failed` in messages on the QSYSOPR message queue or the configured message queue. Often, these messages use the term `Failed` or `Failure` when describing communications problems. For example, these problems can be the result of an unreliable data link, incorrect configuration of the local or remote system, or failures in the remote system. A status of FAILED for the controller, indicates that the only recovery for the error is to vary off and on again. This is in contrast to the use of the term in these messages. You can recover many of the messages that use the term `Failed` without varying off the controller.

If the controller description's status becomes *FAILED and the failure is determined not to be the result of incorrect configuration, contact your IBM representative.

DIAGNOSTIC MODE

Problem analysis procedures use the controller to diagnose problems with the controller. Other users can not use the controller. When the problem analysis procedures are finished, the controller shows a VARIED OFF status, and other operations are allowed on the controller. You can enter problem analysis procedures by using F14 (Run problem analysis) when displaying a message at the QSYSOPR message queue or the configured message queue. You can also use the Analyze Problem (ANZPRB) command.

***DAMAGED**
The controller object is damaged. This is a system error condition. Information indicating when the error occurred appears in the history log (QHST). Further information may be in the Vertical Licensed Internal Code (VLIC) logs. When the controller shows this status, it must be deleted and created once more before it can be used again. The VLIC log information can be used when reporting a problem to IBM service.

***LOCKED**
The actual status of the controller cannot be determined because another job has an exclusive lock on the controller. Make another attempt to display the status of the controller. If the *LOCKED status continues, use the Work with Object Lock (WRKOBJLCK) command to determine which job has the lock on the controller object.

***UNKNOWN**
The status indicator of the controller cannot be determined. This is a system error condition. Use the Dump Object (DMPOBJ) command to dump the contents and attributes of the controller description to a spooled printer file. If the controller description's status becomes *UNKNOWN, contact your IBM representative.

Status Description of Device Objects

The following list provides status descriptions for device objects.

VARIED OFF
The system is not using the device description for communications.

VARY OFF PENDING
The device is in the process of being varied off. During this time the system ends the operations that control the resources associated with this device. When the operations complete, the status is changed to VARIED OFF. The system also ensures the status changes from VARY OFF PENDING to VARIED OFF when the operation completes in error or fails to complete within a set time.

VARY ON PENDING
The device is in the process of being varied on. All device types have this status after you use the Vary Configuration (VRYCFG) command with STATUS(*ON) for the device, but the controller to which it is attached has not yet reached the VARIED ON or ACTIVE status. Further significance of this status varies depending on the type of device. See the "VARY ON PENDING Device Status Information" on page 55 for more information describing VARY ON PENDING for different types of devices.

VARIED ON
This status description varies depending on the type of device. See "VARIED ON Device Status Information" on page 56 for more information describing VARIED ON for different types of devices.

ACTIVE/DEGRADED
This system has an operational connection to the system named in the System column through the adapter resource. This connection is in use by user or agent jobs. The degraded status means that redundancy is reduced because this resource has one or more unusable local or remote links. To

determine the reason for the resource degraded status, use the Display OptiConnect Link Status display (by using the DSPOPCLNK command) to check the status of the links.

VARYON/DEGRADED
This system has an operational connection to the system named in the System column through the adapter resource. This connection is ready, but not in use. The degraded status means that redundancy is reduced because this resource has one or more unusable local or remote links. To determine the reason for the resource degraded status, use the Display OptiConnect Link Status display (by using the DSPOPCLNK command) to check the status of the links.

SIGNON DISPLAY
The subsystem is doing sign-on display processing for the device. When the device has this status, a sign-on display is not necessarily shown at the display station. Instead, the system is preparing the device to receive the sign-on display or sending the sign-on display to the display station, or the actual sign-on display is at the display station.

ACTIVE
This status description varies depending on the type of device. See “ACTIVE Device Status Information” on page 59 for more information describing ACTIVE for different types of devices.

READY
This link is ready for OptiConnect activity if needed.

DOWN
This link is not ready due to a loss of signal or a hardware error. Check the cable or connection to determine the cause of this condition.

Note: Down is a normal link status if the bus owner system is powered off.

SYSREQ
The System Request (Sys Req) key on a 5250 display station was pressed and the current session stopped. If the user is on a 3270 display station or the distributed host command facility (DHCF), the 3270 key mapped to the 5250 Sys Req key was pressed.

HELD The user or the system held the communications device to prevent it from communicating. The Release Communications Device (RLSCMNDEV) command can be used to release the device and to allow communications to continue.

RCYPND
Error recovery is pending for the device. If the device shows this status, a message indicating what type of error occurred on the device appears on the QSYSOPR message queue or the configured message queue. If this is a device on a nonswitched line and an error has occurred on the line, or an error occurred on the controller to which the device is attached, the device stays in this status while the system waits for the system operator to reply to the message for the line or controller error.

The operator can reply with a G (Go), R (Retry), or C (to cancel the recovery) to the message. If the operator replies with a G or R, then this places the device in VARY ON PENDING status again and instructs the system to attempt recovery from the error. If the operator replies with a C to the message, the device goes to the RCYCNL status.

RCYCNL

Error recovery was canceled for the device. An error occurred on this device or associated controller or line, and the operator replied with a C (to cancel error recovery for the device) to a message, or the operator used the End Device Recovery (ENDDEVRCY) command to stop the device recovery. Information about the error appears in the history log (QHST).

You can use the Resume Device Recovery (RSMDEVRCY) command to place the device in a VARY ON PENDING status and resume recovery of the device.

If the error condition cannot be corrected, the operator should cancel error recovery to avoid an inefficient use of the system's resources.

FAILED

An error occurred for the device that varying the device off and on again can only recover. Information indicating what error caused the device to go into this status appears in the history log (QHST). Similar information may also be on the QSYSOPR message queue or the configured message queue.

Do not confuse this status with the term `Failed` in messages at the QSYSOPR message queue or the configured message queue. Often, these messages use the term `Failed` or `Failure` when describing communications problems. For example, these problems can be the result of an unreliable data link, incorrect configuration of the local or remote system, or failures in the remote system. A status of FAILED for the device, in contrast to the use of the term in these messages, indicates that the only recovery for the error is to vary off and on again. Many of the messages using `Failed` can be recovered without varying off the device. If the device description's status becomes FAILED and the failure is determined not to be the result of incorrect configuration, contact your IBM service representative.

DIAGNOSTIC MODE

The device is being used by problem analysis procedures to diagnose problems with the device and cannot be used by other users. When the problem analysis procedures are finished, the device shows a VARIED OFF status, and other operations are allowed on the device. You can enter problem analysis procedures by using F14 (Run problem analysis) when displaying a message at the QSYSOPR message queue or the configured message queue. You can also use the Analyze Problem (ANZPRB) command.

***DAMAGED**

The device object is damaged. This is a system error condition. Information indicating when this occurred appears in the history log (QHST). Further information may be in the vertical licensed internal code (VLIC) logs. When the device shows this status, it must be deleted and created once more before it can be used again. The VLIC log information can be used when reporting a problem to IBM service.

***LOCKED**

The actual status of the device cannot be determined, because another job has an exclusive lock on the device. Make another attempt to display the status of the device. If the *LOCKED status continues, use the Work with Object Lock (WRKOBJLCK) command to determine which job has the lock on the device object.

***UNKNOWN**

The status indicator of the device cannot be determined. This is a system error condition. Use the Dump Object (DMPOBJ) command to dump the contents and attributes of the device description to a spooled printer file. If the device description's status becomes *UNKNOWN, contact your IBM representative or your IBM-approved remarketer.

VARY ON PENDING Device Status Information

The following list provides information on the VARY ON PENDING status for different types of devices.

APPC Devices stay VARY ON PENDING until the controller to which they are attached becomes VARIED ON or ACTIVE. For APPC devices, the system determines if the device needs to receive an ACTLU request. That is, APPC devices attempting to communicate with System/370 host systems, as dependent LUs, expect an ACTLU request. If an ACTLU request is received and the system sends a positive ACTLU response, the device status changes to VARIED ON. This also occurs if the system determines that an ACTLU response is not necessary. If the system sends a negative ACTLU response for this device, information indicating why this happened is placed in the history log. A message may also be placed on the QSYSOPR message queue or the configured message queue.

Asynchronous
Devices stay VARY ON PENDING until the controller to which they are attached becomes VARIED ON or ACTIVE. For asynchronous devices, the status of the device then becomes VARIED ON.

BSC Devices stay VARY ON PENDING until the controller to which they are attached becomes VARIED ON or ACTIVE.

- Nonswitched: A nonswitched BSC device has this status for a very short period of time during internal system processing.
- Switched: If APPTYPE is *RPGT or *BSC38, a switched BSC device stays VARY ON PENDING until a user or IBM program establishes a session either by opening a file or by using an acquire operation, and the connection is made.

  If APPTYPE is *BSCEL, a switched BSC device stays as VARY ON PENDING until (1) a user or IBM program establishes a session (either by opening a file or by using an acquire operation), or (2) an incoming call is received by the AS/400 system and the connection is made.

  – Base (APPTYPE is *BSCEL, *BSC38, or *RPGT) BSC switched. The system then exchanges identifiers with the remote system.

  – MRJE (APPTYPE is *RJE). The system will send the sign-on record to the host system.

DHCF Devices stay VARY ON PENDING until the controller to which they are attached becomes VARIED ON or ACTIVE. For distributed host command facility (DHCF) devices, the status remains VARY ON PENDING even after a positive ACTLU response has been sent to the host system from the AS/400 system. The DHCF device waits for the System/370 attached display station to do an **ACQUIRE operation, after which the host system sends a bind request. If a positive response is sent to the bind request, the status becomes VARIED ON.

Remote work station (except TYPE(3277, 3278, or 3279) with MODEL(*DHCF)), APPTYPE(*NRF, *CTLSSN, *DEVINIT, or *APPINIT)), finance and retail devices, and remote printers
Devices stay VARY ON PENDING until the controller to which they are

attached becomes VARIED ON or ACTIVE, after which the system sends an ACTLU request for the device. If a positive ACTLU response is received and the ACTLU response indicates that the device’s power is on, the device’s status becomes VARIED ON. A positive ACTLU response may be received but may indicate that the device’s power is off. If this is the case, the device’s status remains VARY ON PENDING. If the device is started after an ACTLU response is received, the system expects to receive an LUSTAT from a 5250-type controller, or NOTIFY from a 3270-type controller indicating that the device is available. The device status then changes to VARIED ON.

NRF Devices stay VARY ON PENDING until the controller to which they are attached becomes VARIED ON or ACTIVE. The system expects to receive an activate logical unit (ACTLU) request for the devices from the host. The device status remains at VARY ON PENDING when both of the following apply:

- The ACTLU request is received and the system responds with a positive ACTLU response.
- The device description does not have configured logon text.

The device status becomes VARIED ON when both of the following apply:

- The ACTLU request is received and the system responds with a positive ACTLU response.
- The device description has configured logon text.

SPLS If APPTYPE is *CTLSSN, the devices stay VARY ON PENDING until the controller to which they are attached becomes VARIED ON or ACTIVE. The system expects to receive an activate logical unit (ACTLU) request for the device from the host. When the ACTLU request is received and the system responds with a positive ACTLU response, the device status becomes VARIED ON.

If APPTYPE is *DEVINIT, the devices stay VARY ON PENDING until the controller to which they are attached becomes VARIED ON or ACTIVE. The device status remains VARY ON PENDING until the device is selected for a session with a display. After the session is established, the device status becomes VARIED ON.

If APPTYPE is *APPINIT, the devices stay VARY ON PENDING until the controller to which they are attached becomes VARIED ON or ACTIVE. The device status becomes VARIED ON.

SNA host
Devices stay VARY ON PENDING until the controller to which they are attached becomes VARIED ON or ACTIVE. Then, the system expects to receive an activate logical unit (ACTLU) request for the device from the host system. When the ACTLU request is received and the system replies with a positive ACTLU response, the device’s status becomes VARIED ON. If the system replies with a negative ACTLU response for this device, information is placed in the history log that indicates why this happened. A message may also be placed on the QSYSOPR message queue or the configured message queue.

VARIED ON Device Status Information

The following list provides information on the VARIED ON status for different types of devices.

APPC APPC devices have this status for a very short period of time during internal system processing. The APPC device can bind sessions independently of any user program state; therefore, the device's status becomes ACTIVE.

Asynchronous
The asynchronous device stays VARIED ON for a short time during internal system processing. The system immediately allocates an asynchronous device to prepare to receive program start requests; therefore, the device status becomes ACTIVE.

Base BSC
Base BSC devices (APPTYPE is *BSCEL, *BSC38, or *RPGT): The activity varies depending on whether the connection type (CNN) is multipoint, switched, or nonswitched point-to-point:

- Base BSC multipoint: The system monitors the line for poll and select sequences. The system responds with an end-of-transmission character (EOT) to polls and with a wait-before-transmitting-positive acknowledgment character (WACK) to select sequences. If APPTYPE is *BSC38 or *RPGT, the device remains in this status until a user or IBM program establishes a session by either opening a file or by using an acquire operation.

  If the APPTYPE is *BSCEL, the device remains in this status for a very short period of time during internal system processing.
- Base BSC nonswitched point-to-point: The system monitors the line for incoming enquiry (ENQ) characters and signals the unsolicited data event to inform the user or IBM application program that the remote system wants to send data. If the APPTYPE is *BSC38 or *RPGT, the device remains in this status until the application program establishes a session by either opening a file or by using an acquire operation.

  If APPTYPE is *BSCEL the device remains in this status for a very short period of time during internal system processing.
- Base BSC switched point-to-point: A switched device stays in this status for a very short period of time before becoming ACTIVE if the APPTYPE is *BSC38 or *RPGT, because a file has already been opened or acquired. A switched device stays in this status for a very short period of time before becoming ACTIVE if the APPTYPE is *BSCEL, because either a file was opened or acquired, or an incoming call was received by the AS/400 system.

BSC 3270 emulation
BSC 3270 emulation devices (APPTYPE is *EML): The system monitors the line for the location address specified in the device descriptions. If a poll is received, the system responds with an INTERVENTION REQUIRED status. If a select sequence is received, the system responds with a reverse-interrupt (RVI) control character. The device remains in this status until a user or IBM program establishes a session, either by opening a file or by using an acquire operation.

DHCF The distributed host command facility (DHCF) device is in VARIED ON status if it has sent a positive response to a bind request from the host system. This is the status of the device until a subsystem sends a sign-on display to the 3270 display device. Then, the status of the device is SIGNON DISPLAY.

NRF Devices with configured logon text remain VARIED ON until a 3270-type device in the network requests a session with the NRF primary logical unit (PLU) in the Network Control Program (NCP). After the NRF PLU establishes a session with the device and the NRF session partner PLU establishes a session with the AS/400 system, the NRF PLU sends the start data traffic (SDT) SNA command to the AS/400 system. When a subsystem sends a sign-on display to the 3270 display station, the status of the device becomes SIGNON DISPLAY.

Devices without configured logon text remain VARY ON PENDING until a 3270-type device in the network requests a session with the NRF PLU in the NCP. After the NRF PLU establishes a session with the device and the NRF session partner PLU establishes a session with the AS/400 system, the NRF PLU sends the start data traffic (SDT) SNA command to the AS/400 system. The device status becomes VARIED ON. When a subsystem sends a sign-on display to the 3270 display station, the status of the device becomes SIGNON DISPLAY.

SPLS If APPTYPE is *CTLSSN, the status remains VARIED ON while there is an active system services control point-logical unit (SSCP-LU) session for the device. If a deactivate logical unit (DACTLU) request is received for the device, the device status becomes VARY ON PENDING.

If APPTYPE is *DEVINIT, the status remains VARY ON PENDING until the SNA session is established with the device and the start data traffic (SDT) SNA command has been sent by the AS/400 system and responded to by the device. The device status becomes VARIED ON. When a subsystem sends a sign-on display to the 3270 display station, the status becomes SIGNON DISPLAY.

If APPTYPE is *APPINIT, the status remains VARIED on until a subsystem sends a sign-on display to the 3270 display station. The device status becomes SIGNON DISPLAY.

Intrasystem
Intrasystem devices have this status after they have successfully varied on but before any application program has established a session through either an open or acquire operation.

MRJE No associated line activity occurs during this status except holding up the line with null records.

- Nonswitched: The device remains in this status until a user or IBM program establishes a session by opening or using an acquire operation.
- Switched: A switched device stays in this status for a very short period of time before becoming ACTIVE because a file has already been opened or the device acquired.

Network
Network devices have this status after they have successfully varied on but before a user job or IBM job (for example, TCP/IP, or IPX) is started and attaches the device.

Remote work station (except devices with MODEL(*DHCF), APPTYPE(*NRF, *CTLSSN, *DEVINIT, or *APPINIT)), finance and retail devices, and remote printers
The device has this status after the system receives a positive response to an ACTLU request and an indication that the device’s power is on, or until a display file is opened for it. For work stations, this usually means a subsystem attempts to send a sign-on display to the display station. Bind commands must be successfully exchanged for a session before the sign-on can be placed on the display. If the bind request is successful, the

status of the device becomes ACTIVE. For display devices, however, a sign-on display should appear at the device, and the status of the device is SIGNON DISPLAY.

SNA host
The device has this status if it sends a positive response to an activate logical unit (ACTLU) request, and waits for a user or an IBM program (for example, 3270 device emulation) to open an ICF file. When a file opens, the system sends a NOTIFY response (power is on) to the host system, and changes to an ACTIVE status, after which the host system should send a bind request for the device.

ACTIVE Device Status Information

The following list provides information on the ACTIVE status for different types of devices.

APPC The APPC device has this status if it is prepared to handle APPC sessions. The Display Mode Status (DSPMODSTS) command must be used to display the status of any sessions.

Asynchronous
The asynchronous device stays ACTIVE while waiting for program start requests from a remote system after the vary on process is finished, or if a user job establishes a session for the device. The job associated with the device is shown next to the status.

BSC If APPTYPE is not *BSCEL, the BSC device status becomes ACTIVE if a user or IBM-supplied program successfully opens a file and successfully establishes a session. The job associated with the device is shown next to the status.

If APPTYPE is *BSCEL, the device becomes active as follows:

- Nonswitched: When vary on of the device is complete
- Switched: When a file was opened or acquired, or an incoming call was received by the AS/400 system

DHCF The DHCF device stays ACTIVE if a user or IBM program opens a display file. The job associated with the device is shown next to the status.

NRF The NRF device stays ACTIVE if a display file is successfully opened for the device. The job associated with the device is shown next to the status. If a failure occurs, a message is sent to the QSYSOPR message queue or the configured message queue. The status of the device becomes RCYPND.

SPLS If APPTYPE is *DEVINIT and if a display file is successfully opened for the device, the device stays ACTIVE. The job associated with the device is shown next to the status. If a failure occurs, a message is sent to the QSYSOPR message queue or the configured message queue. The status of the device becomes RCYPND.

If APPTYPE is *APPINIT and if a display file is successfully opened for the device, the device stays ACTIVE. The job associated with the device is shown next to the status. If a failure occurs, a message is sent to the QSYSOPR message queue or the configured message queue. The status of the device becomes RCYPND.

Finance
The finance device has a status of ACTIVE after an INIT-SELF command has been received by the AS/400 system and the session has been bound.

The device also shows a status of ACTIVE if a source ICF program successfully acquires the device, causing finance communications to bind the session.

Intrasystem
The intrasystem device status becomes ACTIVE if a user or IBM-supplied program successfully opens a file and successfully establishes a session. The job associated with the device is shown next to the status.

Network
The network device becomes active when a user job or an IBM job (for example, TCP/IP or OSI) is started and attaches the device. The job associated with the device is shown next to the status.

Remote work station (except devices with MODEL(*DHCF) or APPTYPE(*NRF, *DEVINIT, or *APPINIT)), and remote printers
The device stays ACTIVE if a display file is successfully opened for the device. The job associated with the device is shown next to the status. If there is a failure while the device has this status, a message is sent to the QSYSOPR message queue or the configured message queue. The status of the device becomes RCYPND.

Retail The retail device has a status of ACTIVE after an INIT-SELF command has been received by the AS/400 system and the program started by the INIT-SELF request is successfully started and acquires the requesting device. The device also shows a status of ACTIVE if a source ICF program acquires the device; however, an EVOKE command is required to bind the session.

SNA host
The device has a status of ACTIVE when a file has been opened and a session has been successfully established by a job. The associated job is shown next to the status. If a failure occurs while the device has this status, a message is sent to the QSYSOPR message queue or the configured message queue. The status of the device becomes RCYPND.

Work with APPN Status

The Work with APPN Status (WRKAPPNSTS) command provides information about APPN sessions and about sessions that use high-performance routing (HPR). View information about APPN sessions by choosing option 1, Work with APPN Locations. To view information about sessions that use RTP connections, choose option 2, Work with RTP connections. You can use the WRKAPPNSTS command to view the information discussed in the following sections.

APPN virtual controllers
To work with APPN virtual controller descriptions, specify *YES for the Allow Virtual APPN (ALWVRTAPPN) network attribute. When working with APPN virtual controller descriptions, the association between the device descriptions (with job information) and the controller descriptions (that represent connections to adjacent systems) is no longer accessible by the Work with Configuration Status (WRKCFGSTS) command. This situation occurs because the device descriptions no longer are attached to the controller descriptions that represent the remote system. Rather, the device descriptions are attached to the virtual APPN controller. The Work with APPN Status command provides the information that is needed to determine what session activity actually is occurring between the local system and an adjacent system. See the *AS/400e Information Center* for detailed information on APPN Virtual Controller support.

Work with APPN locations

The Work with APPN Locations option of the Work with APPN Status (WRKAPPNSTS) command allows the user to view session information for APPN and HPR. On the Work with APPN Locations display, the total number of APPN sessions per remote and local location pair is shown under the associated controller. From this display, choose Work with sessions Option 5 to retrieve further detail about the session. The Work with APPN Locations display also allows the user to examine any RTP connections that are using the same location pair or controller. You can view configuration information by choosing Option 12 from this display.

Work with RTP connections

Using high-performance routing (HPR) with a rapid-transport protocol (RTP) connection endpoint, the Work with RTP Connections option from the WRKAPPNSTS command enables the user to view session, route, and configuration information about the RTP connections that originate or end within the node. The Work with RTP Connections display also has options for the user to fine-tune the connection/session path. This allows the user to attempt to path switch the RTP connection, or to end the RTP connection.

For additional information, see the *AS/400e Information Center*.

Displaying Connection Status

You can display current information about connection-oriented protocols in use by, and all acceptable inbound routing data specified for, network devices by using the Display Connection Status (DSPCNNSTS) command. This command is valid for all network devices, but connection-oriented status is provided only for devices with a link type of X.25. You must have operational authority to the network device you are querying to use this command.

The Display Connection Status information is shown on several displays. All displays show the following:

Device

The name of the network device specified on the DSPCNNSTS command.

Type

The network protocol type of the specified device.

***IPX** Internetwork Packet Exchange

***TCPIP**

Transmission Control Protocol/Internet Protocol

***USRDFN**

User-defined communications

Device status

The status of the device.

ACTIVE

The device is in use.

DIAGNOSTIC MODE

The device has been put in diagnostic mode.

FAILED

The device is in an unusable state.

RCYCNL
Error recovery has been canceled for the device.

RCYPND
Error recovery is pending for the device.

VARIED ON
The device is varied on.

VARY ON PENDING
The device is varied on pending completion of some action.

Job
The name of the job associated with the device.

Job name
A 10-character name.

Blank No job is associated with the specified network device.

User
The name of the user associated with the device.

User name
A 10-character name.

Blank No user is associated with the specified network device.

Number
The job number associated with the specified network device.

Job number
A 6-digit decimal value.

Blank No job number is associated with the specified network device.

Link type
The type of line to which the network device is attached.

***ELAN**
Ethernet line

***TRLAN**
IBM token-ring network line

***DDI** Distributed data interface line

***FR** frame-relay line

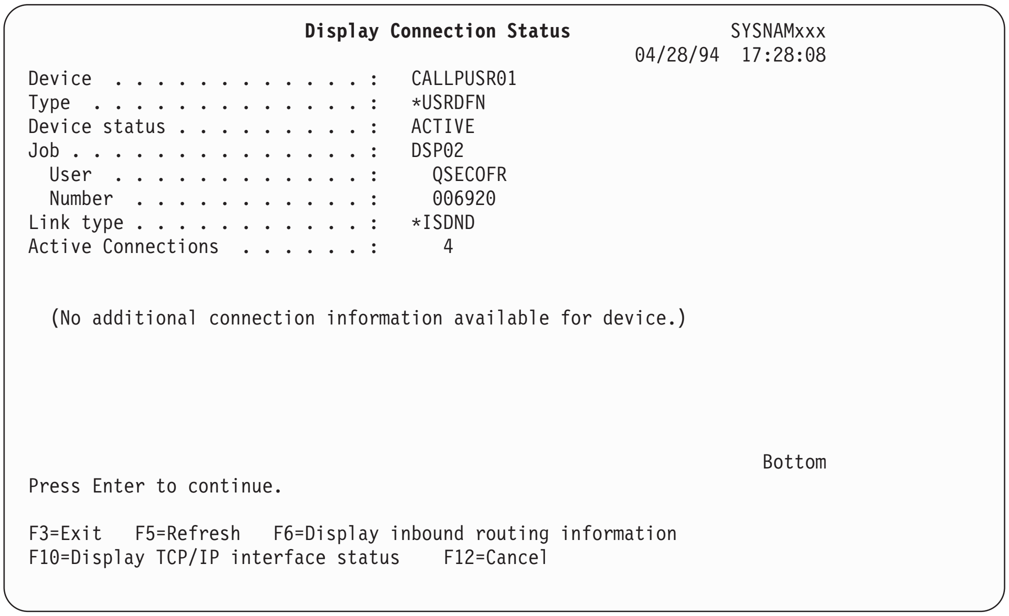
***WLS** Wireless line

***X25** X.25 line

***ISDND**
Integrated Services Digital Network D-Channel

***PPP** Point-to-Point Protocol line

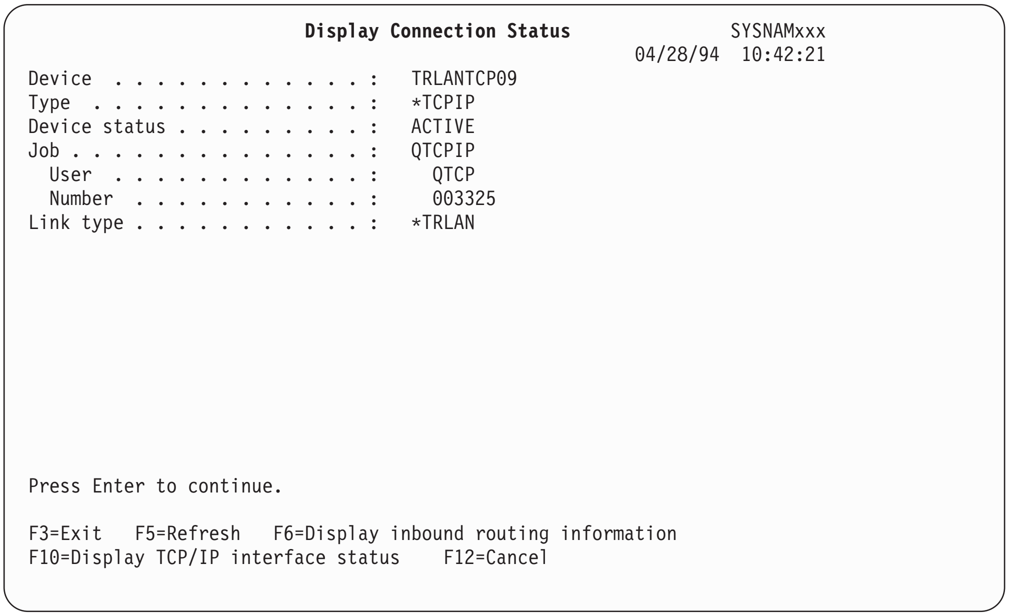
Active connections
If the link type of the network device is X.25 or ISDN, the number of active connections is displayed.



```
                          Display Connection Status             SYSNAMxxx
                                                       04/28/94  17:28:08
Device  . . . . . . . . . . . . :   CALLPUSR01
Type  . . . . . . . . . . . . . :   *USRDFN
Device status . . . . . . . . . :   ACTIVE
Job . . . . . . . . . . . . . . :   DSP02
  User  . . . . . . . . . . . . :     QSECOFR
  Number  . . . . . . . . . . . :     006920
Link type . . . . . . . . . . . :   *ISDND
Active Connections  . . . . . . :      4


  (No additional connection information available for device.)







                                                                   Bottom
Press Enter to continue.

F3=Exit   F5=Refresh   F6=Display inbound routing information
F10=Display TCP/IP interface status    F12=Cancel
```

If the link type of the network device is X.25 and the device has one or more active connections, the connection characteristics are shown for each active connection. A maximum of 64 logical connections is possible.

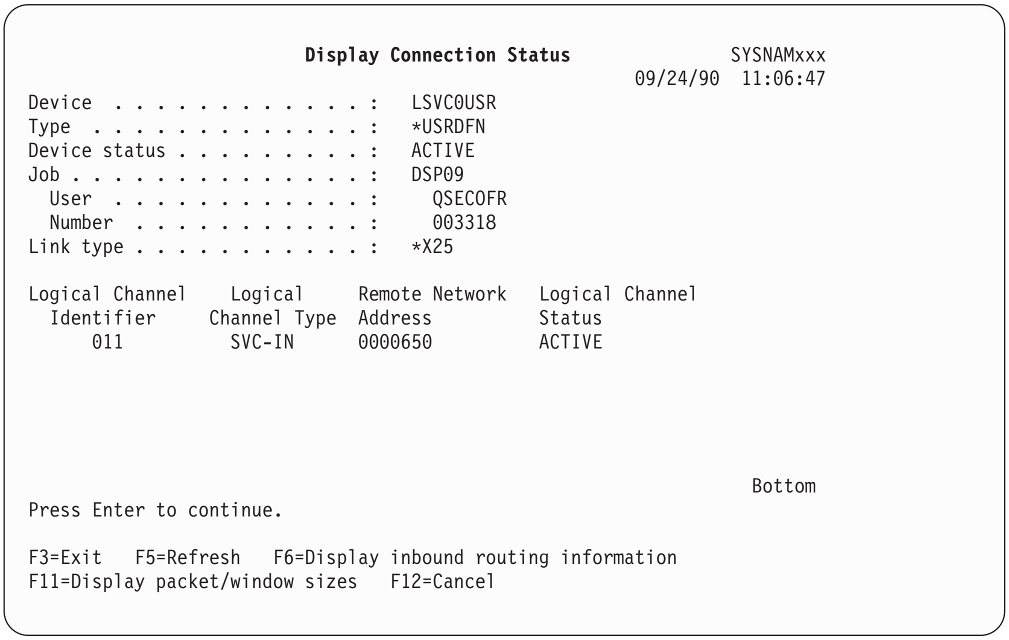
If the network device has a link type of Ethernet or token ring, the following display with no connection characteristics is displayed:



```
                          Display Connection Status             SYSNAMxxx
                                                       04/28/94  10:42:21
Device  . . . . . . . . . . . . :   TRLANTCP09
Type  . . . . . . . . . . . . . :   *TCPIP
Device status . . . . . . . . . :   ACTIVE
Job . . . . . . . . . . . . . . :   QTCPIP
  User  . . . . . . . . . . . . :     QTCP
  Number  . . . . . . . . . . . :     003325
Link type . . . . . . . . . . . :   *TRLAN











Press Enter to continue.

F3=Exit   F5=Refresh   F6=Display inbound routing information
F10=Display TCP/IP interface status    F12=Cancel
```

Each logical channel has a logical channel identifier, logical channel type, remote network address, logical channel status, packet size, window size, protocol identifier, and reverse charging information associated with it. If the DSPCNNSTS command is used for a device without active connections (all valid states other than ACTIVE), the message CPD87B0 `No active connections for device` is displayed.

The following is the first of three displays of connection information:



```
                          Display Connection Status            SYSNAMxxx
                                                       09/24/90  11:06:47
Device  . . . . . . . . . . . . :   LSVC0USR
Type  . . . . . . . . . . . . . :   *USRDFN
Device status . . . . . . . . . :   ACTIVE
Job . . . . . . . . . . . . . . :   DSP09
  User  . . . . . . . . . . . . :     QSECOFR
  Number  . . . . . . . . . . . :     003318
Link type . . . . . . . . . . . :   *X25

Logical Channel    Logical     Remote Network   Logical Channel
  Identifier     Channel Type  Address          Status
      011          SVC-IN      0000650          ACTIVE






                                                                    Bottom
Press Enter to continue.

F3=Exit   F5=Refresh   F6=Display inbound routing information
F11=Display packet/window sizes   F12=Cancel
```

The following information appears on the first display:

Logical channel identifier
The hexadecimal number assigned to a logical channel on an X.25 data link.

’001’-’FFF’
A unique hexadecimal identifier.

***UNKNOWN**
The logical channel identifier is not yet known; a connection is being made.

Logical channel type
This value specifies how this connection is started.

SVC-IN
A switched virtual circuit with an incoming call is used to make the connection.

SVC-OUT
A switched virtual circuit with an outgoing call is used to make the connection.

PVC A permanent virtual circuit (PVC) is used to make the connection.

Remote network address
The network-specific address of the connection.

Remote network address
This value is a 1- through 17-digit decimal number.

Blank The remote network address is not known. This address is never known for a PVC connection.

Logical channel status
The current state of the connection.

ACTIVATE PENDING
The logical channel is being activated.

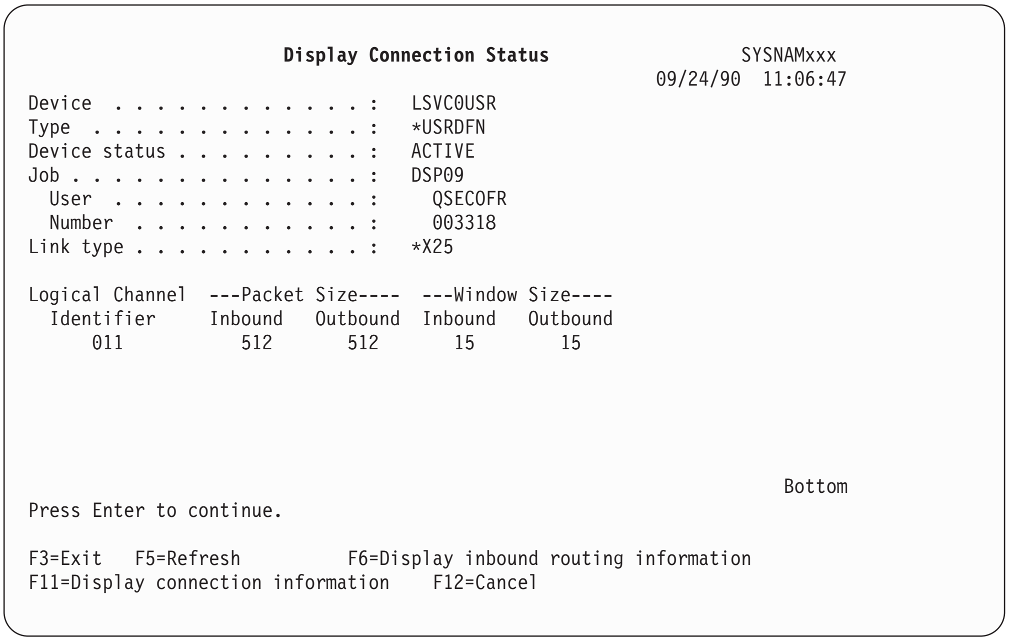
If the logical channel type is SVC-IN, an incoming call packet has been received, and the user of the device description has been notified. The user of the device description did not indicate whether the incoming call packet should be accepted or rejected.

If the logical channel type is SVC-OUT, a call request packet was sent to the X.25 network and the call accept or clear indication packet has not been received.

ACTIVE
The call is successfully made and the logical channel is active.

DEACTIVATE PENDING
The logical channel is in the process of being deactivated. A clear request packet is sent to the X.25 network but the clear confirmation packet has not been received.

Press F11 to see the remaining two Display Connection Status displays. The following is the second display of the Display Connection Status series:



```
                        Display Connection Status                  SYSNAMxxx
                                                           09/24/90  11:06:47
Device  . . . . . . . . . . . . :   LSVC0USR
Type  . . . . . . . . . . . . . :   *USRDFN
Device status . . . . . . . . . :   ACTIVE
Job . . . . . . . . . . . . . . :   DSP09
  User  . . . . . . . . . . . . :     QSECOFR
  Number  . . . . . . . . . . . :     003318
Link type . . . . . . . . . . . :   *X25

Logical Channel  ---Packet Size----  ---Window Size----
  Identifier     Inbound   Outbound  Inbound   Outbound
      011          512       512       15        15





                                                                        Bottom
Press Enter to continue.

F3=Exit   F5=Refresh          F6=Display inbound routing information
F11=Display connection information    F12=Cancel
```

This display contains the following fields:

Packet size
The negotiated packet sizes for the connection inbound indicates the negotiated packet size for receive, while outbound indicates the negotiated packet size for transmit packets.

- *UNKNOWN: The packet size negotiation is not complete.
- 64
- 128
- 256
- 512
- 1024
- 2048
- 4096

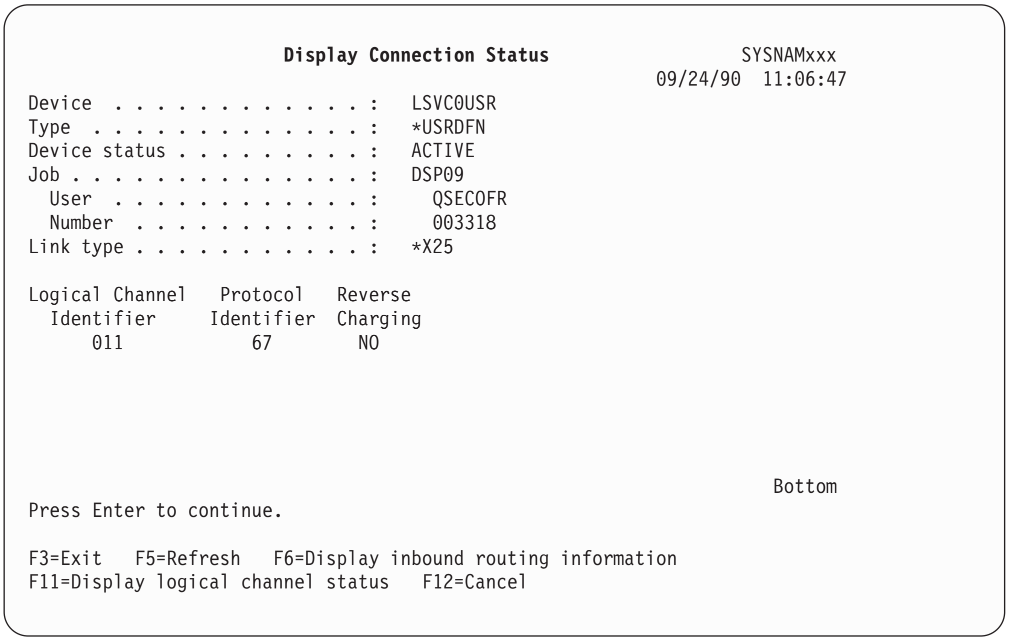
Window size

This value indicates the negotiated window size for the connection. This value is the maximum number of unacknowledged packets that can be outstanding at any time. Inbound indicates the window size coming in from the network. Outbound indicates the window size going out to the network.

***UNKNOWN**
The window size negotiation is not complete.

Window size
An integer value from 1 through 15.

The following is the third display in the Display Connection Status series:



```
                        Display Connection Status                SYSNAMxxx
                                                          09/24/90  11:06:47
Device  . . . . . . . . . . . . :   LSVC0USR
Type  . . . . . . . . . . . . . :   *USRDFN
Device status . . . . . . . . . :   ACTIVE
Job . . . . . . . . . . . . . . :   DSP09
  User  . . . . . . . . . . . . :     QSECOFR
  Number  . . . . . . . . . . . :     003318
Link type . . . . . . . . . . . :   *X25

Logical Channel   Protocol   Reverse
  Identifier     Identifier  Charging
      011           67        NO





                                                                      Bottom
Press Enter to continue.

F3=Exit   F5=Refresh   F6=Display inbound routing information
F11=Display logical channel status   F12=Cancel
```

This display contains the following fields:

Protocol identifier

The first byte of call user data. It is used to identify the higher-level protocol running on this channel.

Hexadecimal value
’00’-’FF’

***NONE**
No protocol identifier exists for this logical channel because no user data was sent with the call request.

Blank The protocol identifier is not applicable to PVC connections.

Reverse charging

A value that indicates whether reverse charging is requested on this channel.

YES For SVC-OUT connections, reverse charging is requested in the call request packet. For SVC-IN connections, reverse charging is requested in the incoming call packet.

NO For SVC-OUT connections, reverse charging is not requested in the call request packet. For SVC-IN connections, reverse charging is not requested in the incoming call packet.

Blank Reverse charging is not applicable to PVC connections.

Equivalent information for SNA protocols can be displayed by using the Display Controller Description (DSPCTLD) command after the controller is active.

Displaying Inbound Routing Information

Pressing F6 (Display inbound routing information) on the Display Connection Status display shows all acceptable inbound routing information for the specified network device. The display you see depends on the link type of the network device (*ELAN, *TRLAN, *DDI, *FR, *WLS, *X25, or *ISDND). All displays show:

Device
The name of the network device specified on the DSPCNNSTS command.

Type
The type of network protocol used for the Create Device Description (Network) (CRTDEVNET) or Change Device Description (Network) (CHGDEVNET) command for the specified network device.

***IPX** Internetwork Packet Exchange

***TCPIP**
Transmission Control Protocol/Internet Protocol

***USRDFN**
User-defined communications

Device status
The status of the device.

ACTIVE
The device is in use.

DIAGNOSTIC MODE
The device is in diagnostic mode.

FAILED
The device is in an unusable state.

RCYCNL
Error recovery is canceled for the device.

RCYPND
Error recovery is pending for the device.

VARIED ON
The device is varied on.

VARIED ON PENDING
The device is varied on pending completion of some action.

Job
The name of the job associated with the device.

Job-name
10-character name.

Blank No job is associated with the specified network device.

User
The name of the user associated with the network device.

User 10-character name.

Blank No user is associated with the specified network device.

Number
The job number associated with the network device.

Number
6-digit decimal value.

Blank No job number is associated with the specified network device.

Link type
The type of line to which the network device is attached.

***ELAN**
Ethernet line

***TRLAN**
IBM token-ring network line

***DDI** Distributed data interface line

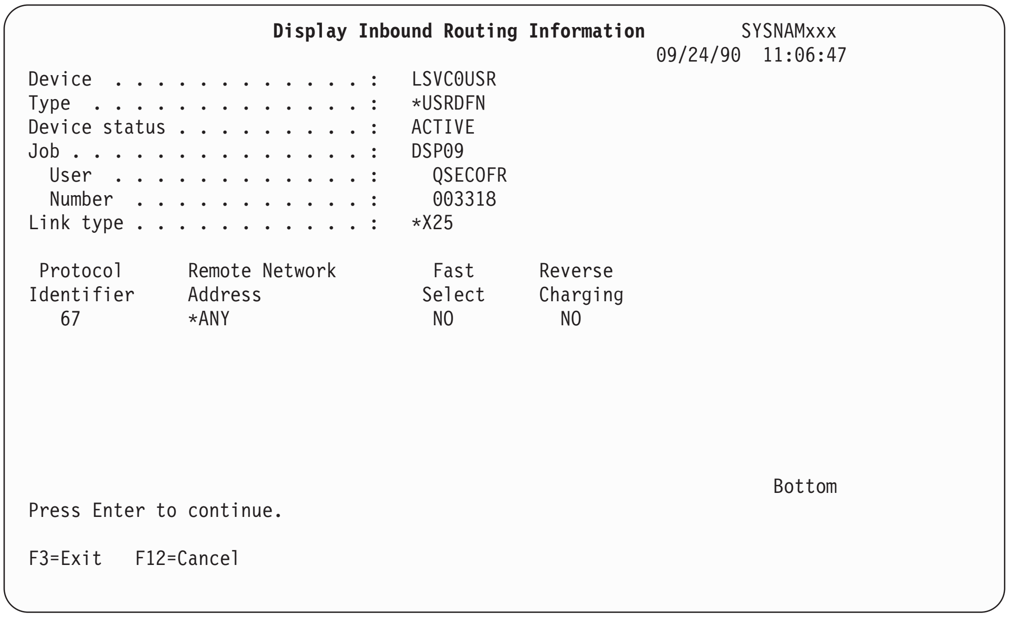
***FR** frame-relay line

***WLS** Wireless line

***X25** X.25 line

***ISDND**
Integrated Services Digital Network D-Channel

If the link type is X.25, the following display is shown:



```
                       Display Inbound Routing Information         SYSNAMxxx
                                                           09/24/90  11:06:47
Device  . . . . . . . . . . . . :   LSVC0USR
Type  . . . . . . . . . . . . . :   *USRDFN
Device status . . . . . . . . . :   ACTIVE
Job . . . . . . . . . . . . . . :   DSP09
  User  . . . . . . . . . . . . :     QSECOFR
  Number  . . . . . . . . . . . :     003318
Link type . . . . . . . . . . . :   *X25

 Protocol      Remote Network         Fast      Reverse
Identifier     Address               Select     Charging
   67          *ANY                   NO          NO








                                                                      Bottom
Press Enter to continue.

F3=Exit   F12=Cancel
```

The protocol identifier, remote network address, fast select, and reverse charging values are used to route incoming calls to the application using the specified device. This information is used only when establishing a session with a remote system.

This display shows:

Protocol identifier
The first byte of call user data on a call request packet.

Hexadecimal value
’00’-’FF’

***NONE**
No protocol identifier exists for this logical channel because no user data was sent with the call request.

Remote network address
The data terminal equipment (DTE) address of the remote system.

Remote network address
This value is a 1- through 17-digit decimal number.

***ANY** Any valid DTE address is accepted.

Fast select
The X.25 facility that allows the user to send 128 bytes of data on a call request.

YES Calls specifying fast select are accepted.

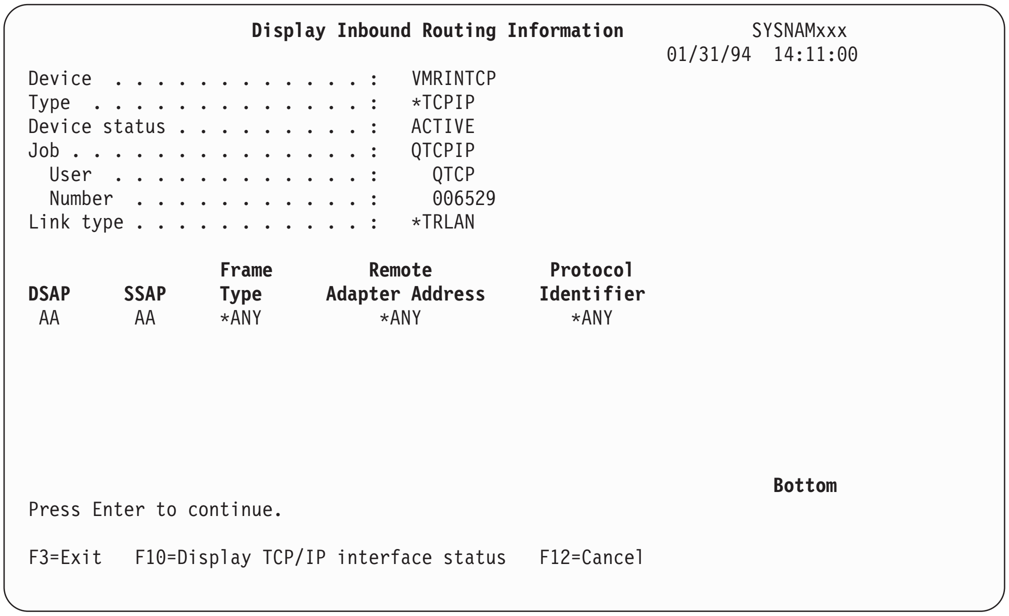
NO Calls specifying fast select are not accepted.

Reverse charging
The X.25 facility that allows reverse charging.

YES Calls specifying reverse charging are accepted.

NO Calls specifying reverse charging are not accepted.

If the link type is LAN (Ethernet or token-ring), the following display is shown:



```
                     Display Inbound Routing Information           SYSNAMxxx
                                                            01/31/94  14:11:00
Device  . . . . . . . . . . . . . :   VMRINTCP
Type  . . . . . . . . . . . . . . :   *TCPIP
Device status . . . . . . . . . . :   ACTIVE
Job . . . . . . . . . . . . . . . :   QTCPIP
  User  . . . . . . . . . . . . . :     QTCP
  Number  . . . . . . . . . . . . :     006529
Link type . . . . . . . . . . . . :   *TRLAN

                  Frame         Remote           Protocol
DSAP     SSAP     Type      Adapter Address     Identifier
 AA       AA      *ANY           *ANY              *ANY







                                                                      Bottom
Press Enter to continue.

F3=Exit   F10=Display TCP/IP interface status   F12=Cancel
```

The DSAP, SSAP, frame type, remote adapter address, and protocol identifier are used to route incoming data to the application using the specified device. This information is used every time incoming data is received.

This display shows:

DSAP
Destination service access point.

Hexadecimal value
A 2-digit hexadecimal number.

Blank There is no destination service access point information.

SSAP
Source service access point.

Hexadecimal value
A 2-digit hexadecimal number.

***ANY** Any source service access point is supported as long as the other three conditions shown on this display are met.

Blank There is no source service access point information.

Frame Type
The protocol of inbound data.

Hexadecimal value
4-digit hexadecimal number.

***ANY** Any frame type is supported as long as the other three conditions shown on this display are met.

Blank There is no frame type.

Remote adapter address
The local adapter address for the remote system. This address describes the system to the network.

Hexadecimal value
12-digit hexadecimal number.

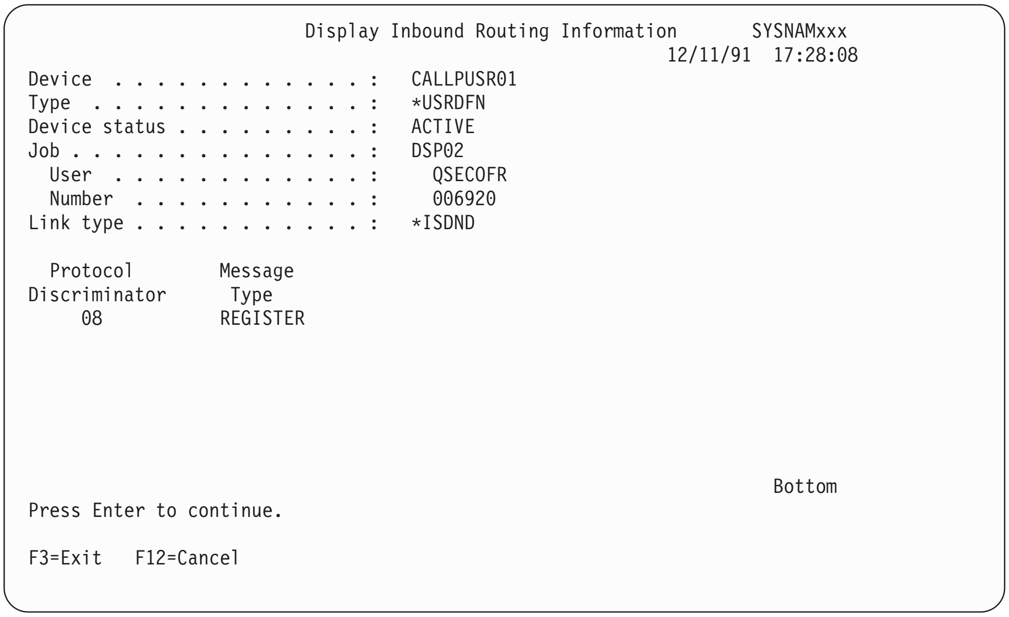
***ANY** Any local adapter address is supported as long as the other three conditions shown on this display are met.

Protocol identifier
Protocol identifier.

Hexadecimal value
6-digit hexadecimal number.

***ANY** Any protocol identifier will be supported.

If the link type is *ISDND, the following display is shown:



```
                          Display Inbound Routing Information       SYSNAMxxx
                                                                  12/11/91  17:28:08
Device  . . . . . . . . . . . . . :   CALLPUSR01
Type  . . . . . . . . . . . . . . :   *USRDFN
Device status . . . . . . . . . . :   ACTIVE
Job . . . . . . . . . . . . . . . :   DSP02
  User  . . . . . . . . . . . . . :     QSECOFR
  Number  . . . . . . . . . . . . :     006920
Link type . . . . . . . . . . . . :   *ISDND

  Protocol        Message
Discriminator      Type
     08          REGISTER







                                                                        Bottom
Press Enter to continue.

F3=Exit   F12=Cancel
```

This display shows:

Protocol discriminator

The first part of every Q.931 or Q.932 message. The purpose of the protocol discriminator is to distinguish messages for user-network call control from other messages. Only the ISDN protocol discriminator, X'08', is supported.

Message type

The third part of every Q.931 or Q.932 message. The message type identifies the function of the message being sent. A message type of *REGISTER* means a third-party connection has been initiated.

Controlling Modes

You can use the following commands to display the mode status, and to start and end modes with remote systems. The *APPC Programming* book contains more information concerning these commands.

Using the Start Mode Command

The Start Mode (STRMOD) command starts a mode. You can establish sessions between the local location and remote location using either the Start mode or the mode you specify on the command as being the one in which the system is to start. Use the STRMOD command to start one or all modes for an APPC or APPN configuration. This command is required only if a user previously ran the End Mode (ENDMOD) command to end the mode. The APPC and APPN support uses an implicit STRMOD command when a device description becomes active, as follows:

- If a device description is automatically created by the APPN support or a device description is manually created with the APPN parameter specified as *YES, the STRMOD command is used when a session establishment request is received.
- If a device description is manually created with the APPN parameter specified as *NO, the STRMOD command is used when the device description is varied on.

Note: If an explicit STRMOD command is used, the remote location must be active; otherwise, the command fails.

To use the STRMOD command, you must specify the following parameters:

Remote location name (RMTLOCNAME)

The remote location name. This is a required parameter.

Device (DEV)

The device description name.

***LOC** The device description is determined by the system. This is the default value.

device-name

The name of the device description.

Mode (MODE)

The mode that starts.

***NETATR**

The mode specified in the network attributes is used. This is the default value.

***ALL** All modes currently in use for the remote location are started.

- For a device description automatically created by the APPN support, or a device description manually created with the APPN parameter specified as *YES, *ALL indicates that any modes that have been used while the remote location was active, but are not currently started, are to start.

- For a device description manually created with the APPN parameter specified as *NO, *ALL specifies that all configured modes for the specified remote location are to start.

BLANK
The mode name, consisting of 8 blank characters, that is used.

mode-name
The name of a mode.

Note: SNASVCMG and CPSVCMG are reserved names and cannot be specified.

Local location name (LCLLOCNAME)
The name of your location.

***LOC** The local location name is determined by the system. This is the default value.

***NETATR**
The default local location name specified in the network attributes is used.

local-location-name
The name of your location. Specify the local location name if you want to indicate a specific local location name for the remote location.

Remote network ID (RMTNETID)
The remote network ID used with the remote location.

***LOC** The system selects the remote network ID. This is the default value.

***NETATR**
The remote network ID specified in the network attributes is used.

***NONE**
The remote network ID is not specified.

remote-network-id
The name of the remote network ID.

Using the Display Mode Status Command

| The Display Mode Status (DSPMODSTS) command shows the status of all of the
| mode descriptions for an advanced program-to-program communications (APPC)
| configuration. The display shows the status of the APPC device description, the
| current number of source, target, and detached conversations in use, and the
| configured and operational session maximum values. This command is only valid
| for APPC device descriptions (including APPC over TCP/IP), and if a mode is being
| used by an APPC device description.

To use the DSPMODSTS command, specify the following parameters:

Device (DEV)
The name of the APPC device description using the mode to be displayed.

Mode (MODE)
The status of the mode being displayed.

***ALL** All of the modes used by the specified device are displayed. This is the default value.

mode-name
The name (8-character maximum) of the mode whose status is displayed for the specified device.

Output (OUTPUT)
The output from the command is shown at the requesting display station or printed with the job's spooled output.

***** The output is shown (if requested by an interactive job) or printed with the job's spooled output (if requested by a batch job). This is the default value.

***PRINT**
The output is printed with the job's spooled output on a printer.

Refer to the *APPC Programming* book for information about the Display Mode Status display and its options.

Using the End Mode Command

| The End Mode (ENDMOD) command ends one or more active modes. You can
| also specify how requested activities at the remote system (those not yet started)
| are to be handled. When an ENDMOD command is run, sessions cannot be started
| between the local and remote locations, or any other mode that has been ended,
| until an explicit Start Mode (STRMOD) command is run.

When the local session maximum is zero and a switched connection is made (either dial or answer), no communications occur on that mode until a STRMOD command allows sessions to be established. However, a local session maximum of zero does not prevent a switched connection from being made. To use the ENDMOD command, specify the following parameters:

Remote location name (RMTLOCNAME)
The remote location name for which one or more modes are ended. This is a required parameter.

Device (DEV)
The device description name.

***LOC** The device description is determined by the system. This is the default value.

device-name
The name of the device description.

Mode (MODE)
The mode that ends.

***NETATR**
The mode specified in the network attributes is used. This is the default value.

***ALL** All modes currently in use by the remote location are ended.

BLANK
The mode name, consisting of 8 blank characters, that is used.

mode-name
The name of a mode.

Note: SNASVCMG and CPSVCMG are reserved names and cannot be specified.

Local location name (LCLLOCNAME)
The name of your location.

***LOC** The local location name is determined by the system. This is the default value.

***NETATR**
The default local location name specified in the network attributes is used.

local-location-name
The name of your location. Specify the local location name if you want to indicate a specific local location name for the remote location.

Remote network ID (RMTNETID)
The remote network ID used with the remote location.

***LOC** The system selects the remote network ID. This is the default value.

***NETATR**
The remote network ID specified in the network attributes is used.

***NONE**
The remote network ID is not specified.

remote-network-id
The name of the remote network ID.

Complete pended requests (CPLPNDRQS)
The remote location can complete pending work or the pending work ends before any other requested activities can start.

***NO** The activities currently in progress at the remote location can finish. Activities that were requested, but not started at the remote location are not performed. This is the default value.

***YES** All requested activities are allowed to complete before the mode is ended.

Changing Maximum Sessions

The Change Session Maximum (CHGSSNMAX) command dynamically changes the maximum number of sessions the local location allows to a mode. If a change to the MAXSSN parameter is made, the remote location is informed and can negotiate for a lower number of maximum sessions. The remote location cannot negotiate a number of maximum sessions higher than the value specified in the maximum session (MAXSSN) parameter. The resulting maximum session parameter is the *current* number of maximum sessions. Neither location can activate more sessions than the current maximum session value. If the requested number of maximum sessions is accepted or negotiated by the remote location, the value requested on the CHGSSNMAX command is stored as the local maximum session parameter. The remote location cannot increase the current maximum session value to a greater number than the value stored as the local number of maximum sessions.

If the request to change the number of maximum sessions is rejected by the remote location, the CHGSSNMAX command ends abnormally and the local maximum session value changes as follows:

- If the number requested for the maximum sessions is greater than the current maximum number, the value changes to the value specified on the MAXSSN parameter.
- If the number requested for the maximum sessions is less than the current maximum number, the local maximum session value does not change.

This new value for the local session is used only if a new maximum session value needs to be negotiated. The current maximum session value controlling the number of sessions that can be active between the local and the remote location does not change if the command fails.

The system operator uses this command to control the number of sessions that can be active with a remote location at the same time when the specified remote location and mode are active. If the current number of active sessions is greater than the maximum number specified on the command, no new sessions are created until the number of active sessions becomes less than the number specified in the command parameter. If the current number of active sessions is less than the maximum number specified, sessions cannot be established until the jobs requiring them begin.

The value determined by the locations remains in effect until another CHGSSNMAX command or an End Mode (ENDMOD) command is used for the same mode, or until all the device descriptions associated with the remote location are varied off.

Many CHGSSNMAX commands can be used before the current maximum number of sessions becomes active. The number specified in the command when it was last used is the current local maximum session parameter.

Notes:

1. If this command decreases the number of sessions with a remote location, the available locally controlled sessions end first, followed by any other available sessions. If the new value for the maximum sessions is still not reached, the other sessions end as the jobs using them end or are canceled.
2. If this command increases the maximum number of sessions that can be used with a remote location, the locally controlled sessions are made available first (depending on the negotiated values), and then the other sessions are made available.
3. This command does not change the value specified for the MAXSSN parameter in the mode description. Use the Change Mode Description (CHGMODD) command to permanently change the value.

Table 4 shows the parameters for the CHGSSNMAX command.

Table 4. CHGSSNMAX Parameters

Parameter Name	Keyword	Description	Valid Values
Remote location name	RMTLOCNAME	Name of the remote location. This is a required parameter.	*remote-location-name*
Device	DEV	Name of the APPC device description.	*LOC (default) or *device-description-name*
Mode[1]	MODE	Name of the mode that changed.	*NETATR, *BLANK, or *mode-name* (8 character maximum)
Maximum session	MAXSSN	Number of sessions allowed with the remote location.	1 through 512

Table 4. CHGSSNMAX Parameters (continued)

Parameter Name	Keyword	Description	Valid Values
Local location name	LCLLOCNAME	The name of your location.	*LOC (default), *NETATR, or *local-location-name*
Remote network ID	RMTNETID	Remote network ID user with the remote location.	*LOC (default), *NETATR, *NONE or *remote-network-id*

Note:

[1] SNASVCMG and CPSVCMG are reserved names and cannot be specified.

Managing Communication Messages

Communications functions use messages to track the status of the connection. Messages informing of the bringup, takedown, and error recovery of communications configuration objects are sent. There are three primary places where communications messages are logged:

QSYSMSG
QSYSMSG is an optionally created message queue in QSYS. This message queue is used for messages of high importantance. See the *CL Programming* book for additional information.

QHST
The history log. More messages are sent to QHST than are sent to QSYSOPR or the configured message queue.

QSYSOPR or the configured message queue
By default, communication messages are sent to the system operator message queue (QSYSOPR). There is a system value, QCFGMSGQ, that can be set so communications messages are sent to a message queue other than QSYSOPR. In addition, several line and controller types have a MSGQ parameter to specify where the messages should be sent for that particular configuration object. Since messages may be sent to either QSYSOPR, or a different message queue specified by the QCFGMSGQ system value or the MSGQ parameter, you will see the wording *QSYSOPR or the configured message queue*. This means to check QSYSOPR or the message queue defined by QCFGMSGQ, or the message queue defined by the MSGQ parameter on the appropriate configuration object.

To help you manage the potentially large number of communications messages, there is the QCFGMSGQ system value. By default this system value is set to QSYS/QSYSOPR, so the communications messages are sent to the system operator message queue. However, more granularity may be necessary to make it easier to manage your messages and your configuration objects.

Supplied with the system is a new message queue, QCFGMSGQ, in the QSYS library. This message queue is created with the same characteristics as the QSYSOPR message queue. Simply changing the QCFGMSGQ system value to QSYS/QCFGMSGQ can remove the communications messages from QSYSOPR. This makes other operating system messages in QSYSOPR more visible. In addition, it isolates the communications messages to their own message queue.

In addition to the QCFGMSGQ system value, several configuration commands have been enhanced with a MSGQ parameter. The following line descriptions have had this parameter added:

- Distributed Data Interface Line
- Ethernet Line
- Frame-Relay Line
- Token-ring Line
- X.25 Line
- PPP Line

The following controller descriptions have had the MSGQ parameter added:

- APPC Controller
- Async Controller
- Local Work Station Controller
- Remote Work Station Controller
- SNA Host Controller
- Virtual Work Station Controller

The default value for the MSGQ parameter on the above configuration objects is *SYSVAL. This means to use the message queue as defined by the QCFGMSGQ system value.

APPC and printer devices already have a MSGQ parameter. The default value of this parameter was QSYS/QSYSOPR prior to V4R4. Beginning in V4R4, the default for the MSGQ parameter on APPC and printer devices has been changed to *CTLD. This means the messages sent for APPC devices and printer devices will be sent to the message queue defined by the MSGQ parameter on the controller description to which the APPC or printer device is attached.

Other device types do not have a MSGQ parameter. For these devices, their messages that used to be sent to QSYSOPR will now be sent to the message queue that is being used by the controller description to which they are attached.

There are a few considerations that you should be aware of when thinking about where you want your messages to be sent.

- Messages sent for APPN communications that are not directly associated with a specific configuration object will be sent to the message queue as defined by the QCFGMSGQ system value.
- There are several configuration object types that do not have a MSGQ parameter. For example, bisync, async, and TDLC lines do not have this parameter. Lines that do not have a MSGQ parameter will have their messages sent to the message queue as defined by the QCFGMSGQ system value.
- SNPT devices do not have a MSGQ parameter. Their messages will be sent to the message queue defined by the controller to which they are attached.
- Network controller and device messages will be sent to the message queue of the line to which they are attached.
- When local and remote workstation controllers are created automatically by the system, they will use the command default of MSGQ(*SYSVAL). If you would like to have the messages for automatically created workstation controllers sent to some other message queue, you can change the command defaults to accomplish this.

- APPN automatically created controllers and devices will use the message queue defined by the line to which they are attached.

The general rule is that all the messages for a given communications configuration hierarchy are intended to go to the same message queue.

The following are a few more things to be aware of:

- The message queue being used for a configuration object is determined when the configuration object is varied on. If you wish to change to use a different message queue, you must vary the configuration off and back on again for the change to take affect.
- If you were to set up a configured message queue, and that message queue were to get deleted or damaged, the messages obviously cannot be sent to that message queue. If this occurs, the messages will revert to being sent to QSYSOPR. A message (ID: CP15742) will be logged in QHST indicating that this condition occurred.
- Configuration objects with the MSGQ parameter can be displayed (e.g., DSPLIND or DSPCTLD) to see the message queue. When you do this, you will actually see two message queue parameters. The current message queue, which is the message queue to which messages are currently being sent for this configuration object, and the configured message queue. The configured message queue is the message queue that was specifed on the MSGQ parameter. This may be different that the current message queue under the following circumstances:
  - The originally configured message queue was damaged or deleted.
  - The MSGQ parameter on the configuration object was changed, but the object had not been varied off and back on for the change to take affect.
- The message support does not apply for TCP/IP and IPX communications. Those messages only are sent to QSYSOPR. In addition, messages for network interface descriptions are only sent to QSYSOPR. Network Server descriptions do have a MSGQ parameter, but that message queue has a different use. The communications messages sent regarding the status of the network server descriptions (e.g., varied on) are sent to QSYSOPR.

The support provided by the message queue configuration capability can be very powerful; you can set up a very granular configuration for where messages are sent. However, the more granular the configuration, the more complex it can be. This complexity can have the opposite effect than what is desired; too complex a configuration may make it harder to find your communications messages rather than easier.

The following examples will help you determine how you may want to take advantage of this function in your environment.

- You have an AS/400 with a relatively small number of users (under 100). You have one LAN line. No changes are needed. All communications messages will be sent to QSYSOPR.
- You have an AS/400 with a moderate number of users. You have a couple LAN lines and perhaps one WAN line. You would like to have the communications messages all go to one message queue, but not QSYSOPR.

  In this case, simply change the QCFGMSGQ system value to QSYS/QCFGMSGQ.
- You have an AS/400 with many users. You have many LAN lines and many WAN lines, with many users on each line. You want to move the communications

messages out of QSYSOPR. In addition, you would like to some of the messages to be sent to separate message queues. Perhaps you want all your workstation messages to go to one message queue, your LAN messages to another message queue, your WAN messages to another, and everything else to QCFGMSGQ. In this example, you would:

- Set up your LAN lines to use the LANMSGQ. All manually created controllers attached to these lines also use the LANMSGQ. All automatically created APPN controllers attached to these lines will also use the LANMSGQ.
- Set up your WAN lines to use the WANMSGQ. All controllers created on these lines also use the WANMSGQ.
- Set up your workstation controllers to use the WSMSGQ. You use automatic configuration for local workstation controllers and devices, so you changed the CRTCTLLWS command's MSGQ parameter to default to WSMSGQ, following IBM guidelines for changing command defaults.
- Change the QCFGMSGQ system value to QSYS/QCFGMSGQ. This has all the remaining communitions messages sent to the QCFGMSGQ.

In this scenario, the multiple message queues can make it easier to find the messages related to a specific configuration. For example, if you have a failure on the WAN line, you will know to look for messages in the WANMSGQ. You will not have to wade through workstation or LAN messages to find the WAN messages. However, this scenario also adds complexity because you will need to know which message queue to look in for a particular error condition. Because this configuration has several message queues to manage, you may want to consider writing a simple operator interface to select which message queue to look at for what configuration objects to help manage the additional complexity.

Chapter 3. Tracing and Diagnosing Communications Problems

This chapter provides information about the commands that enable you to trace and diagnose communications problems. These commands perform several functions. Some commands trace communications information specific to a unique application program. These commands are tailored specifically to the application programming interface, for example, TRCCPIC, TRCICF, STRSRVJOB, ENDSRVJOB, and DSPSRVSTS. Some commands are designed to help you understand the physical layout of your Advanced Peer-to-Peer Networking (APPN) network (see “Isolating Problems in APPN Networks” on page 104). Other commands enable you to trace protocols that are exchanged on communications lines, for example, STRCMNTRC, ENDCMNTRC, PRTCMNTRC, DLTCMNTRC, and CHKCMNTRC.

Communications Problem Analysis

The communications hardware configuration and current status can be displayed or printed with the Work with Hardware Products (WRKHDWPRD) and the Work with Configuration Status (WRKCFGSTS) commands. Use this information when monitoring operations or doing problem analysis.

Normally you should review messages in QSYSOPR or the configured message queue, history log (QHST), and the appropriate job log for problem analysis. APPN, remote job entry (RJE), SNADS, and distributed systems node executive (DSNX) have unique problem analysis information. Examine APPN session information, job logs, message queues, and journals when using these functions.

The SNA alert support on the AS/400 system can be used to assist in problem analysis. Optionally, messages can cause alerts to be created that can be displayed using the Work with Alerts (WRKALR) command at the local system or sent to another system supporting alerts. Alerts must be enabled before this command will provide data. The final destination or focal point could be another AS/400 system or the IBM NetView product for review and problem resolution. If necessary, the system displaying the alerts can use display station pass-through for the System/36, System/38, or AS/400 system. The AS/400 system distributed host command facility (DHCF) through the System/370 Host Command Facility (HCF), network routing facility (NRF), or SNA Primary LU2 Support (SPLS) examines problem information on the troubled system.

Error conditions that are communications-related can also make entries in the system problem log. You can access the log by using the Work with Problem (WRKPRB) command, which shows the Work with Problem display. The log lists problems detected by the system or detected by the user. It is used for additional problem analysis, using menu options, or for documenting problem records. Review this log routinely and remove outdated entries.

The Start Copy Screen (STRCPYSCN) command can be used by an authorized work station user to receive copies of displays from another display device. This command can be used by a help desk operator to analyze both system and application problems.

The Problem Management (use the GO CMDPRBMGMT command) menu, the Network Problem Handling (use the GO NETPRB command) menu, and the Work with Problem (WRKPRB) command provide prompting for problem analysis help.

Refer to the appropriate book for additional information:

- For APPN session status, see the *AS/400e Information Center*.
- For messages, copy display (screen) support, and general problem analysis information, see the *System Operation* book.
- For QSYSOPR, the configured message queue, QSYSMSG, operator and programmed handling of messages, job and QHST logs, printing the error log, and system reply list, see the *CL Programming* book.
- For alerts and distributed systems node executive (DSNX), see the *DSNX Support* book.
- For DHCF, see the *Remote Work Station Support* book.
- For SNADS, see the *SNA Distribution Services* book.
- For RJE, see the *Remote Job Entry (RJE) Guide*.

Running Problem Analysis

Some QSYSOPR messages are added with comments designated by an asterisk (*) in the farthest left position of the display (DSPMSG QSYSOPR). For those messages, additional tests can be performed by pressing F14 (Run problem analysis) and using the additional menu prompts.

The Verify Communications (VFYCMN) command enables you to make sure communications hardware is operating correctly. VFYCMN displays a menu of appropriate hardware test procedures for the communications line that you select. The procedures will display instructions for setting up and running the tests.

In addition, you can use the Verify Communications (VFYCMN) command to perform local communications hardware analysis, link tests, link problem determination aid (LPDA-1 and LPDA-2) tests, ITU-T V.54 loop tests, and communications interface trace, which provide the interface status of EIA-232, V.24, and V.35 protocols. These tests can help identify problems caused by the local AS/400 communications hardware, the local modem, the communications line, the remote modem, or the remote controller.

The capability of using IBM LPDA and ITU-T V.54 tests depends on the modem. This capability is indicated in the modem parameter of the line description.

AS/400 system support includes the following modem tests:

- LPDA
  - Local modem self-test
  - Local modem status test
  - Remote modem self-test
  - Local and remote modem status test
- LPDA-2
  - Local and remote modem and line status
  - Local and remote modem and line test
  - Line analysis
  - Transmit and receive test
- ITU-T V.54
  - Loop 3
  - Local 2 (a remote wrap test)

For more information on the VFYCMN command, contact your IBM service representative.

The Verify Link LPDA-2 (VFYLNKLPDA) Command

The Verify Link Supporting LPDA-2 (VFYLNKLPDA) command enables you to run LPDA-2 tests on digital or analog data circuit-terminating equipment (DCEs) that support LPDA-2. The results can be displayed or printed.

The following LPDA-2 tests are available:

- DCEs and line status

  The local and remote DCEs report information on their configuration, parameters of the line, and the current status and previous activity of the DTE interface on the remote DCE. The information is gathered by background tests.

  The DCEs and line status command is supported by both analog and digital DCEs.
- DCEs and line test

  The local and remote DCEs run stand-alone tests and report information on their configuration, parameters of the line, and the current status and previous activity of the DTE interface on the remote DCE.

  This command is supported by both analog and digital DCEs.
- Analyze line

  The local and remote DCEs exchange test patterns on which they measure parameters of the line signals. The report includes the measurements from both DCEs. It also includes the acceptable limits for the parameters.

  The analyze line command is supported by analog DCEs only.
- Send and receive test

  The local and remote DCEs exchange blocks of test patterns and report the number of detected errors.

  The send and receive test command is supported by both analog and digital DCEs.

These tests can be run on the modems during normal usage of the line. Communication is slower while the tests are running, but is not otherwise disrupted. In cases where the tests are not compatible with normal line usage, the VFYLNKDPDA command returns an error message with online help information that describes the incompatibility.

Refer to your LPDA modem documentation for specific information about these tests and the resulting test output.

The Run LPDA-2 (RUNLPDA) Command

In addition to the test and analysis functions available through the VFYLNKLPDA command, LPDA-2 support on the AS/400 system enables you to establish switched network backup support in the event of nonswitched line connection failure using the Run LPDA-2 (RUNLPDA) command. LPDA-2 support is summarized in Table 5 on page 84.

The RUNLPDA command enables you to run a Link Problem Determination Aid-2 (LPDA-2) operational command on local or remote data circuit-terminating equipment (DCE). The RUNLPDA command can be used to:

- Establish or disconnect a switched telephone network connection.

- Open or close the relay contact in a coupler.
- Determine whether a relay contact is open or closed.
- Determine whether electric current is flowing through an internal sensor.
- Change the transmit speed of a DCE to full or backup.

Restrictions:

- The RUNLPDA command is valid only for an analog LPDA-2 DCE attached to a nonswitched SDLC line.
- To use this command, you must sign on as QPGMR, QSYSOPR, QSRV, or QSRVBAS, or have *ALLOBJ authority.

The RUNLPDA OPTION(*CALL) command can establish a switched network backup (SNBU) connection. If a nonswitched line connection fails, a switched connection can be used in its place until the nonswitched connection can be corrected.

The RUNLPDA OPTION(*SETSPEED) command can change the transmit speed of data terminal equipment (DTE) ports on the DCE. Transmission errors that occur when the DCE is transmitting at full speed might not occur at backup speed. Using backup speed can allow communications to continue on a line with poor signal quality.

Table 5. AS/400 LPDA-2 Support Summary

Feature	VFYLNKLPDA	RUNLPDA
Modems and line status, line test	X	
Send and receive test	X	
Line analysis	X	
Call-out		X
Disconnect		X
Set transmit speed		X
Contact sense		X
Contact operate		X

Start System Service Tools (STRSST) Command

It could be necessary to obtain an error log or communications trace data that can be reviewed by either your IBM service representative or, for the line trace, someone familiar with the protocol being used on the line. You can use these additional functions through the system service tools, using the Start System Services Tools (STRSST) command.

Because SST provides other functions as well, only the correctly authorized personnel having been specified with CRTUSRPRF SPCAUT (*SERVICE) should be allowed to use the STRSST command. The system-supplied profiles QSECOFR and QSRV have this authority.

You can trace multiple lines from each work station using the SST communications trace option. A maximum of two lines on the same communications controller subsystem can be traced at the same time. Only one trace can exist for the same configuration object at the same time. All line speeds and protocols are supported.

The SST communications trace function should be used in the following situations:

- Message information or other problem analysis is not giving sufficient problem identification information.
- Communications support personnel suspect a protocol error.
- Verification that valid data is sent to and received from the system.

For more information on these tests, contact your IBM service representative.

To get the error log data, you can use the SST Product Activity Log option. You also can use the Print Error Log (PRTERRLOG) command to print the error log.

SNA Pass-through Error Message Processing

SNA Pass-through uses the current error recovery for SNA lines, that include the host controllers, and the downstream physical device and associated controller. When SNA Pass-through detects a configuration or SNA error, no automatic recovery occurs. The node where the error occurred produces an informational message that is written to the QSYSOPR message queue or the configured message queue. The message describes the error and gives possible recovery actions.

If the error occurs on a source node, the error message is written to the QSYSOPR message queue or the configured message queue on the source node. The source node also sends a message to the terminal user that indicates that a message was processed. The format of the message is `AS/400 xxxxxxx`, where `xxxxxxx` identifies the message that is placed. This provides the SNA Pass-through terminal user with information with which to diagnose a local problem.

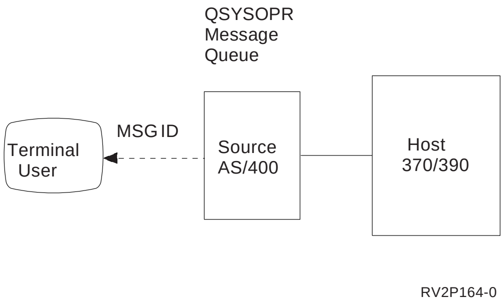


Figure 4. SNA Pass-through Error Messaging, Source Node to Terminal User

In multinode networks (see Figure 5 on page 86), if the error occurs on an intermediate node, the error message is sent back to the source node. The source node processes the informational message that it receives from the intermediate node and produces an informational message that points to the message on the intermediate node. The source node also sends a message to the terminal user that indicates that a message was processed. The format of the message is `AS/400 yyyyyyy`, where `yyyyyyy` identifies the message that is produced on the source node.

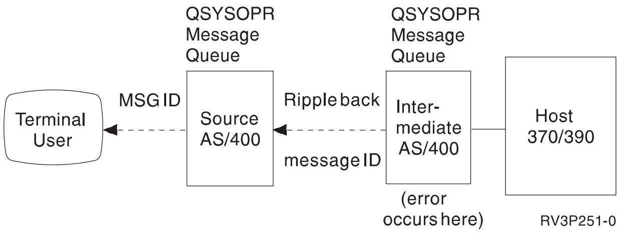


Figure 5. SNA Pass-through Error Messaging, Multinode Network

Tracing Communications Lines, Network Interfaces, and Network Servers

Sometimes program debugging or network management tasks are easier if you can see the data that is sent and received on the communications line or within the network server. You can obtain a communications line trace in various ways. You can use the Start System Service Tools (STRSST) command. (See "Start System Service Tools (STRSST) Command" on page 84 for more information on this command. For more information on how to access SST, see the *Backup and Recovery* book.) You can also use the communications trace commands listed in this section. Refer to the *CL Reference (Abridged)* book for complete information on these commands.

STRCMNTRC
Starts a communications trace for the specified line, network interface description, or network server description. The communications trace continues until:

- The ENDCMNTRC command is run
- A physical line problem causes the trace to end
- The Communications Trace function of the STRSST command is used to end the trace.
- The *STOPTRC parameter is specified and the buffer becomes full.

ENDCMNTRC
Ends the trace currently running on the specified line, network interface description, or network server description, saving the communications trace buffer and the associated System Licensed Internal Code (SLIC) data.

PRTCMNTRC
Writes the communications trace data for the specified line, network interface description, or network server description to a spooled file or a database file. The trace data can be printed multiple times in either form, and parameters on the command allow for subsetting and formatting of the data.

DLTCMNTRC
Deletes the communications trace buffer and associated SLIC data for the specified line, network interface description, or network server description. The communications trace can be deleted once the trace has ended.

CHKCMNTRC
Returns the communications trace status for a specific line, network interface description, or network server description, or for all of the traces of a specific type that exist on the system. The status is returned through a message.

Trace Common Programming Interface Communications

You can use the Trace Common Programming Interface (CPI) Communications (TRCCPIC) command to capture information about CPI-Communications calls that are being processed by your program. The trace information can be collected in a current job or in a job being serviced by a Start Service Job (STRSRVJOB) command.

Service Job Commands that Interact with TRCCPIC

The Start Service Job (STRSRVJOB) command enables you to collect trace records for jobs that are:

- Started from other work stations
- Sent to batch
- Started as a result of program start requests received from remote systems.

After the STRSRVJOB command has been entered, the TRCCPIC command must be entered to start the CPI-Communications trace.

You can use the End Service Job (ENDSRVJOB) command to end the service job request. The trace must be stopped before you can use this command. See the section, “Stopping the Trace” on page 89 for more information.

You can use the Display Service Status (DSPSRVSTS) command to display the status of:

- Trace CPI Communications (TRCCPIC).
- Trace Job (TRCJOB).
- Trace Intersystem Communications Function (TRCICF).
- Debugging jobs. Debug mode is entered by issuing the Start Debug (STRDBG) command.

All of these traces must be ended before issuing an End Service Job (ENDSRVJOB) request.

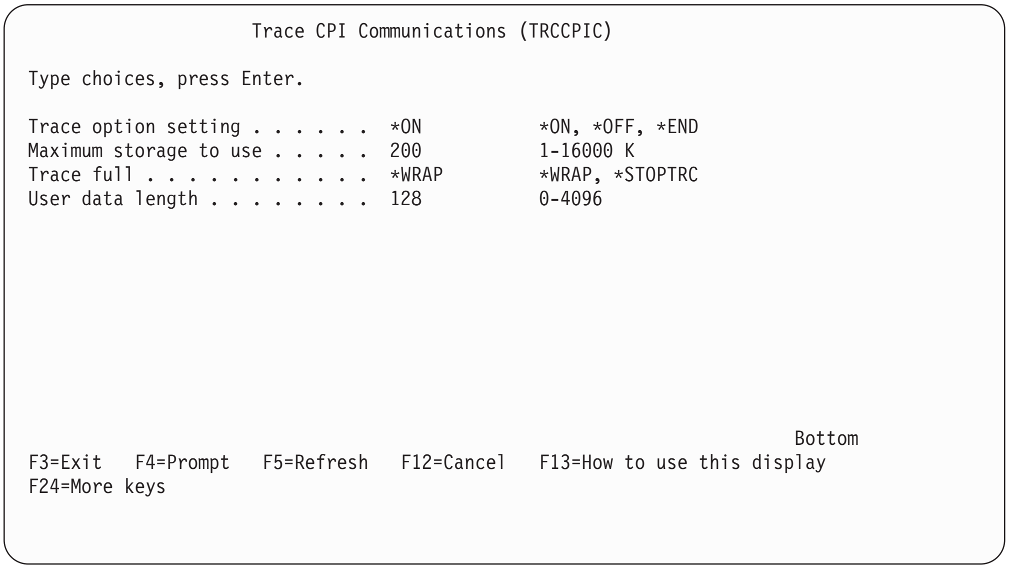
More information about these service commands is in the *CL Reference (Abridged)* book.

Starting Trace CPI Communications

You can start Trace CPI Communications either before running a job or after a job is active (for example, when a job is started as a result of a received program start request). You can issue the TRCCPIC command by:

- Using the System Menu
- Typing TRCCPIC *ON on the command line
- Adding the TRCCPIC command to a CL or a REXX program
- Typing TRCCPIC on the command line and pressing F4 (Prompt)

If you type TRCCPIC on the command line and press F4, an initial prompt is displayed for the *Trace Option Setting*. If *ON is specified and you press enter, the following is displayed.



```
                     Trace CPI Communications (TRCCPIC)

 Type choices, press Enter.

 Trace option setting . . . . . .  *ON           *ON, *OFF, *END
 Maximum storage to use . . . . .  200           1-16000 K
 Trace full . . . . . . . . . . .  *WRAP         *WRAP, *STOPTRC
 User data length . . . . . . . .  128           0-4096










                                                                         Bottom
 F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display
 F24=More keys
```

*Figure 6. TRCCPIC prompt with trace option setting set to *ON*

This display enables you to set the following parameters:

Trace option setting

Specifies whether the collection of trace information is to be started, stopped, or ended.

***ON**

Starts Trace CPI Communications. This is the default value for the prompt.

***OFF**

Stops Trace CPI Communications. The current information is written to the spooled printer file or to the database file, and the trace table and trace information are then deleted.

***END**

Ends Trace CPI Communications. The trace table and all trace information are destroyed.

Maximum storage to use

Specifies the maximum amount of storage to use for the trace information collected. The prompt only appears if you have selected *ON for the *Trace option setting* prompt.

200 K

The number of bytes (1 K equals 1024 bytes) of storage. This is the default value.

1-16000 K

The valid range for the maximum number of bytes used for storing collected trace information.

Trace full

Specifies whether new trace records replace old trace records or whether the trace is stopped when the maximum storage that you specified has been reached. This prompt only appears if you have selected *ON for the *Trace option setting* prompt.

***WRAP**
When the trace storage area is full, new trace information is written over the old trace information, starting at the beginning of the storage area. This is the default value.

***STOPTRC**
No new trace information is saved when the trace storage area is full. You must reissue the TRCCPIC command, specifying (*OFF) for the SET parameter, before you can retrieve the output of the trace information collected in the trace storage area.

User data length
Specifies the maximum length of user data to be saved for each trace record in the storage area. This prompt only affects the tracing of user data on the Send_Data and Receive calls. This parameter does not affect the tracing of log data on Set_Log_Data, Send_Error, or Deallocate calls. This prompt appears only if you specified *ON on the *Trace option setting* prompt.

128
The number of bytes for the user data length. This is the default value.

0-4096
The valid range of bytes for the user data length.

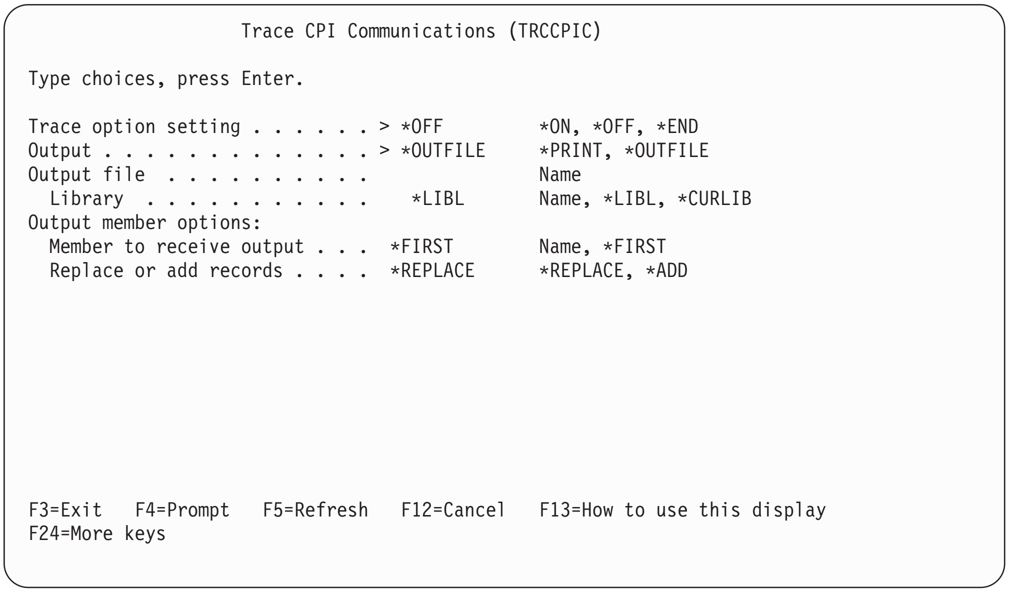
Stopping the Trace

Trace CPI Communications continues to collect trace records until you stop the trace or until the trace storage area becomes full. This depends on the value that is specified on the *Trace full* prompt. If the trace storage area becomes full and the collection of trace records stops, you must enter the TRCCPIC command again to create output. The output created by the TRCCPIC command is directed either to the spooled printer file, QSYSPRT, or to a database output file that you specify. If the output file that you specify already exists, it must have the same attributes as the system-supplied file, QACMOTRC.

You can stop a trace procedure by:

- Using the System Menu
- Typing TRCCPIC *OFF on the command line
- Adding the TRCCPIC command to a CL or a REXX program
- Typing TRCCPIC on the command line and pressing F4 (Prompt)

If you type TRCCPIC on the command line and press F4, and you specify *OFF for the *Trace option setting*, you are prompted for the OUTPUT parameter. If you specify the *OUTFILE value for the *Output* prompt, the following is displayed:



```
                    Trace CPI Communications (TRCCPIC)

Type choices, press Enter.

Trace option setting . . . . . . > *OFF         *ON, *OFF, *END
Output . . . . . . . . . . . . . > *OUTFILE     *PRINT, *OUTFILE
Output file  . . . . . . . . . .                Name
  Library  . . . . . . . . . . .    *LIBL       Name, *LIBL, *CURLIB
Output member options:
  Member to receive output . . .  *FIRST        Name, *FIRST
  Replace or add records . . . .  *REPLACE      *REPLACE, *ADD









F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display
F24=More keys
```

*Figure 7. TRCCPIC prompt with trace option setting set to *OFF*

This display enables you to set the following parameters:

Output

Specifies whether the trace information is to be stored in a spooled file or saved in a database file. This prompt only appears if *OFF is specified on the *Trace option setting* prompt.

***PRINT**

The trace information is sent to the spooled file QSYSPRT in the output queue associated with the job issuing the command. The spooled file can be viewed or printed. Refer to Figure 8 on page 92 for an example of the spooled trace records. This is the default value.

***OUTFILE**

The trace records are to be directed to a database file. Refer to Figure 9 on page 100 for a description of trace records directed to a database file. The *OUTFILE value on the *Output* prompt is only valid if a value is specified for the *Output file* prompt.

Output file

Specifies the name of the database file to which trace records are to be written. This prompt only appears if you have selected *OFF on the *Trace option setting* prompt and *OUTFILE on the *Output* prompt. If the file does not exist, the system creates a new database file with the specified name in the specified library. The new file has the same attributes as the system-supplied file QACMOTRC. If the file already exists, it must have the same attributes as the system supplied file. Possible library values are:

Name

The name of the library where the file is located.

***LIBL**

The file is located in the library list.

***CURLIB**

The file is located in the current library for the job. If no current library entry exists in the library list, the library QGPL is used.

Output member options

Specifies the name of the file member that is to receive the trace information. This prompt only appears if you have selected *OFF for the *Trace option setting* prompt and *OUTFILE for the *Output* prompt. If the output file is to be created by the system, an output member is also created and given the name specified in the *Member to receive output* prompt. If *FIRST is specified for the *Member to receive output* prompt, a member is created and given the name specified in the output file. If the output file exists, but the output member does not, a member with the specified name is created. The options for the *Output member options* prompt are:

Member to receive output

Type the name of the member to receive the output.

<u>*FIRST</u>

The first member in the output file receives the trace information. This is the default.

Name

The specified member receives the trace information.

Replace or add records

The trace information either replaces the existing trace information or is added to the file.

<u>*REPLACE</u>

New trace information replaces trace information already in the file member. This is the default.

***ADD**

New trace information is added to the end of data already in the file member.

Sending Trace Records to a Spooled File

When you select *OFF on the *Trace option setting* prompt and press F4 (Prompt), you are presented with the option on the *Output* prompt to write the trace records to a spooled file (*PRINT) or to a database file (*OUTFILE). The default value is *PRINT. If you choose the *PRINT value on the *Output* prompt, the trace information is sent to the spooled file QSYSPRT.

Figure 8 on page 92 provides an example of a Trace CPI Communications report. An explanation of the numbered items in the report begins on page 98.

```
                          Trace CPI Communications
                           1  SYSTEM27          2 10:34:33   3  PAGE 1

 4  Job:   QPADEV0001    5  User:   QPGMR     6  Number:  003117


  Time  . . . . . . . . . . . . . . :   10:34:02.482   7
  Program issuing call  . . . . . . :   T8189CRS       8
    Library . . . . . . . . . . . . :     QGPL
  CPI Communications call . . . . . :   CMINIT         9
  Conversation_ID . . . . . . . . . :   D             10
  Conversation_state  . . . . . . . :   INITIALIZE    11
  Return_code . . . . . . . . . . . :   CM_OK         12
  Other:   13
    Sym_dest_name . . . . . . . . . :   T8189CSI      14
      Library . . . . . . . . . . . :     QGPL
```

Figure 8. An Example of a Trace CPI Communications Report (Part 1 of 29)

```
  Time  . . . . . . . . . . . . . . :   10:34:02.495
  Program issuing call  . . . . . . :   T8189CRS
    Library . . . . . . . . . . . . :     QGPL
  CPI Communications call . . . . . :   CMSTPN
  Conversation_ID . . . . . . . . . :   D
  Conversation_state  . . . . . . . :   INITIALIZE
   Return_code . . . . . . . . . .  :   CM_OK
  Other:
    TP_name_length  . . . . . . . . :   8
    TP_name . . . . . . . . . . . . :   T8189CRT    15
```

Figure 8. An Example of a Trace CPI Communications Report (Part 2 of 29)

```
  Time  . . . . . . . . . . . . . . :   10:34:02.509
  Program issuing call  . . . . . . :   T8189CRS
    Library . . . . . . . . . . . . :     QGPL
  CPI Communications call . . . . . :   CMSSL
  Conversation_ID . . . . . . . . . :   D
```

Figure 8. An Example of a Trace CPI Communications Report (Part 3 of 29)

```
  Conversation_state  . . . . . . . :   INITIALIZE
  Return_code . . . . . . . . . . . :   CM_OK
  Other:
    Sync_level  . . . . . . . . . . :   CM_CONFIRM    16
```

Figure 8. An Example of a Trace CPI Communications Report (Part 4 of 29)

```
Time  . . . . . . . . . . . . . . . :   10:34:02.744
Program issuing call  . . . . . . . :   T8189CRS
  Library . . . . . . . . . . . . . :     QGPL
CPI Communications call . . . . . . :   CMALLC
Conversation_ID . . . . . . . . . . :   D
Conversation_state  . . . . . . . . :   SEND
Return_code . . . . . . . . . . . . :   CM_OK
Other:
  Partner_LU_name_length  . . . . . :   11
  Partner_LU_name . . . . . . . . . :   RPC.T8189LA     17
  Mode_name_length  . . . . . . . . :   0
  Mode_name . . . . . . . . . . . . :                   18
  TP_name_length  . . . . . . . . . :   8
  TP_name . . . . . . . . . . . . . :   T8189CRT
  Conversation_type . . . . . . . . :   CM_MAPPED_CONVERSATION  19
  Sync_level  . . . . . . . . . . . :   CM_CONFIRM
  Device  . . . . . . . . . . . . . :   T8189DEV1       20
  Local location  . . . . . . . . . :   T8189NY         21
  Local network ID  . . . . . . . . :   RPC             22
```

Figure 8. An Example of a Trace CPI Communications Report (Part 5 of 29)

```
Time  . . . . . . . . . . . . . . . :   10:34:02.757
Program issuing call  . . . . . . . :   T8189CRS
  Library . . . . . . . . . . . . . :     QGPL
CPI Communications call . . . . . . :   CMSST
Conversation_ID . . . . . . . . . . :   D
```

Figure 8. An Example of a Trace CPI Communications Report (Part 6 of 29)

```
Conversation_state  . . . . . . . . :   SEND
Return_code . . . . . . . . . . . . :   CM_OK
Other:
  Send_type . . . . . . . . . . . . :   CM_SEND_AND_PREP_TO_RECEIVE   23
```

Figure 8. An Example of a Trace CPI Communications Report (Part 7 of 29)

```
Time  . . . . . . . . . . . . . . . :   10:34:06.086
Program issuing call  . . . . . . . :   T8189CRS
  Library . . . . . . . . . . . . . :     QGPL
CPI Communications call . . . . . . :   CMSEND
Conversation_ID . . . . . . . . . . :   D
Conversation_state  . . . . . . . . :   RECEIVE
Return_code . . . . . . . . . . . . :   CM_OK
```

Figure 8. An Example of a Trace CPI Communications Report (Part 8 of 29)

```
 Other:
   Send_type . . . . . . . . . . . :   CM_SEND_AND_PREP_TO_RECEIVE
   Prepare_to_receive_type . . . . :   CM_PREP_TO_RECEIVE_SYNC_LEVEL  24
   Sync_level  . . . . . . . . . . :   CM_CONFIRM
   Send_length . . . . . . . . . . :   5
   Request_to_send_received  . . . :   CM_REQ_TO_SEND_NOT_RECEIVED  25
   Buffer  . . . . . . . . . . . . :   03485         26
```

Figure 8. An Example of a Trace CPI Communications Report (Part 9 of 29)

```
 Time  . . . . . . . . . . . . . . :   10:34:06.191
 Program issuing call  . . . . . . :   T8189CRS
   Library . . . . . . . . . . . . :     QGPL
```

Figure 8. An Example of a Trace CPI Communications Report (Part 10 of 29)

```
 CPI Communications call . . . . . :   CMRCV
 Conversation_ID . . . . . . . . . :   D
 Conversation_state  . . . . . . . :   SEND-PENDING
 Return_code . . . . . . . . . . . :   CM_OK
```

Figure 8. An Example of a Trace CPI Communications Report (Part 11 of 29)

```
Other:
   Receive_type  . . . . . . . . . :   CM_RECEIVE_AND_WAIT       27
   Requested_length  . . . . . . . :   25                        28
   Data_received . . . . . . . . . :   CM_COMPLETE_DATA_RECEIVED  29
   Received_length . . . . . . . . :   25                        30
   Status_received . . . . . . . . :   CM_SEND_RECEIVED          31
   Request_to_send_received  . . . :   CM_REQ_TO_SEND_NOT_RECEIVED
   Buffer  . . . . . . . . . . . . :   Coding pencil (sharpened)
```

Figure 8. An Example of a Trace CPI Communications Report (Part 12 of 29)

```
 Time  . . . . . . . . . . . . . . :   10:34:11.024
 Program issuing call  . . . . . . :   T8189CRS
   Library . . . . . . . . . . . . :     QGPL
 CPI Communications call . . . . . :   CMSEND
 Conversation_ID . . . . . . . . . :   D
 Conversation_state  . . . . . . . :   RECEIVE
 Return_code . . . . . . . . . . . :   CM_OK
```

Figure 8. An Example of a Trace CPI Communications Report (Part 13 of 29)

```
 Other:
   Send_type . . . . . . . . . . . :   CM_SEND_AND_PREP_TO_RECEIVE
   Prepare_to_receive_type . . . . :   CM_PREP_TO_RECEIVE_SYNC_LEVEL
   Sync_level  . . . . . . . . . . :   CM_CONFIRM
   Send_length . . . . . . . . . . :   5
   Request_to_send_received  . . . :   CM_REQ_TO_SEND_NOT_RECEIVED
   Buffer  . . . . . . . . . . . . :   03486
```

Figure 8. An Example of a Trace CPI Communications Report (Part 14 of 29)

```
 Time  . . . . . . . . . . . . . . :   10:34:11.064
 Program issuing call  . . . . . . :   T8189CRS
   Library . . . . . . . . . . . . :     QGPL
 CPI Communications call . . . . . :   CMRCV
 Conversation_state  . . . . . . . :   SEND-PENDING
 Return_code . . . . . . . . . . . :   CM_OK
```

Figure 8. An Example of a Trace CPI Communications Report (Part 15 of 29)

```
Other:
   Receive_type  . . . . . . . . . :   CM_RECEIVE_AND_WAIT
   Requested_length  . . . . . . . :   25
   Data_received . . . . . . . . . :   CM_COMPLETE_DATA_RECEIVED
   Received_length . . . . . . . . :   25
   Status_received . . . . . . . . :   CM_SEND_RECEIVED
   Request_to_send_received  . . . :   CM_REQ_TO_SEND_NOT_RECEIVED
   Buffer  . . . . . . . . . . . . :   Eraser (worn)
```

Figure 8. An Example of a Trace CPI Communications Report (Part 16 of 29)

```
 Time  . . . . . . . . . . . . . . :   10:34:16.983
 Program issuing call  . . . . . . :   T8189CRS
   Library . . . . . . . . . . . . :     QGPL
 CPI Communications call . . . . . :   CMSEND
 Conversation_ID . . . . . . . . . :   D
 Conversation_state  . . . . . . . :   RECEIVE
 Return_code . . . . . . . . . . . :   CM_OK
```

Figure 8. An Example of a Trace CPI Communications Report (Part 17 of 29)

```
Other:
  Send_type . . . . . . . . . . . :   CM_SEND_AND_PREP_TO_RECEIVE
  Prepare_to_receive_type . . . . :   CM_PREP_TO_RECEIVE_SYNC_LEVEL
  Sync_level  . . . . . . . . . . :   CM_CONFIRM
  Send_length . . . . . . . . . . :   5
  Request_to_send_received  . . . :   CM_REQ_TO_SEND_NOT_RECEIVED
  Buffer  . . . . . . . . . . . . :   03487
```

Figure 8. An Example of a Trace CPI Communications Report (Part 18 of 29)

```
Time  . . . . . . . . . . . . . . :   10:34:17.040
Program issuing call  . . . . . . :   T8189CRS
  Library . . . . . . . . . . . . :     QGPL
CPI Communications call . . . . . :   CMRCV
Conversation_ID . . . . . . . . . :   D
Conversation_state  . . . . . . . :   SEND-PENDING
Return_code . . . . . . . . . . . :   CM_OK
```

Figure 8. An Example of a Trace CPI Communications Report (Part 19 of 29)

```
Other:
  Receive_type  . . . . . . . . . :   CM_RECEIVE_AND_WAIT
  Requested_length  . . . . . . . :   25
  Data_received . . . . . . . . . :   CM_COMPLETE_DATA_RECEIVED
  Received_length . . . . . . . . :   25
  Status_received . . . . . . . . :   CM_SEND_RECEIVED
  Request_to_send_received  . . . :   CM_REQ_TO_SEND_NOT_RECEIVED
  Buffer  . . . . . . . . . . . . :   Copy holder (adjustable)
```

Figure 8. An Example of a Trace CPI Communications Report (Part 20 of 29)

```
Time  . . . . . . . . . . . . . . :   10:34:21.057
Program issuing call  . . . . . . :   T8189CRS
  Library . . . . . . . . . . . . :     QGPL
CPI Communications call . . . . . :   CMSEND
Conversation_ID . . . . . . . . . :   D
Conversation_state  . . . . . . . :   RECEIVE
Return_code . . . . . . . . . . . :   CM_PROGRAM_ERROR_PURGING
```

Figure 8. An Example of a Trace CPI Communications Report (Part 21 of 29)

```
Other:
  Send_type . . . . . . . . . . . :   CM_SEND_AND_PREP_TO_RECEIVE
  Prepare_to_receive_type . . . . :   CM_PREP_TO_RECEIVE_SYNC_LEVEL
  Sync_level  . . . . . . . . . . :   CM_CONFIRM
  Send_length . . . . . . . . . . :   5
  Request_to_send_received  . . . :   CM_REQ_TO_SEND_NOT_RECEIVED
  Buffer  . . . . . . . . . . . . :   03489
```

Figure 8. An Example of a Trace CPI Communications Report (Part 22 of 29)

```
Time  . . . . . . . . . . . . . . :   10:34:21.100
Program issuing call  . . . . . . :   T8189CRS
  Library . . . . . . . . . . . . :     QGPL
CPI Communications call . . . . . :   CMRCV
Conversation_ID . . . . . . . . . :   D
Conversation_state  . . . . . . . :   SEND-PENDING
Return_code . . . . . . . . . . . :   CM_OK
```

Figure 8. An Example of a Trace CPI Communications Report (Part 23 of 29)

```
Other:
  Receive_type  . . . . . . . . . :   CM_RECEIVE_AND_WAIT
  Requested_length  . . . . . . . :   40
  Data_received . . . . . . . . . :   CM_COMPLETE_DATA_RECEIVED
```

Figure 8. An Example of a Trace CPI Communications Report (Part 24 of 29)

```
  Received_length . . . . . . . . :   40
  Status_received . . . . . . . . :   CM_SEND_RECEIVED
  Request_to_send_received  . . . :   CM_REQ_TO_SEND_NOT_RECEIVED
  Buffer  . . . . . . . . . . . . :   The requested part was not found.
```

Figure 8. An Example of a Trace CPI Communications Report (Part 25 of 29)

```
Time  . . . . . . . . . . . . . . :   10:34:28.238
Program issuing call  . . . . . . :   T8189CRS
  Library . . . . . . . . . . . . :     QGPL
CPI Communications call . . . . . :   CMSDT
Conversation_ID . . . . . . . . . :   D
Conversation_state  . . . . . . . :   SEND-PENDING
Return_code . . . . . . . . . . . :   CM_OK
```

Figure 8. An Example of a Trace CPI Communications Report (Part 26 of 29)

```
Other:
  Deallocate_type . . . . . . . . :   CM_DEALLOCATE_FLUSH   32
```

Figure 8. An Example of a Trace CPI Communications Report (Part 27 of 29)

```
Time  . . . . . . . . . . . . . . :   10:34:28.346
Program issuing call  . . . . . . :   T8189CRS
  Library . . . . . . . . . . . . :     QGPL
CPI Communications call . . . . . :   CMDEAL
Conversation_ID . . . . . . . . . :   D
Conversation_state  . . . . . . . :   RESET
Return_code . . . . . . . . . . . :   CM_OK
```

Figure 8. An Example of a Trace CPI Communications Report (Part 28 of 29)

```
Other:
  Deallocate_type . . . . . . . . :   CM_DEALLOCATE_FLUSH
```

Figure 8. An Example of a Trace CPI Communications Report (Part 29 of 29)

The following explains the content of the sample Trace CPI Communications report. The reference numbers in the explanation below correspond to the numbers in the sample report.

1 The system name.

2 The date on which the output was created.

3 The time at which the output was created.

4 The job name of the job that is being traced.

5 The user identification (user ID) used to start the job that is being traced.

6 The job number assigned to the job step when the job being traced started.

7 The time at which the call was issued.

8 The name of the library where the program resides, and the name of the program that issued the call that is being traced. When the program that issues the CPI Communications call is a REXX program, REXX/400 is shown as the program name, and the library name is not printed. When a CPI Communications call is issued for your program by the system, OS/400 program is shown as the program name, and the library name is not printed. A Deallocate call can be issued by the system for your program when your job ends while conversations are active. The Deallocate call also can be issued when a Reclaim Resource (RCLRSC) command is issued in a call level higher than the one in which the conversation originated.

9 The CPI Communications call that was issued.

10 The *conversation_ID* on which the call was issued.

11 The *conversation_state* that exists for the conversation at the completion of the call.

12 The *return_code* indicating the success or failure of the call.

13 This section is printed when there are parameters or characteristics that affect the call. Only the parameters that affect the call are printed. If you expect to see a characteristic on a traced call, and it is not printed, then you may be using the characteristic incorrectly.

Note: If a *return_code* for a given call renders a parameter meaningless, that parameter is not printed. For example, *data_received* is not printed when the *return_code* on the Receive call is not CM_OK or CM_DEALLOCATED_NORMAL.

If the value of a characteristic refers to another characteristic, both are printed. For example, on a Send_Data call, if the *send_type* is CM_SEND_AND_PREPARE_TO_RECEIVE, then the *prepare_to_receive_type* is printed as well. If the *send_type* is CM_BUFFER_DATA, the *prepare_to_receive_type* is not printed.

14 The symbolic destination name (*sym_dest_name*) and the library in which it was found. If the communications side information object identified by the *sym_dest_name* was not found, *LIBL is the value printed for Library.

15 The *TP_name*, which is the name of the partner program.

16 The *sync_level* that is set for the conversation.

17 The *partner_LU_name*, which is the remote network identifier concatenated with the remote location name. It identifies the remote system on which the *TP_name* resides.

18 The *mode_name* that is used for the conversation. Note that the *mode_name_length* of 0 indicates that a *mode_name* of 8 space characters was actually used. If a *mode_name* of “BLANK” is used in the side information, CPI Communications converts it to 8 space characters.

19 The *conversation_type* that is being used for the conversation.

20 The name of the device description that was selected when the route to the remote system was resolved.

21 The local location name configured in the device description that was selected.

22 The local network identifier that is configured in the network attributes of the local system.

23 The *send_type* that is set for the conversation.

24 The *prepare_to_receive_type* and the *sync_level*. They are printed here because the *send_type* refers to the *prepare_to_receive_type* which refers to the *sync_level*.

25 The *request_to_send_received*, which indicates whether the partner program has issued a Request_to_Send call.

26 The *buffer* of user data. The amount of data printed for this characteristic is dependent on the amount of data present (in this case on the *send_length*) and on the value specified for the DTALEN parameter of the TRCCPIC command.

27 The *receive_type* that was used on the Receive call.

28 The *requested_length* that was used on the Receive call.

29 The *data_received* value that was returned when the Receive call completed.

30 The *received_length* that was returned when the Receive call completed.

31 The *status_received* value that was returned when the Receive call completed.

32 The *deallocate_type* that is being set.

The following is an example of the outfile record format for Trace CPI Communications records written to a database file:

```
 *
 * TRACE CPI COMMUNICATIONS (TRCCPIC) OUTFILE RECORD FORMAT
 *
A                                      CCSID(65535)
A          R CM0RCD                    TEXT('TRCCPIC Record')
A            CM0SYS          8         COLHDG('System' 'name')
A            CM0JOB         10         COLHDG('Job' 'name')
A            CM0USR         10         COLHDG('User' 'name')
A            CM0NBR          6         COLHDG('Job' 'number')
A            CM0CEN          1         COLHDG('Retrieval' 'century')
A                                      TEXT('Retrieval century:  +
A                                      0=20th, 1=21st')
A            CM0DAT          6         COLHDG('Date')
A                                      TEXT('Date of entry:  +
A                                      Year/Month/Day')
A            CM0TIM          9         COLHDG('Time')
A                                      TEXT('Time of entry:  +
A                                      hour/minute/second/millisecond')
A            CM0PGM         10         COLHDG('Program' 'name')
A                                      TEXT('Name of program')
```

Figure 9. DDS for trace records written to a database file (Part 1 of 4)

```
A            CM0LIB        10          COLHDG('Library' 'name')
A                                      TEXT('Library where +
A                                      program resides')
A            CM0CID         8          COLHDG('Conversation_' 'ID')
A                                      TEXT('conversation_ID')
A            CM0CAL        10          COLHDG('Call')
A                                      TEXT('CPI Communications call')
A            CM0CST         9B         COLHDG('Conversation_' 'state')
A                                      TEXT('conversation_state')
A            CM0RC          9B         COLHDG('Return_' 'code')
A                                      TEXT('return_code')
A            CM0CTY         9B         COLHDG('Conversation_' 'type')
A                                      TEXT('conversation_type')
A            CM0SLN         9B         COLHDG('Send_' 'length')
A                                      TEXT('send_length')
A            CM0DRV         9B         COLHDG('Data_' 'received')
A                                      TEXT('data_received')
A            CM0DTY         9B         COLHDG('Deallocate_' 'type')
A                                      TEXT('deallocate_type')
A            CM0EDI         9B         COLHDG('Error_' 'direction')
A                                      TEXT('error_direction')
A            CM0FIL         9B         COLHDG('Fill')
A            CM0LGL         9B         COLHDG('Log_' 'data_' 'length')
A                                      TEXT('log_data_length')
A            CM0LGD       512          COLHDG('Log_data')
```

Figure 9. DDS for trace records written to a database file (Part 2 of 4)

```
A            CM0MDL         9B         COLHDG('Mode_' 'name_' 'length')
A                                      TEXT('mode_name_length')
A            CM0MDN         8          COLHDG('Mode_' 'name')
A                                      TEXT('mode_name')
A            CM0PLL         9B         COLHDG('Partner_' 'LU_name_' +
A                                      'length')
A                                      TEXT('partner_LU_name_length')
A            CM0PLN        17          COLHDG('Partner_LU_name')
A            CM0PTR         9B         COLHDG('Prepare_to_' +
A                                      'receive_' 'type')
A                                      TEXT('prepare_to_receive_type')
A            CM0RTY         9B         COLHDG('Receive_' 'type')
A                                      TEXT('receive_type')
A            CM0RLN         9B         COLHDG('Receive_' 'length')
A                                      TEXT('receive_length')
A            CM0RQL         9B         COLHDG('Requested_' 'length')
A                                      TEXT('requested_length')
A            CM0RTS         9B         COLHDG('Request_' 'to_send_' +
A                                      'received')
A                                      TEXT('request_to_send_received')
```

Figure 9. DDS for trace records written to a database file (Part 3 of 4)

```
A            CM0RCT          9B        COLHDG('Return_' 'control')
A                                      TEXT('return_control')
A            CM0STY          9B        COLHDG('Send_' 'type')
A                                      TEXT('send_type')
A            CM0SRC          9B        COLHDG('Status_' 'received')
A                                      TEXT('status_received')
A            CM0SYN          9B        COLHDG('Sync_' 'level')
A                                      TEXT('sync_level')
A            CM0SNM          8         COLHDG('Sym_dest_' 'name')
A                                      TEXT('sym_dest_name')
A            CM0SYL         10         COLHDG('Library')
A                                      TEXT('Library where side +
A                                      information object resides')
A            CM0TPL          9B        COLHDG('TP_name_' 'length')
A                                      TEXT('TP_name_length')
A            CM0TPN         64         COLHDG('TP_name')
A            CM0DEV          8         COLHDG('Device')
A            CM0LCL         10         COLHDG('Local' 'location')
A            CM0LNT         10         COLHDG('Local' 'network' 'ID')
A            CM0RES         81         COLHDG('Reserved')
A            CM0TLN          9B        COLHDG('Traced' 'data' 'length')
A                                      TEXT('Length of user data +
A                                      traced')
A            CM0BUF       4096         COLHDG('Buffer')
```

Figure 9. DDS for trace records written to a database file (Part 4 of 4)

Ending the Trace

You can end Trace CPI Communications by:

- Using the System Menu
- Typing TRCCPIC *END on the command line
- Adding the TRCCPIC command to a CL or a REXX program
- Typing TRCCPIC on the command line and pressing F4 (Prompt) to show the *Trace option setting* prompt on the Trace CPI Communications display. Type *END and press the Enter key. This causes TRCCPIC to end tracing and delete all trace records. No output is created.

Additional Considerations

The TRCCPIC command only prints the information that is relevant to the call being traced. If you are not seeing information printed on a given CPI Communications call, research the usage notes in the *Common Programming Interface Communications Reference* for that call.

When output is directed to a user-specified database file, each record contains a field for each of the CPI Communications characteristics, as well as the system specific fields. All characteristics are stored for each of the records when possible.

Some *return_codes*, like CM_PROGRAM_PARAMETER_CHECK, make it impossible to trace all of the characteristics. In these cases, the field will contain binary zeros. If a characteristic is available to be traced, the contents are placed in the appropriate field, even if the characteristic is not used for the call that is traced in that record. For example, if a Send_Data call is traced and the *return_code* is CM_OK, the *receive_type* characteristic will still be traced. Characteristics that cannot be set by a Set call will not be traced unless they are parameters on the call being traced. For the Send_Data call in our previous example, the *data_received* characteristic would not be traced because it is not a characteristic that can be set by issuing a Set call, and therefore it is not saved between calls. Instead the value would be binary zeros. User programs that process these database files should be designed with this in mind, and should only use a field when the field has some bearing on the call that is traced in that record.

You may need to vary the values that you specify for the maximum storage (MAXSTG) and the data length (DTALEN) parameters on the TRCCPIC command, in order to capture all the trace information that you need.

Storage Considerations

The trace storage header area for control information requires 64 bytes. Each trace record that is logged requires approximately the following amount of storage:

- 292 bytes to trace control information and all the CPI Communications characteristics except *log_data* and *buffer*.
- Up to 512 bytes per record to save the *log_data*. If *log_data* is set with the Set_Log_Data call, it is included in each internal trace record until the *log_data* is reset by a Send_Error, a Deallocate, or a Set_Log_Data call. The DTALEN parameter of the TRCCPIC command has no effect on whether the *log_data* is saved internally.
- Up to 4096 bytes per record for saving the user data contained in the *buffer* characteristic. The amount of data that is saved is controlled by the DTALEN parameter of the TRCCPIC command. That is, if DTALEN(0) is specified, no user data is saved.

Given the above guidelines, you can use the parameters on the TRCCPIC command to ensure that the data you want traced is retained.

For example, if you want to estimate the trace storage requirements for a program and you know that:

- The program makes 200 CPI Communications calls, most of which are Send_Data and Receive_Data
- The *send_length* and *received_length* are usually 100 bytes
- The program is not setting *log_data*

You would calculate the storage as follows:

```
storage = (200 records * (292 + 100)) + 64
        = 78464
```

The TRCCPIC command variation would be:

```
TRCCPIC SET(*ON) MAXSTG(79) DTALEN(100))
```

The TRCCPIC command requires that you allocate at least enough storage to trace one record. The storage for that one record is calculated as follows:

- 64 bytes for the header
- 292 bytes for the control information and characteristics
- 512 bytes for possible *log_data*
- X bytes (specified by the DTALEN parameter on the TRCCPIC command) for user data contained in *buffer*
- n bytes for rounding because each record is fixed length
- The calculation is

```
Record storage = 292 + 512 + X + n
```

where n rounds the value to a multiple of 292.

- Minimum storage = 64 + record storage

Trace Intersystem Communications Function

You can use the Trace Intersystem Communications Function (TRCICF) command to trace communications information concerning the intersystem communications function (ICF) operations and functions that are used by a user program. The trace information can be collected in the current job or in the job being serviced by a Start Service Job (STRSRVJOB) command. The TRCICF command is similar to the TRCCPIC command. Detailed information concerning the TRCICF command and the output it creates can be found in the *ICF Programming*.

Isolating Problems in APPN Networks

Isolating a routing problem in an APPN network can be challenging. The Display APPN Information (DSPAPPNINF) command can assist you in understanding the topology of the network, determining the names of all known remote control points and their locations, displaying intermediate sessions, and displaying link status information.

The Work with APPN Status (WRKAPPNSTS) command enables users to view session, configuration, and route information. It allows users to examine network attributes, mode, COS, and topology information when expecting any of the following:

- Sessions that are using different remote or local locations
- A different path (which is shown only for rapid transport protocol connections)
- More or fewer sessions
- A different mode or class of service (COS) associated with the session
- APPN to be running instead of HPR (or vice versa)

The WRKAPPNSTS command also can be used to end RTP connections or to attempt to switch the path that is being used by the RTP connections. See the APPC, APPN, and HPR articles in the *AS/400e Information Center* for more detail.

When you encounter problems that indicate that the route to the remote location cannot be found, you can attempt to make the connection again with the Start Pass-Through (STRPASTHR) command. The STRPASTHR command has built-in diagnostic capabilities that exceed those provided by other interfaces that utilize APPN networks. These diagnostic capabilities include problem analysis and problem logging and error logging functions.

When a STRPASTHR command fails to contact a remote location on an APPN network, a record is written to the problem log. That is, if there is an associated problem analysis for analyzing the data. If you are running your own APPC application and receive a route initiation failure, use this information to assist with debugging. You can rerun the same mode and remote location information with STRPASTHR. The Work with Problem (WRKPRB) and Analyze Problem (ANZPRB) commands enable you to examine and interpret the problem log to help you isolate the problem.

If no problem analysis exists for a particular type of error, no entry will be written to the problem log. However, all errors are recorded in the error log. The error log entry may help your service personnel isolate the problem. The format of the error log entries is explained in the APPC, APPN, and HPR articles in the *AS/400e Information Center*.

Configuring SNA/APPN and ANYNET

If you have an existing SNA/APPN configuration and are also configuring ANYNET, you cannot use the control point name from a remote AS/400's network attributes for the ANY parameter (RMTCPNAME, RMTLOCNAME). You may encounter connectivity problems.

SNA applications, such as pass-through, may receive the following messages:

- Message CPF8933 with RC80140001: route to specified location not found
- Users may also see SRCB6007101 msg-state 4060 f/#1mtask: The request for a route has failed. ROUTE_NOT_AVAILABLE condition has been detected. No destination NN or Virtual nodes available for intermediate routing.

To solve this problem use the Display APPN Information (DSPAPPNINF) command. This will show valid routing information. If you require a *LENNODE type connection, like ANYNET, between two APPN network nodes, configure the RMTCPNAME in the ANYNET controller description as something other than the actual control point name on the adjacent system.

Configuring an ANYNET (*LENNODE) controller and using the actual control point name may cause a continuous topology update problem. If a problem is occurring, the control point presentation services (CPPS) or the topology routing services (TRS) task continually shows a high CPU utilization (greater than 30%). When the number of network nodes and improperly configured *LENNODES increases in a network, the problem becomes more severe.

For more information on network problem analysis, see the APPC, APPN, and HPR articles in the *AS/400e Information Center*.

Chapter 4. Handling Communications Errors

This chapter discusses the following topics:

- The types of error conditions that potentially are encountered when the AS/400 system is communicating with remote systems or controllers
- The hierarchy of AS/400 functions that are provided to detect communications error conditions
- How communications errors are reported to the operator or application program, and how the user recovers from them

In many situations, you can set up the system to automatically attempt recovery of communications to remote systems and controllers, so that operator intervention is not required if recovery attempts are successful.

System configuration and management affect system performance during communications error recovery, and also are covered in this chapter.

Concepts

The following topics play a role in the performance and behavior of the system when a communications error occurs.

Configuration Considerations

When discussing the subject of error recovery, it is important to understand that the process of error recovery actually includes three steps. These include:

1. Detection and reporting of the error condition
2. Disconnecting all active users and recovering from the error condition
3. Connecting back to the system to get all users online

To facilitate this process, there are several communication configuration options to consider. These options and considerations will affect system performance during error recovery.

Note: These same considerations apply when many users connect or disconnect from the system at any point in time.

General Configuration Considerations

ONLINE (*YES or *NO): The ONLINE parameter specifies whether configuration objects are varied on online, at IPL. The ONLINE parameter is on the CHGLINxxx, CRTLINxxx, CHGCTLxxx, CRTCTLxxx, CHGDEVxxx, CRTDEVxxx, and commands.

During the IPL step that is indicated by SRC C900 2B40, the console controller and device descriptions, QCTL and QCONSOLE, are varied off. Their counterpart controller and device descriptions are varied on. Usually, these descriptions are named CTL01 and DSP01. There is one exception to this processing. If the IPL type system value QIPLTYPE is ’2’ (Attended IPL, console in debug mode), QCTL and QCONSOLE are not varied off. In addition, when QIPLTYPE is ’2’ an A900 2000 SRC is posted at the end of the IPL. Also, during the C900 2B40 IPL step, processing to vary on configuration objects is performed as follows:

1. Network server descriptions and their attached lines are varied on in the QSYSCOMM1 system job.

2. QSYSARB, QLUS, and communication arbiter (QCMNARBxx) system jobs perform other ONLINE(*YES) processing.
3. In QSYSARB, the objects vary on in hierarchical order. Network interface descriptions vary on first, followed by line descriptions, controller descriptions, and those devices that are managed by the QSYSARB job.

   Note: The communication arbiters vary on those devices descriptions that are assigned to them.
4. APPC devices are varied on under the QLUS system job.
5. Within each object type, the objects are varied on in alphabetical order. The name of an object determines when it is varied on during IPL.

By default, the communications configuration objects are created with ONLINE(*YES). As a result, the objects are varied on at IPL time. This may affect performance on other processes that run during the IPL.

Note: New configuration object types, such as PPP lines, have the ONLINE parameter default set to *NO on the create commands (for example, CRTLINPPP).

To optimize performance, ONLINE(*YES) configuration should be used with discretion. Consider doing the following:

- Limit the objects that vary on during IPL with ONLINE(*YES) to those that are critical to become available for general system use. Objects that may be considered critical are tape drives, CD-ROM drives, and selected local workstations.
- Place critical users in a subsystem group, and vary on configuration objects for this group using ONLINE(*YES). This allows critical users to get back online sooner.
- For noncritical users, vary on configuration objects at a later point by specifying ONLINE(*NO). Use a CL program or change the system startup program to manage vary on of the remaining configuration objects.
- For controllers on LANs, use the auto-configuration parameter AUTOCRTCTL(*YES) on the appropriate LAN line descriptions, and let the system vary them on when they are needed.
- Whenever possible, avoid varying on any configurations that would fail in attempts to connect to remote system. For example, avoid varying on controllers with a link type of *LAN that have an initial connection of *DIAL, when remote systems are not available. PCs on LANs typically do not respond to connection attempts. In this instance, attempts to connect to unavailable systems result in unnecessary communications recovery.

Automatically created virtual display and printer devices now set the ONLINE parameter to *NO. These devices are automatically varied on before being used. This saves time at IPL, but adds a little extra time when the virtual device is used for the first time.

Avoid varying on network server descriptions during the IPL. When network server descriptions are varied on, the integrated PC server is reset. This reset process can take a long time. During this time, the job performing the vary on is unavailable to do work. At IPL time, this job is QSYSCOMM1. When network server descriptions are varied on at IPL, QSYSCOMM1 is unavailable for other work.

System Value Considerations

The following system values can affect system performance during communications error recovery. Each of the following system values also is covered in the *Work Management* book.

QCMNARB

Communication arbiters. This system value specifies the number of communication arbiter system jobs that are available to process work for communications. The QCMNARB system value supports the following values:

- ***CALC, 0-99**

Communications work that previously was done in QSYSARB has been moved out of QSYSARB, and into the communication arbiter jobs. In addition, the auto configuration function that previously was performed in QLUS is now performed in the communication arbiter jobs. The communication arbiter jobs start at system IPL. The QCMNARB system value determines how many communication arbiter jobs start. The communication arbiter jobs are named QCMNARBxx, where xx can be 01 to 99.

If this system value is zero, QSYSARB and QLUS perform the work, not the communication arbiters. Setting this value to zero is not recommended.

*CALC is the default value for the QCMNARB system value. The system will determine the number of communications arbiter jobs based on your hardware configuration.

The setting of this system value can affect the performance of start-up, take down, and error recovery for communications. If there is an excessive amount of these system activities, having more QCMNARB jobs may result in improved communications performance.

During an IPL, controllers are assigned to a communication arbiter job. The system attempts to assign controllers to the communication arbiter job that is serving the least number of objects (combined controllers and devices). This balances the amount of work each communication arbiter job must perform. All devices that are attached to a controller are serviced by the communication arbiter job to which the controller is assigned. Work that was previously performed in QSYSARB for these controllers and devices is now performed in the communication arbiter job.

To determine which communication arbiter is assigned to a particular APPC controller description, use the Display Controller Description (DSPCTLD) command. To determine which communication arbiter is assigned for a device, use the Display Device Description (DSPDEVD) command.

Note: Certain types of devices such as optical devices do not attach to a controller. These devices are assigned to the QSYSARB job. Tape devices which are attached to controllers are also assigned to the QSYSARB job.

QCFGMSGQ

Configuration message queue. This system value allows you to specify the default message queue the system will use when sending messages for lines, controllers, and devices. A change to this system value takes effect when a line, controller, or device description is varied on. If this system value changes after

| a line, controller, or device description has been varied on, the configuration will
| need to be varied off, and back on, in order to use the updated system value.

QCMNRCYLMT

Communications recovery limit. The QCMNRCYLMT system value specifies for device descriptions the number of recovery attempts that are made. It also specifies when an inquiry message is sent to the system operator when the specified number of recovery attempts have been reached. If the CMNRCYLMT parameter value is specified as *SYSVAL for a network interface description, a line description, or a controller description, then the QCMNRCYLMT system value also is used for those descriptions. See “Automatic Communication Recovery” on page 126 for additional information.

Automatic Communications Error Recovery

Automatic communications error recovery is controlled by the QCMNRCYLMT system value or the CMNRCYLMT parameter on the configuration object. The CMNRCYLMT parameter is on the CHGLINxxx, CRTLINxxx, CHGCTLxxx, CRTCTLxxx, CHGNWIxxx, and CRTNWIxxx commands.

The recovery limits specify the time limit and intervals in which the system will perform automatic communications error recovery.

There are configurations in which the most appropriate option is no automatic recovery. An example of this is a PC LAN. If the connection to the PC is lost, the AS/400 will not be able to reconnect, since the PC generally must initiate the connection. In this case, setting the recovery limits to specify no recovery can result in better performance of the system when outages occur. However, this option has its disadvantages. If automatic communications error recovery is not used, manual recovery is necessary. This requires operator intervention. A practical option is to set the automatic recovery limits to one retry.

Recovery Commands

The system provides a set of commands to control automatic communications error recovery.

There is a set of commands that end automatic error recovery procedures. If any type of failure occurs after this command is run, an inquiry message is sent to the system operator. The end recovery commands are:

- ENDNWIRCY — End Network Interface Recovery
- ENDLINRCY — End Line Recovery
- ENDCTLRCY — End Controller Recovery
- ENDDEVRCY — End Device Recovery

There is a set of commands that resume automatic error recovery procedures. The resume recovery commands allow you to resume automatic error recovery procedures after they have been stopped. The resume recovery commands are:

- RSMNWIRCY — Resume Network Interface Recovery
- RSMLINRCY — Resume Line Recovery
- RSMCTLRCY — Resume Controller Recovery
- RSMDEVRCY — Resume Device Recovery

See “Communications Error Recovery” on page 121, for more information on error recovery limits.

QDEVRCYACN

Device Recovery Action. See “Device Recovery Action (DEVRCYACN)” on page 117, and the *Work Management* book for additional information.

QPASTHRSVR

Pass-through servers. The QPASTHRSVR system value controls how many pass-through server jobs are available for processing display station pass-through requests. These jobs are involved in the start-up and take down of pass-through sessions.

The default setting of this system value is calculated based on the hardware configuration of your system. Multiple pass-through server jobs may improve system reconnection performance in error recovery situations, or when many users connect or disconnect their 5250 display station pass-through sessions at the same time.

Setting the QPASTHRSVR value to 0 is not recommended. The QPASTHRSVR value of 0 is intended for use in migrating from the use of communications jobs for the 5250 target display station pass-through function to use the pass-through server jobs. The target pass-through communications jobs that are associated with the APPC device description are replaced by communications server jobs in the QSYSWRK subsystem. The use of communications server jobs eliminates the overhead that is associated with starting and ending the communications job. As a result, AS/400 system performance and user connect-time performance is improved when using the pass-through servers.

For more information on the pass-through servers, see the *Remote Work Station Support* book.

Network Attribute Considerations

The following network attributes can affect system performance during communications error recovery.

ALWVRTAPPN network attribute

The Allow Virtual APPN support network attribute is used to chose whether APPN devices should be attached to the real APPN controller, or to a virtual APPN controller. The default value is *NO, which means that virtual controllers will not be used.

One reason you may wish to use virtual APPN controllers is to limit the number of devices that go into error recovery when a failure occurs. When virtual APPN controllers are used, only the real APPN controller goes into error recovery. Since the devices are attached to the virtual APPN controller, they do not go through the traditional error recovery work flows. This can limit the amount of work the system must perform in an error recovery scenario.

If the Allow APPN Virtual Support or Allow HPR Transport Tower, network attributes are set to *YES. Subsequently, the system supports the automatic creation of APPN controllers and devices that are attached to them. To control the upper limit of automatically created APPC devices on APPN virtual controllers, use the Virtual Controllers Autocreate Device (VRTAUTODEV) network attribute.

For more information on the ALWVRTAPPN network attribute, see the APPC, APPN, and HPR articles in the *AS/400e Information Center*.

VRTAUTODEV network attribute

When ALWVRTAPPN(*YES) or ALWHPRTWR(*YES), is specified the VRTAUTODEV parameter indicates the maximum number of automatically created devices per virtual controller. Manually created devices may still be created if the VRTAUTODEV parameter value is less than the limit of 254. You can use the VRTAUTODEV network attribute to control the upper limit for the number of automatically created APPC devices on APPN virtual controllers. One reason for limiting the number of devices on APPN virtual controllers is performance. Each varied on virtual controller description is managed by a task on the system. Having multiple virtual controller descriptions allow for some parallel processing to occur. This cuts down on certain queue lengths. For example, the queue that is used to maintain the list of devices that are managed by the APPN virtual controller task. The default value for the VRTAUTODEV network attribute is 100. This means that for every 100 new APPN locations that your system communicates with, a new virtual APPN controller will automatically be created. See the APPC, APPN, and HPR articles in the *AS/400e Information Center* for more detail.

Line Configuration Considerations

The following line configuration options can affect system performance during communications error recovery.

AUTOCRTCTL(*NO, *YES): Automatic creation of APPC controllers typically does not happen during error recovery processing. However, the characteristics of these automatically created controllers, and how the system handles error recovery for them, can influence the performance of the system.

The system supports automatically creating APPC controller descriptions on LANs. Configuring the appropriate LAN lines (token-ring, Ethernet, FDDI, or wireless) with the autocreate controller (AUTOCRTCTL) parameter that is set to *YES accomplishes this. The AUTOCRTCTL parameter is on the CHGLINETH, CRTLINETH, CHGLINTRN, CRTLINTRN, CHGLINDDI, CRTLINDDI, CHGLINWLS, and CRTLINWLS commands. When the system automatically creates controller descriptions, these controller descriptions and attached devices are automatically deleted if not used within a specified amount of time. Automatic creation of APPC controller and device descriptions is performed in the communication arbiter jobs. When the system automatically creates controller descriptions, it uses the default values on the CRTCTLAPPC command. The exception is the ONLINE parameter, which is configured to *NO. APPC controller descriptions are automatically created with the following parameters:

- ONLINE(*NO)
- INLCNN(*DIAL)
- DIALINIT(*LINKTYPE)
- APPN(*YES)
- SWTDSC(*YES)
- MINSWTSTS(*VRYONPND)
- AUTODLTDEV(1440)

This chapter discusses all of the above parameters. Included is information on how they each play a role in error recovery. The default settings indicated above may not be desirable for your network. If that is the case, use a model controller description to override the default values.

Automatic creation and deletion of these configuration objects add some overhead to the system (noticeable in the QCMNARBxx jobs). If this is a problem, you can manually configure your configuration objects or choose not to have the system delete them for you. For more information on model controllers, see the *APPN Support* book.

AUTODLTCTL(1440, *NONE, or wait-time): Automatically created controllers and devices by default are automatically deleted by the system with the AUTODLTCTL parameter on the line description. The default time interval after which APPC controllers and devices are deleted if they have not been used is 1440 minutes (24 hours).

However, if the default time value is specified, there is a potential for controllers and devices to be automatically deleted over a weekend, for example. The resulting automatic creation of controllers and devices the following Monday morning may place unnecessary work loads on the system. Consider having the AUTODLTCTL parameter specify a value larger than 1440 minutes to cover weekends and holidays.

You may want the system to automatically delete controllers and devices if you use system-to-system APPN communications, and there are multiple routes through the network. In this instance, you can end up with multiple configuration objects since each route results in unique objects. You can clean up these objects by using automatic deletion. You could also use APPN virtual controllers to solve the problem of multiple device descriptions in APPN networks with multiple routes.

If the controllers and devices represent PCs on LANs, do not have the system delete and re-create these objects. Duplicate configuration objects will not be created. Limit the amount of work the system does for this environment by turning off autodeletion (or setting the value relatively high) for these configurations.

The AUTODLTCTL parameter is on the CHGLINETH, CRTLINETH, CHGLINTRN, CRTLINTRN, CHGLINDDI, CRTLINDDI, CHGLINWLS, and CRTLINWLS commands.

Link-level timers and retries: The configuration of link-level timers and retries can have a significant affect on network performance.

For complete information on link-level timers and retries, see the appropriate protocol-specific publication.

Link Types	Publication
Asynchronous	*Communications Management*, *Asynchronous Communications Programming*
Binary Synchronous	*Communications Management*, *BSC Equivalence Link Programming*
DDI Network	*LAN, Frame-Relay and ATM Support*
Ethernet Network	*LAN, Frame-Relay and ATM Support*
Frame-Relay Network	*LAN, Frame-Relay and ATM Support*
ATM Network	*LAN, Frame-Relay and ATM Support*
ISDN Data Link Control (IDLC)	*ISDN Support*
Integrated Services Digital Network (ISDN)	*ISDN Support*
Point-to-Point Protocol	*TCP/IP Configuration and Reference*
Synchronous Data Link Control (SDLC)	*Communications Management*
Token-ring Network	*LAN, Frame-Relay and ATM Support*
Wireless Network	*LAN, Frame-Relay and ATM Support*

Link Types	**Publication**
X.25	*X.25 Network Support*

ACTLANMGR (*YES or *NO): By default, the LAN manager function is set to *YES when a line is created for token ring. ACTLANMGR must be set to *YES when RSRCNAME(*NWID) is specified.

Specifying *NO disables the ring error monitor (REM) and configuration report server (CRS) functions. No error information is sent to the history log queue (QHST) or the message queue specified on the token ring line description (MSGQ parameter). Setting ACTLANMGR to *NO may increase network processing performance.

The ACTLANMGR parameter is on the CHGLINTRN and CRTLINTRN commands.

For more information on LAN Manager, see the *LAN, Frame-Relay and ATM Support* book.

Controller Configuration Considerations

The following controller and device configuration options can affect system performance during communications error recovery.

AUTODLTDEV (1440, *NO, or wait-time): Device descriptions that are automatically created by the system may also be automatically deleted by the system. The default is to delete devices that were automatically created after 1440 minutes (24 hours) if they have not been used in that time period.

Specifying the default has the potential side-effect of having device descriptions deleted over weekends. This can cause a system slow-down. For example, when users reconnect on Monday morning (after a 48-hour period of system inactivity), they may discover that they need to re-create their device descriptions.

It may be desirable to have the AUTODLTDEV parameter default to a value larger than 24 hours; 72 hours may be more appropriate to cover weekends. Use a model controller to change this value for any autocreated controller descriptions.

The default value for devices that are attached to automatically created APPN virtual controllers is 10,000 minutes.

The AUTODLTDEV parameter is on the CHGCTLAPPC, CRTCTLAPPC, CHGCTLHOST, and CRTCTLHOST commands.

INLCNN (*DIAL or *ANS): In error recovery scenarios, the actions that are taken to recover the controller depend on whether the controller description was created with *DIAL or *ANS. The default value for INLCNN is *DIAL. The INLCNN parameter is on the CHGCTLxxx and CRTCTLxxx commands.

Consider the following in creating the controller description:

- Use INLCNN(*DIAL) for AS/400-to-AS/400 connections when either system may initiate a connection with the other.

  Note: The time at which the system actually attempts to *DIAL depends on the setting of the APPN, DIALIMMED, MINSWTSTS, and CTLOWN parameters. See Table 6 on page 116 and Table 7 on page 116 for details on the interactions of these parameters.

- For PCs, use INLCNN(*ANS) to avoid unnecessary recovery attempts when the PC shuts down.

APPN Minimum Switched Status: The default value for the APPN minimum switched status (MINSWTSTS) parameter is *VRYONPND. Specify this to make APPN controllers in vary on pending status available for APPN route selection. Specify MINSWTSTS(*VRYON) to limit the routes that APPN recognizes as available. This prevents APPN from selecting routes that have a controller in varied on pending status on one system, but are varied off or inoperative on an adjacent system.

Note: MINSWTSTS(*VRYON) requires SWTDSC(*NO).

The setting of the MINSWTSTS parameter affects when the system will attempt to dial out to a remote system. Consequently, this affects how the AS/400 attempts to recover after a connection to a remote system is lost. The MINSWTSTS parameter can also play a role in system behavior when a configuration error has occurred. For example, if an incorrect remote NETID is configured, the value of MINSWTSTS determines the controllers that are used to attempt to locate a potentially nonexistent remote system.

SWTDSC (*YES or *NO): By default, the value for SWTDSC is *YES. This configuration is recommended for switched connections. It allows a switched line to be disconnected when the application is no longer using the line.

For PCs that are connected to a LAN, consider using SWTDSC(*NO). Connections between PCs that are running Client Access/400 and an AS/400 may be disconnected automatically if the following conditions exist:

- The Client Access/400 router is started
- An application such as a 5250 emulation session or a network drive is not running over the connection
- The application does not start within the time limit that is specified by the disconnect timer (DSCTMR) parameter

Specifying SWTDSC(*NO) keeps the connection to the AS/400 active, even if no applications are started.

The SWTDSC parameter is on the CHGCTLxxx and CRTCTLxxx commands.

Disconnect Timer (DSCTMR) parameter

The disconnect timer (DSCTMR) parameter controls the time before a connection without activity is dropped, or the amount of time to delay the automatic disconnection. The default value is 170 seconds. The range is 0–65535 seconds.

The DSCTMR parameter is on the CHGCTLxxx and CRTCTLxxx commands.

APPC Controller Recovery Summary: The action the system takes when APPC controller descriptions go into recovery depends on the settings of the parameters previously discussed. In addition, there is interaction between different parameters.

The tables that follow help describe when an AS/400 controller description on a LAN will attempt to reconnect to the remote system. The tables should be used to help you select the appropriate configuration parameters to best optimize system behavior when APPC controllers that represent PC clients go into error recovery.

In all cases in the following tables:

- The tables address APPC-attached PCs on a LAN that use Client Access/400.

  This often is the environment where the number of users is the largest. Other environments may result in different actions.
- The tables assume the following configuration values:

  DIALIMMED(*LINKTYPE)
  For LANs, this is *IMMED.

  Note: If DIALIMMED(*DELAY) is used, no dial attempts are made for the recovery or vary on cases.

  APPN CP sessions
  In all cases, there are no APPN CP sessions. APPN CP sessions will change when the system attempts to connect to the remote system.

Table 6. When does the AS/400 attempt to connect the remote system?

MINSWTSTS	**INLCNN**	**APPN**	**CTLOWN**	**Power PC Off (recovery)**	**Manual Vary On**
*VRYONPND	*DIAL	*YES	*SYS	Dial not attempted	Dial is attempted
*VRYONPND	*DIAL	*YES	*USER	Dial not attempted	Dial is attempted
*VRYONPND	*DIAL	*NO	*SYS	Configuration not allowed	
N/A	*DIAL	*NO	*USER	Dial not attempted	Dial is attempted
*VRYONPND	*ANS	*YES	*SYS	Dial not attempted	Dial not attempted
*VRYONPND	*ANS	*YES	*USER	Dial not attempted	Dial not attempted
*VRYONPND	*ANS	*NO	*SYS	Configuration not allowed	
N/A	*ANS	*NO	*USER	Dial not attempted	Dial not attempted

*Table 7. MINSWTSTS(*VRYON) affect on AS/400 attempts to connect to the remote system*

APPN	**INLCNN**	**CTLOWN**	**SWTDSC**	**Power PC Off (recovery)**	**Manual Vary On**
*YES	*DIAL	*SYS	*YES	Configuration not allowed	
*YES	*DIAL	*SYS	*NO	Dial is attempted	Dial is attempted
*YES	*DIAL	*USER	*YES	Configuration not allowed	
*YES	*DIAL	*USER	*NO	Dial is attempted	Dial is attempted

In all cases where a dial is attempted, and the remote system is a PC that uses Client Access/400, that dial attempt will fail with the following message:

- CPA57EF to QSYSOPR (Controller contact not successful)

Mode Considerations

Modes that are considered critical for correct system operations can no longer be deleted. QPCSUPP, QSERVER, and the mode description that are identified in the system's network attributes cannot be deleted.

If you have your own mode description, an accidental deletion can still occur. If the mode does get deleted, error messages in the configured message queue will indicate that the requested mode is not configured.

Job Considerations

When a line or controller failure occurs, and the application programs are notified, those jobs must often end and later must be started again after the communications resource is recovered. Job ending (particularly abnormal job ending) can be considered from a performance perspective as an extremely complex transaction. In a communications environment, many of these jobs may be ended at the same time. An example of this is a line failure on a LAN on which several PCs are attached. When the line is declared inoperative, all the jobs that are associated with all the PCs end at the same time. This causes a work peak within the system.

When the inoperative condition signals to the system, each job is notified of the error. This occurs for the application programs currently outstanding operations or next read/write operations. This occurs, for example, in a work station environment where an read/write operation is generally outstanding (a display is shown on the device waiting for user input). This indicates that several application programs have their input or output operation completed at the same time. They are now ready to run again, while peaking the activity levels of their storage pool. Because ending a job is expensive, these jobs may stay at peak-active levels until the end of a time interval. These peak activity levels may cause degraded performance of jobs seemingly not associated with the error handling situation, but that are sharing the same pool. It is important to ensure that your application programs are detecting the error and are not repeating I/O operations. Repeated I/O operations contribute to degraded performance.

You can use exception routines to help end the job as smoothly as possible. You may want to monitor for failure messages and then call the SIGNOFF command without requesting that a job log be written.

Device Recovery Action (DEVRCYACN)

Specifies the recovery action to take for the job when an I/O error is encountered on the *REQUESTER device for interactive jobs. This attribute is ignored on non-interactive jobs. This also is a system value (QDEVRCYACN). For more information on the system value, see the *Work Management* book.

The DEVRCYACN parameter has the following values:

***SYSVAL (default)**
The system value QDEVRCYACN is used. The system value can be any of the following values. The default system value is *MSG.

***MSG** The application programs monitor messages and uses information from the major or minor return codes to provide error handling routines.

Note: If you disconnect from the AS/400 by ending your session abnormally (such as powering off your terminal), you may create a security concern. If you end your session abnormally, another user could possibly attach to the system by using the same device description. That user could acquire the job you were running when you ended your session.

If you specify *ENDJOB or *ENDJOBNOLIST for the DEVRCYACN parameter, your application ends the job when the error is received. Also, the system sign on is displayed to the next user. This eliminates the possibility of the next user that acquires an abnormally ended job using the same device description.

***DSCMSG**
The job is disconnected. When the job is reconnected, the error message that is sent to the application program indicates that the device was lost and recovered since the last input/output operation.

To use this value, existing applications have to be re-coded to recognize this type of error, and to differentiate between this error and other input/output errors.

***DSCENDRQS**
The job is disconnected. When the job is reconnected, the request is canceled and control of the job is returned to the previous request level.

***ENDJOB**
The job is ended and a job log is produced. To minimize the performance affect of the ending job, the job's priority will be lowered by 10, time slice will be set to 100 milliseconds, and the purge attribute will be set to YES. A message that indicates that the job ended due to a device error is sent to the job log and the QHST log.

If many jobs specify this value, excessive system resources could be required to end all the jobs.

***ENDJOBNOLIST**
The job is ended, and no job log is produced. A message that indicates that the job ended because of a device error is sent to the QHST log.

Joblog Generation

When an error condition occurs in which the active jobs are ended, consideration should be given as to whether joblogs should be generated. Producing joblogs can use considerable system resources, especially during error recovery situations in which many jobs end at one time. In this case, it may be better to not generate joblogs.

You can specify that job logs are not generated using the following:

- DEVRCYACN of *ENDJOBNOLIST

  Note that there also is a QDEVRCYACN system value for ease in configuration.
- Change the job description (or the job itself through an initial program for the user profile) to LOGLVL(4 0 *NOLIST)

  Joblogs are not generated if jobs end normally, but are generated if jobs end abnormally.

Prestart Jobs

Using prestart jobs greatly reduces startup time. Jobs are reused rather than started and ended, and after an error, users are able to reconnect more quickly.

Prestart jobs are most helpful for short transactions. For instance, if a transaction will take 1/2 second, the overhead (time and resources) to start a job is a very large percentage of the work. If a transaction takes 60 seconds, the overhead of starting a job is much less significant.

Prestart jobs are used for many of the server functions the system provides, including:

- File Server
- Central Server
- APPC Signon Server
- Original License Management Server

Using prestart jobs for user applications requires some minor modification in how the target program is coded. Specifically, it must be coded to have a loop that accepts and processes the incoming program start request. The job then should continue the loop to process a new request, or if an error occurred, end the job so error information can be gathered.

Prestart job entries are shipped with the system for QCMN, QBASE, and QSERVER for these server jobs. You may want to change the prestart job entries, based on the usage of your system and the servers. How the prestart job entries are configured depends on what is important for your system and for your users. Consider the following:

- STRJOBS(*YES and *NO)
- INLJOBS
- THRESHOLD
- ADLJOBS
- MAXJOBS

For example, you may prefer configuring a larger number of available jobs (INLJOBS) under the following conditions:

- Many users connect to the system
- You want the connect processing performed as quickly as possible

However use care so that jobs are not unnecessarily started and ended without being used. This can occur if the THRESHOLD value is slightly below the total number of active users, and ADLJOBS is more jobs than ever, is used.

As user applications are developed, consider using prestart jobs to reduce program start request startup processing. For more information on prestart jobs, see the *APPC Programming* and *Work Management* books. For more information on prestart job entry for Client Access, see the *Client Access Express for Windows Host Servers* book.

Using the CHGSYSJOB Command

The CHGSYSJOB command allows users to change the run priority of a system job. The following are system jobs of interest for communications recovery:

- QCMNARB01 through QCMNARB99
- QSYSCOMM1

In general, these system jobs should be allowed to run at their system-provided priority, which is the default. However, if one of these jobs begins using large amounts of CPU, and affects other work on the system, it is possible to lower its priority.

Note: This may result in queued-up work for that job.

Device Wait Timeout

The device wait (DEVWAIT) timeout is used to limit the amount to time a subsystem takes for workstation input/output to complete.

File Wait Time

The file wait time (WAITFILE) parameter is the time the program waits for file resources to be available at file open time.

System Tuning

The overall performance tuning of the system can play a significant role during error recovery scenarios. For example, a machine pool that is too small can result in excessive error recovery times.

Most first-level error recovery procedures are performed in the communications controller. However, when the inoperative condition is signaled to the system, a significant amount of error handling code first must be paged into the machine pool. Because the error handling code is not normally resident in the machine, acquiring the code causes a peak in machine-pool paging. If this situation occurs frequently, you may want to increase the machine-pool size so that the effect on other machine-pool work is decreased.

Performance Adjustment—QPFRADJ

The automatic performance adjustment function of the system is set to ’2’ when the system is shipped. By using this value, the system can automatically adjust the performance of the system. This may be a desirable feature, particularly when unexpected loads hit the system. Automatic performance adjustment can help the system perform better through these peak loads.

You can consider moving performance-sensitive jobs or jobs that run in especially error-prone environments to separate pools or subsystems. Such movement reduces the performance degradation of seemingly unaffected jobs. Creating additional pools, however, may cost in overall system throughput. Pools essentially are logical boundaries within main storage that divide the storage into parts, and that assign those parts to particular jobs, preventing global sharing. See the *Work Management* book for additional information and considerations.

Subsystem Considerations

You should consider configuring communications users (whether they are remote workstation users or APPC communications users) into multiple subsystems. If a communications failure occurs, all users that run in a single subsystem may be affected as a result of the communications recovery that is performed in the subsystem.

To provide better system performance, subsystems should be configured so that approximately 200-250 users run in one subsystem. Users who are related in their configuration should be placed in the same subsystem. For example, if a single communications controller (APPC or a remote workstation) has many users, all those users could be configured to run in one subsystem set (QINTER and QCMN). Other users would be assigned to run in other subsystems. If the communications controller was to fail, it would limit the affect of the communications recovery to only those users in the subsystem set.

Problem Logging Considerations

The system problem log displays a list of all the problems that have been recorded on the system.

- If the system job QSYSCOMM1 becomes extremely busy, is consuming excessive CPU, and is affecting the overall system performance, the priority of the job can be changed using the CHGSYSJOB command.
- If the system is logging many duplicate problem logs for the same error, the problem logging can be temporarily turned off with the CHGMSGD command, specifying LOGPRB(*NO) for the repetitive error message.

  Once the cause of the problem logs has been identified and corrected, the message should be changed back to LOGPRB(*YES), to prevent loss of data if new or different problems occur.
- The system can detect when many identical errors are being logged in the problem log. When this occurs, the system may filter out some of the duplicate errors, depending on where in the system the messages originated. The user still may see many messages logged, but the number of duplicate messages will be reduced.

Communications Error Recovery

Communications errors are classified to help you select the correct system and operator actions necessary to recover from the error. In general, the AS/400 system will automatically recover from the different types of errors as described in this chapter. The error classifications are:

- Class 1 (data transmission integrity errors)

  These errors are detected by the communications data link control (DLC) protocols. They include:

  – Received frames that are corrupted by line noise
  – Received frames without error but received out of order because of previously corrupted frames
  – Discarded frames because of temporary resource restrictions
  – Physical interface errors with the data circuit-terminating equipment (DCE)

  These errors may occur during normal user or system data transfer, or when communications is started with a remote system, and the systems or controllers are attempting to contact each other.
- Class 2 (errors that leave the affected network interface, line, controller, or device in a PENDING status)

  The operations that result in these errors cannot be retried because they occur through normal use of the product. For example, during the use of the product, you can expect that a display that is attached to a remote controller would be turned off after the first contact. The system puts the device description in a PENDING status, waiting for the device to be turned on again. Another example is an automatic dial operation that results in a NO ANSWER condition. In this instance, the line returns to a PENDING status and is available for further use. This operation occurs without operator intervention.

If a switched connection is established, but contacting the remote station (from a protocol standpoint) is unsuccessful, the controller description is placed in VARY ON PENDING status. It is then ready to make another connection. These errors generally result in messages being sent to the system operator message queue (QSYSOPR) or the configured message queue, the history log (QHST), and the affected jobs. The AS/400 system, the remote system, or the remote controller may require configuration changes .

- Class 3 (errors that require a vary off and vary on sequence)

  You cannot use the affected network server, network interface, line, controller, or device until a vary off and vary on sequence is complete. This is a requirement to force system cleanup and start of communications functions again. For these errors, the status of the affected objects is FAILED. Such errors are generally exception conditions. These errors result in messages that are sent to QSYSOPR or the configured message queue, QHST, and the affected jobs.

- Class 4 (errors that are caused by an application program or device protocol errors)

  These kinds of errors normally do not cause QSYSOPR or the configured message queue, or QHST messages. In most cases, an OS/400 message is sent to the affected job. This type of error may be caused by 5250 data streams that are not valid, or by input data that is received when an output operation is attempted. Your program must anticipate these conditions by examining return codes.

  The *CL Programming* book discusses how to use the system debug functions to solve errors that are found in writing your application programs.

  Refer to the specific product manuals or the *ICF Programming* book for information on writing communications application programs.

Communications Error Recovery Problem Determination

When problems occur in communications, there are many sources for error message information and additional information to help resolve the problem. The following are the most common places to look for SNA Communications error information:

Message Queues

QSYSOPR
System operator message queue.

QSYSMSG
Optionally created message queue in QSYS (See the *CL Programming* book for additional information).

QHST
History log. More messages are sent to QHST than are sent to QSYSOPR or the configured message queue (See the *CL Programming* book for additional information).

For a thorough description of message queues, see “Managing Communication Messages” on page 76.

Joblogs

Many joblogs may contain information to help determine the cause of the problem. In addition, many joblogs contain informational messages that also can help you understand how the system has responded in regard to your communications functions. See the *CL Programming* and *Work Management* books for additional information. The following are the most common joblogs to consider:

QSYSARB
System arbiter. This joblog is for devices, and communications in general. It also contains ONLINE at IPL messages.

QSYSCOMM1
Communications and input/output system job. This joblog is for problem logging and LAN manager messages. It also contains ONLINE at IPL messages for network servers and their lines.

QCMNARB01 through QCMNARB99
Communications arbiters. These jobs contain ONLINE at IPL messages for the devices that are allocated to them.

QLUS
Logical unit services. This joblog is for APPC.

QLUR
LU6.2 resynchronization job. This joblog is for two-phase commit resynchronization processing.

QPASVRP
Target 5250 display station pass-through primary server program. This joblog is for target pass-through communication functions. See *Remote Work Station Support* for additional information.

Subsystem Jobs (QINTER and QCMN)
Interactive subsystem and communications subsystem. These joblogs are for subsystem jobs.

Other Logs
The following are other areas in which information is logged for SNA communications:

- Problem Activity Logs (within STRSST)
- LIC Logs (licensed internal code log, within STRSST)
- WRKPRB (work problem log)

CL Programming Considerations

The AS/400 system provides a wide range of system functions to assist you in providing an environment capable of functioning without an operator. The *Work Management* book provides information for setting up an unattended environment in addition to the books that are shown in “Communications Problem Analysis” on page 81.

First-level retries are performed by the system without an indication to any message queue unless the threshold parameter has been specified in the line description. If the first-level retries are exceeded, an inquiry message is sent to the QSYSOPR message queue or the configured message queue. If another message queue has been specified in the device type supporting the message queue parameter (CRTDEVPRT and CRTDEVAPPC), a message is not sent to QSYSOPR or the configured message queue.

Control language (CL) programs can be written to process inquiry messages when the receiving message queue is set to *BREAK delivery mode and a user-written program to handle break messages is specified. That program can perform any action that is determined by a particular application environment, such as one intended to be as independent of operator intervention as possible. Refer to the *CL Programming* book for special considerations when using break programs.

If you do not want to have programmed handling of messages, you can use the following two system functions to provide automatic inquiry message handling:

- The message queue is set to the default delivery mode. In this situation, the default response that is specified in the message description is used by the system.
- The system reply list is used. Inquiry messages are processed by the system reply list when the job sending these messages specifies INQMSGRPY(*SYSRPYL). The system spooled jobs that are specified in the job description are shipped specifying the use of the system reply list.

  The system reply list allows a specific response for a particular message or group of messages. The system compares values at offsets in the message text to take different actions for the same message identification. For example, you can take different actions, depending on device names, for messages to load forms.

The Retrieve Configuration Status (RTVCFGSTS) command can be used with a CL program to determine the status of a network server, network interface, line, controller, or device. While this command can be used for all status conditions, one of its primary uses is to automatically start jobs after a communications error has been corrected.

For example, assume that a host line failed and thus ended remote job entry (RJE) and 3270 printer device emulation operation. For off-hour operation, a CL program starts, which periodically (for example, every 10 minutes) monitors the session by using the RTVCFGSTS command for two devices, one for RJE, and one for 3270 device emulation. When the device status indicates VARIED ON, status code 30, or ACTIVE, status code 60, the program starts the RJE session and the 3270 device emulation printer job again by sending the appropriate Start RJE Session (STRRJESSN) and Start Printer Emulation (STRPRTEML) commands.

Handling Class 1 Errors

Class 1 errors are handled by the system at two distinct and interrelated levels. First-level error recovery generally is performed by the system. Error recovery procedures are based on the concept of trying the operation again and are generally a part of the data link protocols. The second level of communications error recovery is performed by the OS/400 licensed program. Second-level error recovery coordinates the operating system, the application program, and operator intervention.

The communications recovery limit plays a significant role in controlling second-level error recovery. The recovery limit is specified as follows:

- CMNRCYLMT parameter value for network interface descriptions, line descriptions, and controller descriptions
- QCMNRCYLMT system value for device descriptions

The two levels of error recovery can be thought of as nested loops. The **outer loop** controls the system’s attempts to again establish that communications with the remote end at the network interface, line, controller, or device level. The **inner loop** makes repeated attempts to communicate with or prepare the system to respond to communications attempts that are started by the remote system or controller.

The behavior of the inner and outer loops depends on:

- The particular error that is detected
- The various timer and retry parameter values (see the topic "Error Recovery Procedures Parameter Selection Flow Charts" on page 158 of the network interface description, line description, and controller description)
- The CMNRCYLMT parameter values of the network interface description, line description, and controller description
- QCMNRCYLMT system-wide value
- The operator response to inquiry messages
- Operator commands

First-Level Error Recovery

First-level recovery procedures are based on configuration parameters in the network interface description, line description, and the controller description. These first-level recovery procedures are performed automatically by the system when an attempt is made to contact a remote system or controller, or when data integrity errors are detected during communications.

See "Error Recovery Procedures Parameter Selection Flow Charts" on page 158 to select the first-level error recovery procedure parameters that are used by a specific protocol and configuration.

If the system carries out a significant number of retries, and the recovery is successful, degradation in your application performance is possible without error messages being sent to either the application program or the operator. This means that successful first-level error recovery is transparent to the application program. Although its performance characteristics may change, the application program is neither informed of nor involved in first-level error recovery (except for asynchronous communications). Messages are sent to QHST, QSYSOPR or the configured message queue, and the affected application programs only after the network interface, line, controller, or device has been declared inoperative. This occurs only after first-level recovery has been unsuccessful.

For IBM token-ring Network, Ethernet, SDLC, X.25, ISDN, DDI, frame-relay, and wireless communications, first-level recovery procedures do not use operating system resources. For binary synchronous communications (BSC), these procedures are performed cooperatively by the system and the communications controller. For asynchronous communications, error detection and recovery are a shared application program and system function. When errors occur that require first-level recovery procedures to be run, effective line use and throughput are degraded, and response time increases.

The system problem analysis process provides additional information on errors that have occurred and the probable causes of those errors. See the *Basic System Operation, Administration, and Problem Handling* for more information on this process.

The threshold process is a system function that helps in understanding the quality of the line and the behavior of the data link controls. See "Chapter 5.

Communications Threshold Process” on page 173 to understand how the system can be set up to report error conditions that are degrading line performance before the system declares the line or controller inoperative.

The system also keeps detailed statistics on data transmission integrity errors and protocol events for various lines and controllers. This information may be used to describe your environment so that you can set up the error recovery procedures to accommodate that environment. See the *Work Management* book for information about retrieving and analyzing this information.

Second-Level Error Recovery

Second-level error recovery is called when, to the degree to which the network interface or line was configured, transparent first-level error recovery was unsuccessful. For other errors, if first-level retries are not considered effective, immediate notification is made to the operating system so that second-level error recovery can begin. An example of this type of error is a lost-modem signal. For operations that can be tried again, the configuration values in the line description and controller description are used to specify the persistence of first-level error recovery, and to indicate when further retries are useless. If first-level retries are unsuccessful, or if the system cannot try the operation again, the application programs and the operator are informed. The second-level error recovery procedures generally involve the operating system, the application program, and the operator.

Second-level error recovery can be thought of as having two parallel paths in the system: a path for application program error recovery and a path for operator and operating system error recovery. One goal of second-level error recovery procedures is to automatically recover failed devices and enable the application program to automatically continue when the basic communication resource is recovered.

Automatic Communication Recovery

The communications recovery limit is controlled by the following:

- The CMNRCYLMT parameter value specifies the retry value for most network interface descriptions, line descriptions, and controller descriptions.
- The QCMNRCYLMT system value specifies the retry value for device descriptions. If the CMNRCYLMT parameter value is specified as *SYSVAL for a network interface description, a line description, or a controller description, then the QCMNRCYLMT system value is also used for that description.

The following examples discuss the CMNRCYLMT parameter.

The CMNRCYLMT parameter values for the network interface description, line description, and the controller description are used to control automatic second-level error recovery of the network interface, controller, and line descriptions. These parameter values contain the two following related numbers:

- The number of second-level recovery attempts automatically performed by the system (count limit)
- The length of time in which the specified number of second-level recoveries can occur (time interval)

The format is as follows:

```
CMNRCYLMT (x y)
```

where

```
x=count limit, 0 through 99 (default=2)

y=time interval, 0 through 120 (default=5)
```

or

```
CMNRCYLMT (*SYSVAL)
```

where

```
*SYSVAL=system value specified in QCMNRCYLMT
```

The count limit can be from 0 (no recovery is attempted) to 99. The time interval can be 0 (which represents zero time), or a value from 1 to 120 (minutes). A count limit of 0 and a time interval of more than 0 effectively disables automatic second-level error recovery. Turning off second-level recovery may cause the devices and controllers to go into recovery pending (RCYPND) state, and require operator intervention. A count limit of more than 0 and a time interval of 0 allows automatic second-level error recovery continuously. However, this is not recommended because of the system resources that may be used, and because performance may be affected.

The operator can control the parameter value CMNRCYLMT on an object basis for most network interfaces, lines, and controllers, to minimize the need for operator intervention because of excessive retries.

Note: For APPC controller descriptions that use TDLC lines, the CMNRCYLMT value has no meaning because the error recovery procedure always runs.

If the parameter value CMNRCYLMT is *SYSVAL, then the system values of QCMNRCYLMT are copied into the object, and used to control second-level error recovery until the next vary on sequence. If the network interface, line, or controller objects have other parameter values for the CMNRCYLMT, these values are used for each error recovery.

The device object, different from the line and controller objects, does not have a CMNRCYLMT parameter. Instead, when a device is varied on, the value of QCMNRCYLMT is copied into the object, and is used to control second-level error recovery for that object. If a device needs a different value for QCMNRCYLMT, vary that object on after changing the QCMNRCYLMT value.

When the first-level error recovery is not successful, the value of CMNRCYLMT or QCMNRCYLMT that are copied into the object is examined by OS/400. Depending on the values that are specified, attempts at second-level error recovery are made to contact the line, the remote controller, or the device.

For example, if the CMNRCYLMT parameter value of a line is specified as `'2 5'`, then two automatic recovery attempts can be made within a 5-minute time interval before the operator is notified to attempt manual error recovery. The following discussion shows how this operation works.

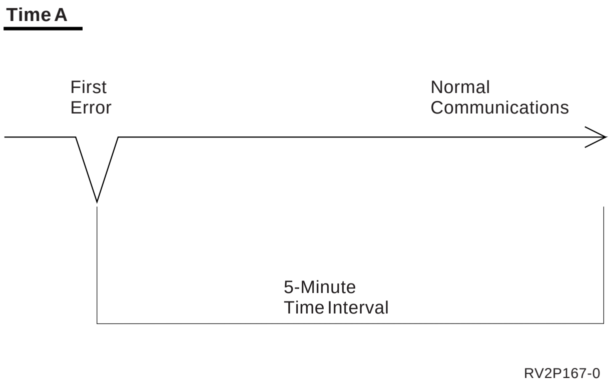


Figure 10. Error Recovery Example: Time Interval 1

In Figure 10, an error occurs at time A, which establishes the start of a 5-minute time interval. Now, the first attempt to try the operation again is also recorded.

Because the error occurred within the recovery limits, second-level error recovery is attempted. Assume that this recovery attempt is successful.

Note: The action that is taken by the system for a second-level recovery attempt is generally similar to those that occur when an object is first varied on. Recovery is successful when events occur similar to those that moved the object from VARY ON PENDING to VARIED ON and then to ACTIVE status. That is, second-level recovery is similar to a normal vary off that is followed by a vary on. The difference is that the application program does not necessarily have to be canceled and started again. See Chapter 2. Working with Communications Configurations and Status for more details on recovery status and system actions. There are separate counters for the number of times the operation has been tried again for network interface, line, controller, and device errors.

At time B, only two minutes later, another second-level error occurs. The system determines if this error exceeds the specified time interval that started at time A. (See Figure 11.)

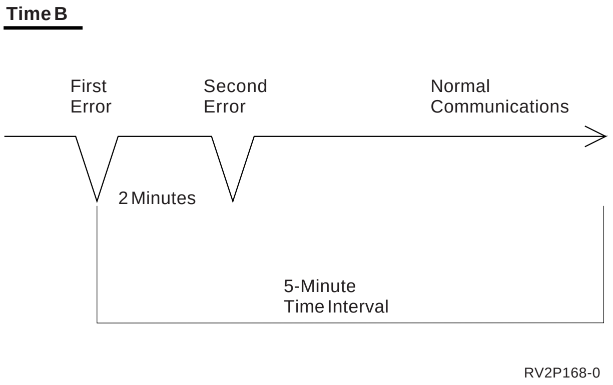


Figure 11. Error Recovery Example: Time Interval 2

In this example, neither the specified count limit (2) nor the time interval (5 minutes) is exceeded. Second-level error recovery is attempted. Assume that the recovery attempt is successful.

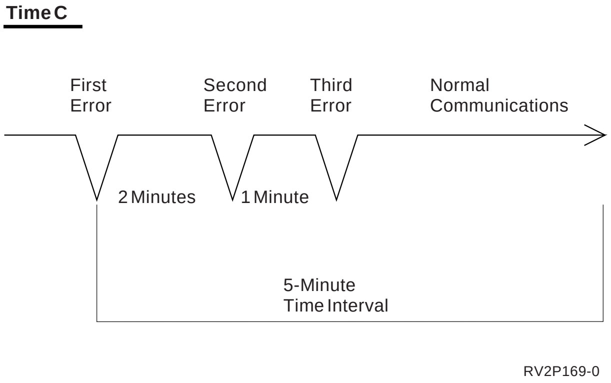


Figure 12. Error Recovery Example: Time Interval 3

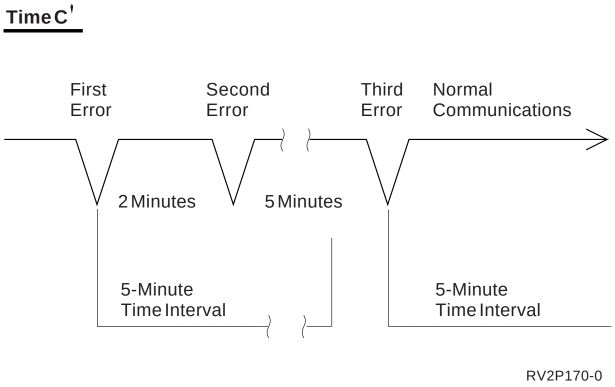


Figure 13. Error Recovery Example: Time Interval 4

At time C, only one minute after the second error, a third error occurs. (See Figure 12.)

Because the third error exceeds the specified recovery count limit (2), an inquiry message is sent to the operator. The operator must respond to the inquiry message.

Instead of the scenario in Figure 12, assume that the third second-level error occurs 5 minutes after the second error (as shown in Figure 13.) Because the third error exceeds the 5-minute time interval starting at time A, a new time interval starts, and second-level error recovery is attempted.

If more second-level retry attempts are required because of the number of irrecoverable first-level errors that are occurring, automatic recovery is ended and the operator is informed. System configuration can be changed to do error recovery indefinitely.

An example of this is setting the parameter values of CMNRCYLMT as follows:

```
count limit=1

time interval=0
```

No retries can be performed in zero time, so the second level of recovery may call on the first level of recovery indefinitely. Automatic second-level error recovery ends when successful or when the number of recovery attempts, occurring within the time interval, exceeds the count limit of the CMNRCYLMT parameter.

If first-level error recovery is not successful, a message is sent to QSYSOPR or the configured message queue that informs the operator of the failure. A message is also sent to QHST. If CMNRCYLMT specifies no second-level attempt to try the operation again, the message is an inquiry message with options G (Go), R (Retry), and C (Cancel). If CMNRCYLMT retries are not completed and this error can be automatically retried, the message to QSYSOPR or the configured message queue is informational: `Communications has failed and recovery is in progress`. The operation (an attempt to contact a network interface, line, remote controller, system, or device) is retried.

QHST receives a message for each following first-level failure. Only when the number of retries has exceeded the time interval is QSYSOPR or the configured message queue, informed again. The message is an inquiry. You can either let the operator reply to the message, or you can program an automatic reply by using the system reply list function. (For a description of the system reply list function, see the *CL Programming* book.) If the operator replies, the options are either G (Go), R (Retry), or C (Cancel). If G is specified, the operation range of time is reset, and up to the full number of retries are performed again. If R is specified, one more attempt is made. If C is specified, recovery ends, and any suspended file open operations complete with a failed return code. Later file open operations are rejected until recovery is started again.

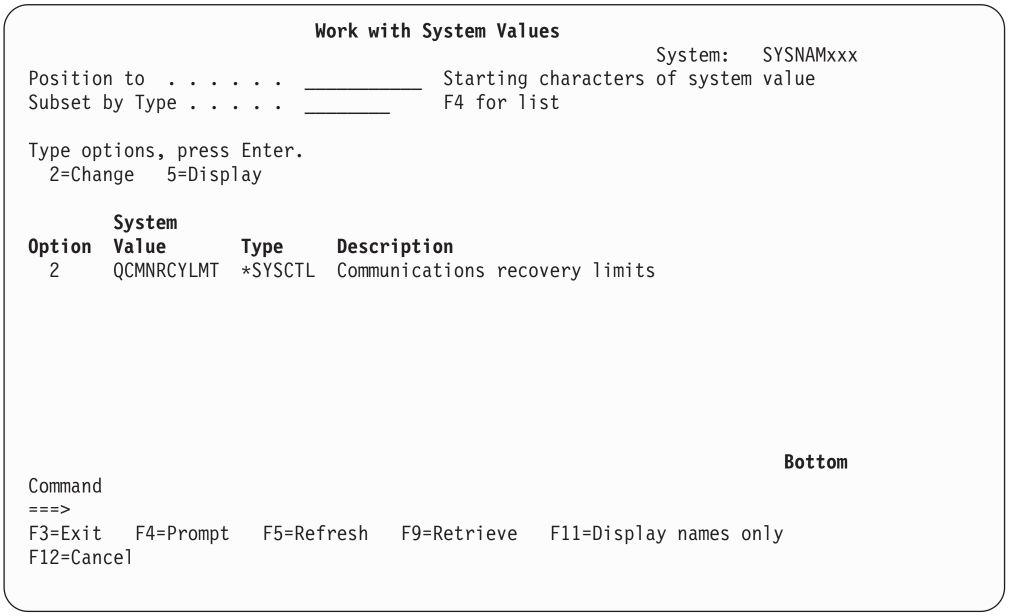
Note: If G or R is the response to *all* communications inquiry messages, functional loops can result; for example, a loose connection at a modem causes an error.

Changing QCMNRCYLMT System Value

The following displays show how to display QCMNRCYLMT values, change the values, and display the values again for verification.

The QCMNRCYLMT value is always copied into the device description. The QCMNRCYLMT is copied into the network interface description, the line description, and the controller description if CMNRCYLMT(*SYSVAL) is specified. After you change the QCMNRCYLMT value, the new values for the network interface, line, controller, or device take effect after the next vary on sequence.

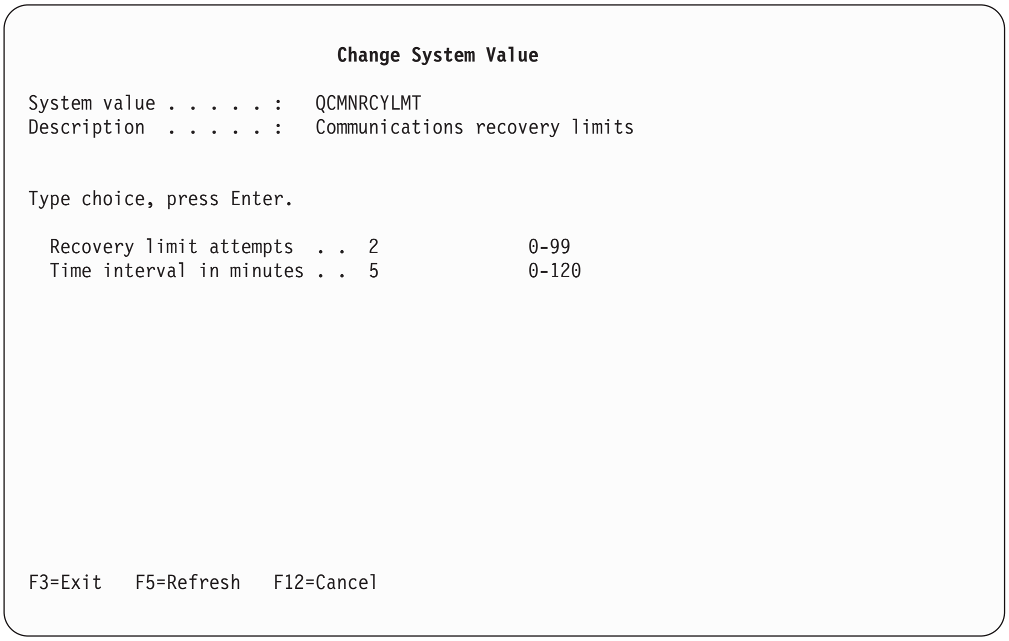
You can display and change the system values by entering `WRKSYSVAL QCMNRCYLMT` and pressing the Enter key. The following Work with System Values display shows the system value, type, and description:



```
                           Work with System Values
                                                           System:   SYSNAMxxx
Position to  . . . . . .  ___________  Starting characters of system value
Subset by Type . . . . .  ________     F4 for list

Type options, press Enter.
  2=Change   5=Display

        System
Option  Value       Type     Description
  2     QCMNRCYLMT  *SYSCTL  Communications recovery limits









                                                                     Bottom
Command
===>
F3=Exit   F4=Prompt   F5=Refresh   F9=Retrieve   F11=Display names only
F12=Cancel
```

From this display you can select options to display or change system values.

The following display allows you to change the system values:



```
                             Change System Value

System value . . . . . :   QCMNRCYLMT
Description  . . . . . :   Communications recovery limits


Type choice, press Enter.

  Recovery limit attempts  . .  2             0-99
  Time interval in minutes . .  5             0-120















F3=Exit   F5=Refresh   F12=Cancel
```

For detailed information on how to display and change system values, see the *Work Management* book.

Other Operator Second-Level Recovery Controls

The operator may decide that no amount of first- or second-level retries will resolve the problem. For example, the modems or the remote station may lose power, the line may be broken, or cables may be loose. Commands are provided to stop second-level recovery the next time the system is notified of the failure. These commands are End Network Interface Recovery (ENDNWIRCY), End Line Recovery (ENDLINRCY), End Controller Recovery (ENDCTLRCY), and End Device Recovery (ENDDEVRCY).

These commands control second-level recovery operations in the outer-loop only. They take effect the next time the network interface, device, controller, or line fails (makes a pass through the outer loop).

To end any undesired first-level recovery, you can vary off the associated network interface description, line description, controller description, or device description.

Note: Any jobs that are associated with the device description must end before the vary off operation can be done.

The commands to resume recovery when the problem has been corrected are Resume Network Interface Recovery (RSMNWIRCY), Resume Line Recovery (RSMLINRCY), Resume Controller Recovery (RSMCTLRCY), and Resume Device Recovery (RSMDEVRCY).

These commands can be used to restart communications recovery after second-level error recovery was ended using the ENDNWIRCY, ENDLINRCY, or ENDCTLRCY commands. Resume recovery commands are valid only when the communications object is in recovery pending status.

Automatic Communication Recovery—Examples

AS/400-to-Remote BSC System on a Nonswitched Point-to-Point Line

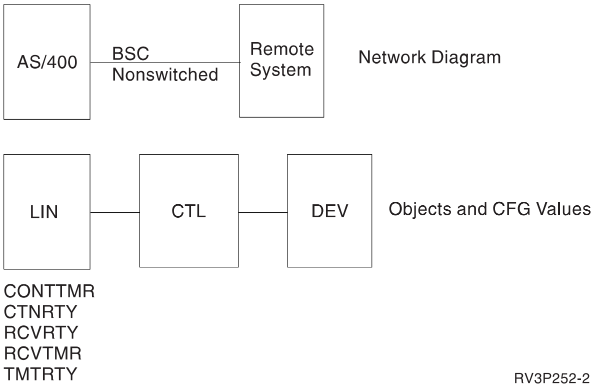


Figure 14. AS/400-to-Remote BSC System on a Nonswitched Point-to-Point Line

When the line description on the AS/400 system is varied on, the system begins to communicate with the modem. When the binary synchronous communications (BSC) controller description is varied on, contact is established even though, at this point, no data passed between the two systems. The operator sees a message that indicates that contact is made. When the device description is varied on, the communications line is open. The session becomes active when the local application program on the AS/400 system opens the file for input or output. At this point, data can flow from the local application program to the remote application program.

The BSC protocol ensures that the data is sent and received correctly. If a data block is sent to the remote end and the local system does not receive the appropriate response, first-level error recovery procedures are started. This generally means that the local system transmits the data again until successful or the user-configured retry limits have been exceeded. You can configure the persistence of the first-level error recovery procedures by the data state retry (DTASTTRTY) and the contention state retry (CTNRTY) parameters in the line description. If the default (7) is being used, the system tries 8 times (original transmission and 7 additional attempts) before declaring a session failure or a line inoperative. If the operation is successful, no notification is made to the operator or application program. A performance degradation may be noticed. The first-level error recovery procedures are designed to handle transient and temporary bursts of line noise. Noisy lines may require raising the data state retry (DTASTTRTY) value for those lines. The cost of raising this value is that, when permanent errors occur, it takes the system longer to detect loss of contact with the remote application program.

Other line description parameters that can be used when the first-level error recovery is being performed are:

- Continue timer (CONTTMR)
- Contention retry (CTNRTY)
- Receive retry (RCVRTY)
- Receive timer (RCVTMR)
- Transmit retry (TMTRTY)

Usually, the defaults are effective, but in certain environments the values may have to be adjusted (with the parameters that are listed above) to be more useful. See *Communications Configuration* book for more information on these parameters.

If the error is identified as a session failure only, the application program is notified. The application program at this point should close and again open the file and attempt another transmission, if the application program is monitoring for these errors.

Note: If the error is suspected to be in the hardware, the operator is notified. The operator can then decide if a hardware problem actually exists.

If the error is serious enough, the line may be declared inoperative, and a message is sent to QSYSOPR or the configured message queue, and QHST, informing the operator that the line is inoperative. The QCMNRCYLMT value or the operator can inform the system to attempt to keep the line active. The application programs are informed by return codes about any I/O operations in process or about the next I/O operation. The application programs should monitor for these situations. At this point, the application program can close and again open, and then attempt to retransmit the data.

AS/400 System to 5294 on an SDLC Nonswitched Point-to-Point Line

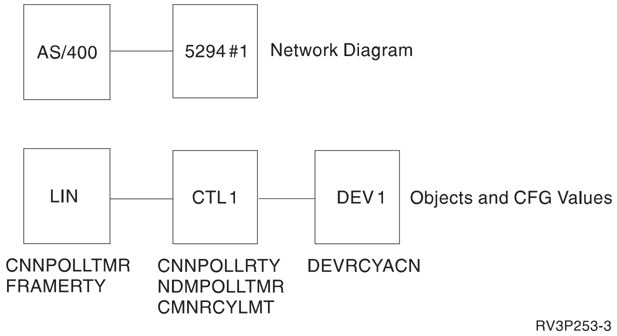


Figure 15. AS/400 System to 5294 on an SDLC Nonswitched Point-to-Point Line

In this example, the AS/400 system is the primary station, and the 5294 controller is the secondary station.

When the line description on the system is varied on, the system attempts to communicate with its modem. When the controller description is varied on, the system attempts to communicate with the remote controller. Communication is attempted by polling with an SDLC command that is called exchange ID (XID). The number of times this connection poll is sent (if no response is received) is controlled by the connect poll retry (CNNPOLLRTY) parameter in the controller description. The default of *CALC, in this case, means that retries are made indefinitely. This happens so that the system is ready to communicate when the remote secondary controller is. Following each poll, the system waits for the remote controller to respond. It will wait according to the amount of time specified by the connect poll timer (CNNPOLLTMR) parameter in the line description. The normal disconnect mode poll timer (NDMPOLLTMR) parameter in the controller description controls the interval between poll. The default of *CALC (only one controller on the line) means 0.5 seconds will be used between polls (repeatedly without an intervening wait). When the remote controller becomes available, it responds to the next valid poll with its XID, and the system goes on to contact the remote controller. The system shows a successful contact with the remote controller by posting a message to QSYSOPR or the configured message queue.

At this point, the system and the remote controller are considered to be in normal response mode, and information frames can be exchanged between the remote controller and the system. These frames may be system protocol frames or user data frames. SDLC protocol ensures frames are received correctly and in order.

If a frame is sent to the remote controller (while in the normal response mode) and the system does not receive the appropriate response, first-level error recovery procedures are called. This generally means the system attempts to send the data again until it is successful or until the user-specified retry limits are exceeded. You can specify the persistence of the first-level error recovery by the FRAMERTY parameter on the line description. If the default (7) is used, the system tries 8 times (original transmission and 7 additional attempts) before the controller is declared inoperative.

If the attempt is successful, no notification is made to either the operator or the application program (see “Chapter 5. Communications Threshold Process” on page 173 for exceptions to this statement). You may be aware of degraded performance and increases in response time. Threshold messages can show this condition when no first level retries are exceeded. This first-level error recovery is designed to handle transient and temporary bursts of line noise. Noisy lines may require raising the FRAMERTY value for those lines. The cost of raising this value is that when permanent errors occur (for example, when a line is cut, or when the remote controller loses power indefinitely), it takes the system longer to detect loss of contact with the remote controller. This may be longer than the time it takes for remote operators to call the system operator who still sees the network interface, line, controller, and device as ACTIVE during this first-level error recovery.

If first-level retries are exceeded, a message is sent to QSYSOPR or the configured message queue, and QHST informing the operator that the controller is inoperative. Now the controller and associated devices are in a RECOVERY PENDING status. When first-level retries are exceeded, CMNRCYLMT (QCMNRCYLMT if *SYSVAL is specified or the object is a device) values control second-level recovery.

If the CMNRCYLMT defaults of 2 5 are used (two or less errors occur in five minutes), and if values are not exceeded by this error, this message is informational and another attempt is made. Now the status goes to VARY ON PENDING.

If the error caused the CMNRCYLMT values to be exceeded, or if the count limit is set to 0, the controller and associated devices go to RECOVERY PENDING status. QSYSOPR or the configured message queue, is sent an inquiry message asking the operator to correct the situation. It also provides the system direction on what to do next. For example, if the remote controller loses power, or the line to the remote controller breaks, the operator can wait until the problem is corrected. Then it can attempt to contact the remote controller again by specifying an R or G response to the inquiry message. This response directs the system to contact the remote system or controller again. Contact is made like the first attempt, using the contact algorithms specified by the CNNPOLLRTY parameter being indefinite and NDMPOLLTMR parameter being 5 (0.5 seconds). These are the values that are used if *CALC is specified, and only one controller is on a nonswitched line. The system continues to automatically poll the remote controller until either the polls are answered or the operator varies off the controller description. During this polling, the controller description and the associated device descriptions are in the VARY ON PENDING status.

When the remote controller responds to the contact poll and later protocol frames are exchanged, the status goes to VARIED ON and the devices are again ready to communicate. Generally, this means the subsystem monitor recovers the sign-on displays to the devices.

When the controller becomes inoperative, the application programs are informed by feedback about any input/output operations they had outstanding or about their next input/output operation. If the application program sends its display files again and, in the meantime, recovery is successful, that input/output operation may complete successfully.

AS/400 System to Three 5394s on SDLC Nonswitched Multipoint Line

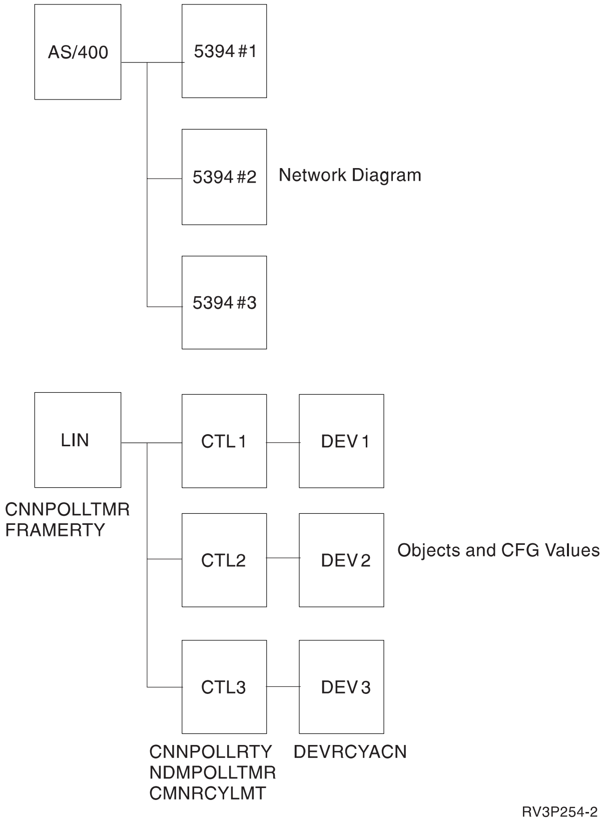


Figure 16. AS/400 System to Three 5394s on Nonswitched Multipoint Line

In this example, the AS/400 system is the primary and the three 5394 controllers each take on a secondary role. In this configuration, each remote controller has a unique address. Controller error recovery is very similar to the point-to-point example above. Differences are due to the fact that the line is being shared and the controllers must take turns being polled. The performance (response time) for devices on one controller are affected by traffic on other controllers. In an error condition, a polled controller may not respond and polls to following controllers must wait for time-out conditions. Here, the performance of one controller may degrade because of errors in data transmission when frames are sent to other controllers. See “Polling” on page 205 for the SDLC polling algorithm to assess how first-level error recovery may affect performance.

Line Failure after Successful Switched Connection Made by 5394 to AS/400 System

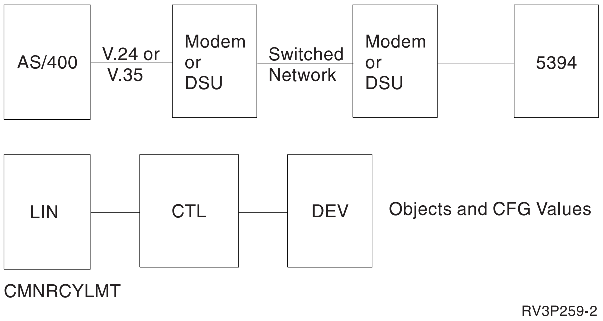


Figure 17. Line Failure after Successful Switched Connection Made by 5394 to AS/400 System

In this example, a switched connection was started and successfully established by a 5394 controller to an AS/400 system. Assume that the application programs are running. If a line or controller inoperative condition is reported to the OS/400 program (that is, first-level error recovery procedures were run, the retry limits were exceeded, and recovery was not successful), a message is sent to the QSYSOPR message queue or the configured message queue, and QHST. The QHST message is informational.

If the CMNRCYLMT retries are not exceeded, the QSYSOPR message is informational and a second-level recovery attempt is made. If the CMNRCYLMT retries are zero or the limits are exceeded, then the message is an inquiry message. A response of R or G causes a second-level recovery attempt. If the response is C, the error recovery is canceled and the controller description and the device descriptions change to a RECOVERY CANCELED status. The controller description and the device descriptions are not available if the connection is again attempted from the 5394 controller.

If second-level recovery is done, the recovery procedure run by the system is for the purpose of preparing for another connection. The controller description changes to a VARY ON PENDING status and it is no longer associated with the line description. The line description changes to a CONNECT PENDING status and becomes available for use.

The user jobs are informed by major and minor return codes about their currently outstanding or next input or output operation. Generally, the jobs should end, and the associated device descriptions return to a VARY ON PENDING status. Any open operations during this time are put on a queue until either of the following occurs:

- The connection is established again and the open operation is successful
- a C response is made to the recovery inquiry message, in which case, the open operation fails or the file's WAITFILE timer limit is exceeded

In this example, the AS/400 system recovery is complete when the line, controller, and device return to a VARY ON PENDING status. At this point, a switched connection can be started from the 5394 controller as originally done. If the connection is successfully established, for those devices whose jobs ended, the sign-on display is shown again by the controlling subsystem.

If the controller is in the RECOVERY PENDING status (an outstanding inquiry message is at QSYSOPR), or is in the RECOVERY CANCELED status (the operator or default reply responded with a C to the QSYSOPR inquiry message or the configured message queue), connection attempts by the 5394 controller are not successful because the controller description and associated device descriptions are considered to be in use. Also, by specifying CMNRCYLMT to nonzero retries (for example, value '2 5'), the AS/400 system recovery is automatic and requires no operator intervention (unless more than 2 such recoveries are required in the 5-minute time interval).

AS/400 System to 5494 on an SDLC Nonswitched Point-to-Point Line

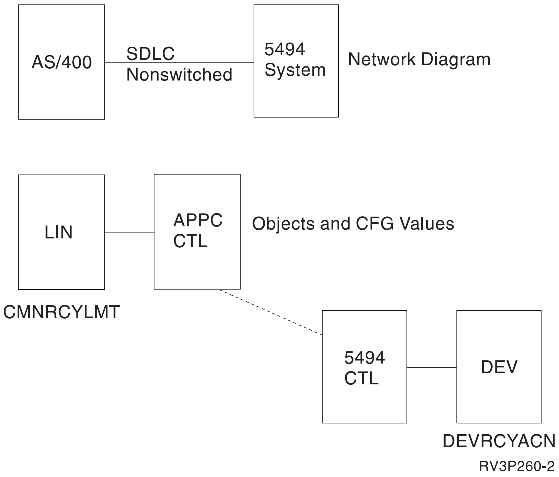


Figure 18. AS/400 System to 5494 on an SDLC Nonswitched Point-to-Point Line

The 5494 controller description is not attached to the line, but the 5494 controller description is associated with an APPC controller description attached to the line. The 5494 controller description is logically connected to the APPC controller description by an APPC conversation that is started by the 5494 remote controller. For each device attached to the 5494 controller description, the AS/400 system will start a parallel APPC conversation.

When errors occur on the line, the operation for the 5494 remote controller is similar to the example of the 5294 on an SDLC nonswitched point-to-point line. The exception is the APPC controller description goes through recovery and the APPC conversations receive errors. As a result of the APPC conversation errors, the device description receives a power-off notification. The 5494 controller description is disconnected. If the values of the CMNRCYLMT parameter are not yet exceeded, the system automatically attempts to recover the 5494 controller description. It goes to VARY ON PENDING status so it is ready when the APPC controller description is recovered.

If the error caused the CMNRCYLMT values to be exceeded, or the count limit is set to 0, the 5494 controller description and associated devices go to RECOVERY PENDING status. An inquiry message is sent to the QSYSOPR message queue or the configured message queue, asking the operator to correct the situation and to give the system directions. This is similar to the previous description for the 5294 controller. When the line is recovered, the 5494 remote controller reestablishes communications with the AS/400 system by starting an APPC conversation. The

| AS/400 system starts APPC conversations for all of the device descriptions
| attached to the 5494 controller description. The subsystem monitor recovers the
| sign-on displays to the devices.

When the 5494 controller and devices become inoperative, the application programs are informed by feedback about any outstanding input/output operations or about their next input/output operation. If the application program sends its display files again and, in the meantime, recovery is successful, that input/output operation may complete successfully.

You can specify the device recovery action for the job by using the DEVRCYACN parameter.

AS/400 System to 5494 on a Token-Ring Network

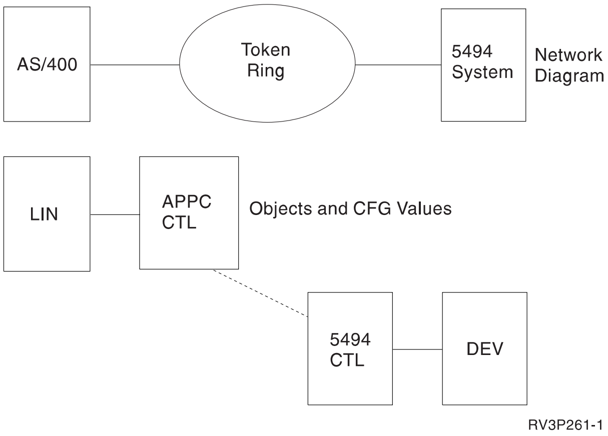


Figure 19. AS/400 System to 5494 on a token-ring Network

The 5494 remote controller description is not attached to the token-ring line description, but is associated with APPC controller descriptions attached to the line. The AS/400 system views the token-ring network as a logically switched network. That is, although the physical connection to the network is fixed (not physically switched), the nodes on a token-ring network are considered independent. Each node may contact one of the other nodes, based on the application program. These self-started connections are logically point-to-point and have a switched nature.

Similar to the 5494 nonswitched connection, the 5494 remote controller is responsible for starting the connection by polling the system. The system determines which description it should use to contact the 5494 remote controller. When the contact sequences are exchanged, the 5494 remote controller starts the APPC conversation and the AS/400 system starts APPC conversations for each device description attached to the 5494 device description.

The methods for detecting that the 5494 remote controller is offline are similar to the example for Figure 20 on page 140 for the APPC controller description. The 5494 controller description and its associated device descriptions are informed of errors the same way as the example of Figure 18 on page 138.

AS/400 System to Personal Computers on a Token-Ring Network

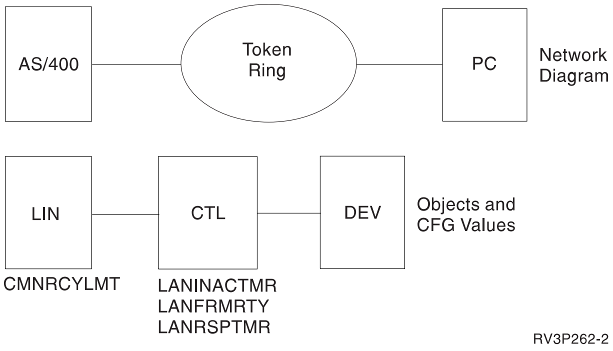


Figure 20. AS/400 System to Personal Computers on a token-ring Network

The AS/400 system views the token-ring network as a logically switched network. That is, although the physical connection to the network is fixed (not physically switched), the nodes on a token-ring network are considered independent and each may elect to contact one of the others, based on the application program. These self-started connections are logically point-to-point and have a *switched* nature.

In this example, the token-ring network is a network on which personal computers make contact with the AS/400 system to run programmable work station support. On the system side, the line description is either varied on or active, and the associated controller description is in the VARY ON PENDING status as is the device associated with that controller description. The personal computer starts the connection by polling with information the system can use to decide which of possibly several controller descriptions it should use to contact the personal computer. When the contact sequences are exchanged, user data can be sent.

Now assume that the user is running programmable work station support and that the personal computer is turned off or is being started again while in active session with the system. The system must first detect that the personal computer is no longer communicating. This is done by the logical link control (LLC) protocol in the AS/400 communications controller. The amount of time this takes depends on the current state of the last communications between the personal computer and the system, and the parameters in the associated controller description for that personal computer.

The following algorithm decides, in general, when the system detects that the personal computer is offline and declares the remote personal computer inoperative:

```
LANINACTMR + ((LANFRMRTY + 1) * LANRSPTMR)
```

Where:

LANINACTMR
specifies how long to wait to poll the personal computer since the last communications with the personal computer.

LANFRMRTY
specifies the number of times the poll is retried.

LANRSPTMR
specifies the time the system waits for a response to its poll from the personal computer.

For example, using the default of 10 seconds for the LANINACTMR parameter, and the value of 10 for the LANFRMRTY parameter at an interval of 1 second, it takes about 21 seconds to detect if the personal computer turned off or restarted.

At this point, the personal computer is declared inoperative and second-level error recovery takes over. Again, both the application program path and the operator or operating system path are used. The application program is informed on its next or currently outstanding input/output operation. In the programmable work station environment, AS/400 system jobs are APPC *target* jobs. Ending the remote work station function makes it appear to the system as if the personal computer emulating a work station is simply turned off. Any interactive application program is notified as if it is running to an AS/400 system local work station, and the power is turned off. The application program generally ends.

The operator or operating system error recovery path sees the inoperative condition signal, and places the controller and associated devices in RECOVERY PENDING status. If CMNRCYLMT retries are greater than 0, the controller and devices are automatically placed in a VARY ON PENDING status, waiting for the personal computer to start another connection. If the CMNRCYLMT retry value is set to 0, an inquiry message is sent to QSYSOPR or the configured message queue. The operator has to answer R to the message to put the controller description and associated device into VARY ON PENDING status. The system is now ready to receive another call from the personal computer.

If the personal computer starts a connection while the system is in the process of detecting that the personal computer was turned off, or was being restarted while in an active session, that connection attempt results in a message to QSYSOPR or the configured message queue: `Controller not varied on or not known to local system`. The message was sent because the correct controller description was *already in use* from the operating system’s standpoint. Also, the same message is sent to QSYSOPR or the configured message queue, if the correct controller description is in the RECOVERY PENDING status with an outstanding inquiry message to the operator.

For the above reasons, you may want to change the CMNRCYLMT parameter to have the recovery occur automatically and as soon as possible without operator intervention.

The following is an example of how the CMNRCYLMT time interval value works. If the CMNRCYLMT values are set to `'2 5'`, this specifies performing up to two second-level recovery attempts within a 5-minute time interval. In this personal computer configuration, the retry attempt is preparing the system to accept a connection started by the personal computer. The example described above is the first retry attempt. If the same personal computer was turned off, or was being started again while in an active session within 5 minutes of the first second-level retry attempt, the system would again automatically perform the operation that moved the controller description and associated device from RECOVERY PENDING to VARY ON PENDING status. It would now be ready to accept another call from the personal computer. However, if a third such start or loss of power occurred within 5 minutes of the first, instead of automatic system recovery, an inquiry message would result and operator intervention would be required to allow that personal computer to successfully connect again. As stated before, while the inquiry message is outstanding, later attempts result in a QSYSOPR message: `Controller not varied on or not known to local system`.

Continuing error detection and recovery operations represent a system resource burden. The CMNRCYLMT parameter value and QCMNRCYLMT system value allow you to distinguish between acceptable levels of automatic second-level error recovery and unacceptable levels, requiring operator analysis and intervention.

| Even if the controller description is created automatically by the system, the error recovery processing still works in the same way as it is described above. If the controller description is in a recovery pending status, or recovery cancel status, the AS/400 system will not automatically create a different controller description when the personal computer attempts to call the AS/400 system. For example, the following call attempts by the personal computer result in the following message in QSYSOPR, or the configured message queue: `Controller not varied on or not known to the local system`. The AS/400 system will not create a controller description automatically while the inquiry message is outstanding.

AS/400 System to System/370 Host on an SDLC Nonswitched Line

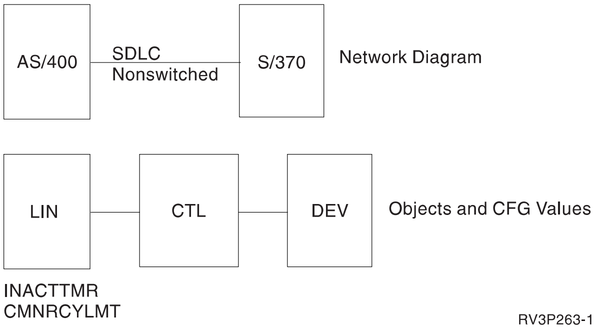


Figure 21. AS/400 System to System/370 Host on an SDLC Nonswitched Line

When communicating with a System/370 host, the AS/400 system is always the secondary system. A secondary role means that you can only send data when polled. Problems arise when the host stops polling. In these cases, RJE jobs do not make progress, work stations running 3270 emulation seem to stop, and AS/400 user programs stop communicating to the host. Cancellation of any of those jobs normally includes data exchanges with the host. Because the polling stops, those exchanges cannot be completed and time-outs occur. The cancellations seem to take longer than normal.

The line description parameter INACTTMR allows you to configure the system to distinguish between polls that are slow in coming and those that are not coming. The AS/400 communications controller SDLC records the time since the last poll. If INACTTMR amount of time elapses with no poll being received, the line is declared inoperative, the line description and associated controller and devices are placed in a RECOVERY PENDING status, and second-level recovery is called. Any application programs are notified of the error on their currently outstanding or next I/O operation.

RJE sessions must be started again when recontact with the host is made. 3270 emulation sessions are ended and the work stations are returned to the user. Program-to-program applications may end or again open their ICF files. The system coordinates with the operator or operating system path, suspending open operations until contact with the host is again made. See the *ICF Programming* and the appropriate language manuals for information on how to code exception handling in your application programs.

The CMNRCYLMT value can call for the operation to be tried again automatically or for the operator to reply with an R or G to the inquiry message that resulted because of the inoperative condition. A second-level attempt to try the operation again simply means that the line description is placed in a VARIED ON status. The controller and associated devices are in a VARY ON PENDING status waiting for the host to attempt to contact the system. No other recovery can be done by a secondary system.

Using the INACTTMR allows the operator to be aware when the host system has failed. It also starts the application program and device-level error recovery. Without this recovery, the AS/400 system would wait indefinitely to be polled. Jobs to the host appear stopped and canceling them is slow because communications with the host is generally required. If no jobs are running, varying off devices and controllers generally creates a time-out because these functions also require communications with the host system.

Notice that in a secondary environment, loss of contact on a nonswitched line only results in one pass through the outer loop that is controlled by CMNRCYLMT. That one pass resets the affected controller description and device descriptions back to the VARY ON PENDING status, waiting to be contacted by the host system.

The inactivity timer is started when the first frame from the host system is received. The response to that frame may be corrupted by the network and not received by the host system. In this case, the AS/400 system waits for another poll to resend the response. If the host system is *slow polling* the AS/400 system at an interval longer than the value specified by the inactivity timer (INACTTMR) parameter, the AS/400 system times out and declares the host system inoperative. Second-level retries (by either automatically using the CMNRCYLMT values or by the operator responding with an R or G to the inquiry message) reset the AS/400 system to accept the next poll and continue with the connection process. To avoid this situation, set the INACTTMR value to longer than the host slow-poll interval. The INACTTMR value can be increased (larger than the default value) on multipoint lines for a large number of secondary controllers.

Using the CMNRCYLMT parameter provides the quickest possible recovery with no operator intervention.

AS/400 System to System/370 Host Receiving Abnormal DACTLU

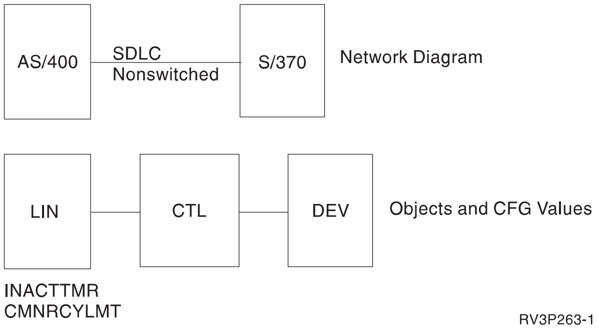


Figure 22. AS/400 System to System/370 Host Receiving Abnormal DACTLU

Sometimes, when communications between an AS/400 system and a System/370 host takes place over an SNA link, the host system sends a deactivate logical unit (DACTLU) request while it is in session with an AS/400 logical unit (LU) device.

This can happen for several reasons, such as when a particular VTAM LU is forced off by a Vary Inactive (V INACT) VTAM command.

If the DACTLU request is received while the LU device is in session with the host (for example, an SNA bind occurred), the DACTLU request is out of sequence because the AS/400 system expects an SNA unbind command first. In this case, the DACTLU request is considered to be abnormal.

In the case of an abnormal DACTLU request, the AS/400 system detects an error condition, but error recovery depends on the device type (SNA LU type). A description of the recovery procedures done on APPC, DHCF, and 3270 emulation displays when an abnormal DACTLU request is detected follows.

Recovery Procedures—Examples

Recovery Procedures for a DHCF Device

When a System/370 distributed host command facility (DHCF) work station is in session with the AS/400 system and an abnormal DACTLU request is sent by the host, the host system simply returns to the host session. Communications between the work station and the AS/400 system is ended. The job on the AS/400 system is notified that power has been lost on the DHCF device, and the job should end after doing any necessary cleanup. After the host sends the ACTLU again, the DHCF session with the system may again be established by the remote terminal user. The QCMNRCYLMT value is not used for abnormal DACTLU handling for DHCF devices. See *Remote Work Station Support* book for more information about DHCF recovery procedures.

Recovery Procedures for a 3270 Emulation Device

When the abnormal DACTLU is sent to the AS/400 system for the emulation device, the device does not stop emulation. There are two choices for the device user at this point:

- The user can either exit emulation through the normal exit procedure and return to the AS/400 session.
- The user can wait for the host to send another SNA ACTLU request and again establish the host session by the normal method.

QCMNRCYLMT is not used for handling DACTLU requests that are not normal that are sent to 3270 emulation devices.

Recovery Procedures for an APPC Device

When an APPC device receives an abnormal DACTLU request, second-level recovery procedures are called, using CMNRCYLMT values to decide the recovery action. If the CMNRCYLMT values are set to `'0 0'`, or the number of previous retries occurring within the time interval (both specified in the CMNRCYLMT value) is exceeded, then inquiry message CPA2601 is sent to the QSYSOPR message queue or the configured message queue. If an R or G response to the message is specified, then the device description is placed in a VARY ON PENDING status. This also occurs if processing the CMNRCYLMT parameter causes a recovery attempt. It is now ready to respond to the recovery procedure started by the host system.

If the message is outstanding when the host system sends an ACTLU command, the AS/400 system responds, but the device stays in the RECOVERY PENDING status until the message is answered. At this point, the APPC device goes to the ACTIVE status.

Note: A secondary system can send nothing to the host system to cause a recovery. The host system must start the recovery procedure.

During this time, if the AS/400 system is the source side of the pair of communicating programs, the LU device can be allocated again by a CPI communications allocate call or by an ICF acquire operation. The call operation is suspended until one of the following occurs:

- For ICF, the time specified in the WAITFILE parameter is exceeded.
- The host system sends an activate logical unit (ACTLU) request and the APPC session is established again.
- A *C* response is given to the inquiry message.
- The End Device Recovery (ENDDEVRCY) command is processed.

Using APPC, the source side can acquire the session and call the target application program again. The previously evoked target application program should do any necessary cleanup and then end the job. It is not possible for the source program to again establish communications with the same job on the target system. A new session must be started and the target program started again.

Considerations for Asynchronous Communications Error Recovery

Asynchronous (start—stop) communications error recovery is designed to pass as many errors as possible to the application program. No first-level error recovery is available except for line failures where the system-level recovery is similar to other communications protocols. All errors not classified as line errors are returned to the application program where recovery, if any, is to be performed. Errors returned to the application program include:

- Parity
- Framing (stop bit)
- Buffer overrun
- Data inactivity
- Break signal receipt
- Data lost

No line errors are retried by asynchronous support. When line errors occur, the system-level communications recovery limit causes attempts to establish communications again with the remote system. However, all data being processed at the time of the error is lost. No attempt to ensure data accuracy is provided by asynchronous communications support except when an application program is using file transfer support.

Line-level errors include signal failures, such as the dropping of:

- Data set ready (DSR)
- Carrier detect (CD)
- Clear to send (CTS)
- Ready to send (RTS)

and switched-connection failures. Asynchronous communications ignores all signal drop and rise warnings.

Considerations for X.25 ELLC Error Recovery for SNA Controllers

The use of extended logical link control (ELLC) protocol between two adjacent SNA stations causes the AS/400 system to perform additional error detection and recovery procedures. For certain classes of errors, this recovery occurs automatically without operator intervention as part of first-level error recovery.

Some examples of additional error detection and recovery are:

- End-to-end acknowledgment of logical link protocol data units (PDUs)

  Data lost or duplicated by internal network services are detected and recovered automatically.
- Validity checking on PDUs

  Data errors caused by internal network services are detected and recovered automatically.
- Virtual circuit assurance

  ELLC procedures define recoverable error conditions reported by the internal packet switched network services by CLEAR, RESET, and RESTART request packets. For those recoverable conditions, an attempt is made to automatically reestablish a virtual circuit between the two adjacent link stations. For switched virtual circuits (SVC), this action results in automatic recall or answer attempts.

  When a reconnection is in process, the AS/400 system reserves an SVC logical channel of the appropriate type. The reconnection may occur on a different logical channel than used for the original connection. An incoming call may be rejected if the only available SVC logical channel of the appropriate type is reserved for pending ELLC reconnection attempts.

This additional error detection and recovery reduces performance because the ELLC requires up to 6 bytes of header information per PDU, and longer delays are possible before reporting failures. In addition, enhanced logic link control (ELLC) causes more data packets to be sent (logic link control data acknowledgments are sent as data packets) and, therefore, communication costs may increase. Whether you choose ELLC or QLLC for SNA stations depends on the level of service provided by the underlying network service.

Considerations for Modem Error Recovery

If the PRE-DIAL DELAY is set to a value of zero (0) for a line that has a modem attached, and does a reset at data terminal ready (DTR) drop, the line may hang when attempting a vary on. As a result, DTR is momentarily raised, then dropped, when an AS/400 line is varied on. This causes a modem, for example, the IBM 7852-400, to go through a reset operation. In this situation, the modem fails to respond correctly when the line is varied on.

To alleviate this problem:

- Use a PRE-DIAL setting other than zero (use 1, 2, or the default of 6)
- Or, reconfigure the modem so that it does not reset at DTR drop

Handling Class 2 Errors

Class 2 errors are handled by the system in a way similar to class 1 errors at both error recovery levels. QCMNRCYLMT and CMNRCYLMT are not generally used to control second-level error recovery procedures. Class 2 errors each require specific handling.

Remote Work Station Loss of Power and Subsystem Recovery

If an allocated remote work station is turned off while in an active session, the remote controller notifies the system of this fact, and the system reports an error to the application program. Generally, this causes the user application programs to end. In addition, the system puts the device in a VARY ON PENDING status, waiting for the device to be turned on again. When the device is again turned on, the controller reports this condition to the system. If the work station device has been allocated to a subsystem, that subsystem attempts to perform a series of exchanges called sign-on processing. During this series of exchanges, the device status shows sign-on. If those exchanges are successful, the device eventually shows the sign-on display. In a communications environment, however, there can be delays. These delays can be because of retransmission on the line. Retransmission can slow both the requests (for example, a bind) sent to the remote controller and the expected response (for example, a bind response). Delays can occur in both point-to-point and multipoint configurations, but may occur more often in multipoint environments where frames for a particular controller may have to wait their turn.

Delays can also be caused by the buffering in the remote controller. This temporary buffering can be because of the controller servicing other devices attached to that controller. While the subsystem is performing sign-on processing, other subsystem requests are put on a queue. These requests can include other sign-ons, sign-offs, job cancellation requests, transferring to secondary jobs, and so on. Therefore, under no circumstances should a remote controller (or remote controller emulator) indefinitely queue SNA exchanges for which no operator action is required. It is recommended that you have separate subsystems for local and remote devices. This reduces the chance for delays caused by subsystem I/O processing in a remote communications environment.

The subsystem protects itself from long delays in sign-on processing by the DEVWAITTMR parameter in the controller description for work station controllers. This parameter specifies the length of time the subsystem waits for the responses to any requests it sends and for which no device operator action is required. If the value of the device wait timer is exceeded, the subsystem sends a message to the job log for the subsystem and then varies off the device. When the device is varied on again, the subsystem attempts sign-on processing again. If five repeated sign-on processing attempts result in subsystem time-out and in varying off the device, the subsystem no longer allocates the device and attempts sign-on processing again. The system protects itself from repeated error situations that may affect overall system performance. If the problem has been corrected, and sign-on processing is desired, either the original subsystem must be ended and restarted (which resets the retry counter), or the device needs to be allocated by another subsystem. This can be done by adding a work station entry in an inactive subsystem and then starting that subsystem.

Unsuccessful Automatic Dial on a Switched Line

Another example of a class 2 error is a call failure in an automatic dial environment. The controller description INLCNN(*DIAL), the line description AUTODIAL(*YES), and SWTCNN(*BOTH) or (*DIAL) set up an automatic dial environment. For this example, assume a non-X.21 circuit-switched network. Dial operations can start as the result of an application program opening an ICF file to a device associated with a controller description set up for dialing. The system examines the candidate list, which the SWTLINLST parameter specifies in the controller description. It then selects the first line available to make the call. A line must be in the CONNECT PENDING status, and not already in use, to be available. The system instructs the automatic dialing hardware to make a call to the number specified in the controller description. First-level dial retries can be configured by the parameters PREDIALDLY, REDIALDLY, and DIALRTY. These error recovery procedures are performed by the AS/400 communications controller. If the line is consistently busy or if there is no answer, the system reports the unsuccessful attempt at contacting the remote. Now, OS/400 second-level error recovery procedures take control. At this point, the line description is no longer associated with the controller description, placed in the CONNECT PENDING status, and is generally available again. The controller description is placed in the VARY ON PENDING status and is generally also available for use. Like the nonswitched environment, both an application program path, and an operator or operating system path are used. The operator is informed of the failure by an inquiry message to QSYSOPR or the configured message queue. The responses include C (Cancel) or R (Retry). If C is selected, the system then fails the application program's file open attempt and returns control to the application program.

The operating system does not use the CMNRCYLMT value to automatically attempt second-level redial attempts. If the system performs a second-level redial attempt, the user must answer an R to the inquiry message. The system reply list support or the default response to the QSYSOPR message or the configured message queue, may be used so that operator intervention is not required.

A second-level attempt includes all the processing that was performed on the first attempt. This includes the candidate selection for the line description (that is, while the inquiry message is outstanding, a call may be received on the line that was used and, therefore, that line is not available for use). It also includes all of the associated first-level error recovery procedures, if necessary. When the connection is successfully made, station-to-station contact sequences can proceed. If these are successful, a session can be established and the file open attempt can complete with a good status.

Remotely Started Normal Deactivation Sequences

After successful contact and communications with a remote system, the remote system may start normal deactivation sequences. For example, in an AS/400-to-AS/400 system nonswitched point-to-point environment, one end of the line may need to be varied off. In this case, the application programs are either complete or canceled, and the device descriptions, the controller description, and the line description are varied off. Varying off includes sending SNA protocol messages, informing the other end of the line at both the device and controller level. At each stage, the system not varied off receives SNA control information, informs the OS/400, and places the device description and controller description in the VARY ON PENDING status. If the side that is up is either negotiable or primary, connection polling begins. If polling is set up to be performed indefinitely, no further messages are received by the operator until the remote system is once again varied on and contact is attempted. If polling is not indefinite and the retry connection limit is exceeded, the controller becomes inoperative and second-level error recovery procedures are called as described previously in the nonswitched point-to-point work station example.

If the secondary end of the line is not varied off, it waits until it is contacted again by the primary end or until it is varied off.

AS/400-to-AS/400 System on an Ethernet Network

In this example, two AS/400 systems are connected by an Ethernet network. The application program is for an environment where users from either system want to pass through to the other system to access application programs and data.

An Ethernet network is a logically switched type of connection. Therefore, no successful connection can be made until both systems are operating, and ready with their respective local area network line descriptions varied on and their respective controller descriptions and devices in the VARY ON PENDING status. If one system is set up as DIAL and the other as ANSWER, either both systems have to be coordinated so that the ANSWER system is always ready before the DIAL system, or an operator has to periodically answer inquiry messages to attempt redialing, hoping that the ANSWER system is ready. In this situation, the last system ready starts the connection.

On an AS/400 system, configuration can be done through the configuration parameters in the system controller description and the second-level error recovery procedures, so that the desired connection is made without operator intervention regardless of the order in which the systems are started. See the *LAN, Frame-Relay and ATM Support* book for more information on local area network connections.

In this example, both systems are configured to start the connection (by specifying INLCNN(*DIAL) on the CRTCTLAPPC command, and varying on the controllers). One system is not ready when the other starts operating. The first system attempts to contact the second at vary on time. The LANCNNRTY and the LANCNNTMR parameters control the persistence of this first-level attempt. In this example, further attempts are useless. At this point, the operating system is notified that the contact attempt failed. The controller description is no longer associated with the line description (like dial failures on physically switched lines) and placed in a VARY ON PENDING status. In addition, an inquiry message to QSYSOPR or the configured message queue, notifies the operator of the connection failure.

If the message is answered with an R, another connection attempt is made. If the message is answered with a C, no dial is attempted now. However, if the message is not answered, and the remote system is then started and a connection is made, that connection is successful because the system automatically puts the controller description and the associated devices back to the VARY ON PENDING status. When the connection is made, answering outstanding inquiry messages has no effect.

The CMNRCYLMT value is not used to automatically attempt redialing. However, it is used to change a controller description from the RECOVERY PENDING status to the VARY ON PENDING status if failures occur after a successful connection is made. You can set up the communications line description and controller description and specify the CMNRCYLMT parameter value to create a local area network environment that requires little operator intervention. See the *LAN, Frame-Relay and ATM Support* book for configuration examples.

Remote Printer Considerations

A loss of power causes an exception to be reported by the system as a `Request Shutdown` message or a 081C0200 error to printer support in the OS/400 licensed program. Printer support sends the following messages:

Message or Exception	Open	Output	Close	Major or Minor Error Code
081C0200	CPF4192	CPF5143	CPF4533	8197
Request Shutdown	CPF4192	CPF5143	CPF4533	82A6

When the application program is the printer writer, the action that is taken depends on the current activity of the printer writer. If the printer writer is between files, and contact is again established with the printer before a file is ready for printing, the printer writer sends message CPF3421, attempts to recover from the loss of power, closes and again opens the printer file, and continues printing. If the error occurs while the printer writer is printing, the printer writer ends abnormally, and the exception is recorded in the job log of the printer writer.

When sending a file directly to a printer, without using a printer writer, exceptions are sent to the user's application program. The recommended action is to close and again open the printer file when the device is available.

Handling Class 3 Errors

Class 3 errors are generally failures in the communications controller hardware, AS/400 Licensed Internal Code, or situations detected for which continuing is useless or data accuracy is lost. These errors are identified by communications objects that have a FAILED status. Partial damage of these objects may also have occurred. Partial damage means a program running for the object failed to complete as expected. By marking the objects failed or partially damaged, various I/O operations are prevented, containing the failure. Failed objects are not to be confused with messages that use the word "failed". For failed objects, I/O operations should cease, jobs should be ended, and the affected objects should be varied off and varied on to recover. This forces a resetting of control blocks and the restarting of the underlying system tasks that supported that communications. It is possible that the sequence of events that led to the failure may not occur again, or not for a significant amount of time. Therefore, varying on the failed object again may recover the situation.

For especially severe errors that affect an AS/400 communications controller, you may have to vary off all line descriptions associated with the communications controller and specify the RESET(*YES) option on the Vary Configuration (VRYCFG) command for the first line being varied on again. This effectively starts up the communications controller again without requiring the system to be restarted. For information on how to determine all the lines on a communications controller, see the *Communications Configuration* book.

Class 3 errors should be reported to your service representative. The message that is sent to QSYSOPR or the configured message queue, contains instructions on how to report the errors.

Application Program Error Recovery

It is normally desirable to incorporate error recovery into application programs. This is true for user-written application programs, as well as system-supplied application programs. The error recovery may consist of the same program attempting to reestablish communications, or it may consist of simply ending the program and starting it again. The AS/400 operating system does not automatically restart application programs when errors occur. Depending on the function that an application program performs, it may be possible for a user to write programs to restart user or system-supplied application programs. However, programming an application program to automatically be restarted requires a thorough understanding of the following:

- Communications device status fields (see “Retrieving Configuration Status” on page 32)
- The application program
- Recovery messages (such as contact messages) being sent to message queues

The following information discusses general error recovery considerations for user-written application programs and for system-supplied application programs.

User-Written Applications

Depending on the communications method that is being used, a user-written application program can be designed to attempt to reestablish communications after an ending communications error has occurred. The communications method that is used dictates how the application is informed of communications errors. It also dictates which error recovery procedures can be attempted. Generally, any attempts at error recovery should be limited in number to prevent looping conditions.

The following sections contain general error recovery considerations for the various communications methods that your application programs may be using to communicate with other programs.

Using ICF

In each high-level language that supports ICF communications, the application program has access to major and minor return codes. The major and minor return codes inform the application program of the results of the last communications operation. All application programs should monitor these codes and take action that is based on them.

Some major return codes (such as 04, 34, 08, 11, and 83) show errors that can be corrected by the application program itself. Other major return codes (80, 81, and 82) can show configuration, hardware, or communication problems.

The operating system does not automatically start application programs again. For major return codes 80, 81, and 82, the session is ended. The application program can attempt to again establish a session by acquiring the appropriate program device or closing and again opening the ICF file if the acquire program device (ACQPGMDEV) parameter is used. The system suspends the open or acquire operation while the operator or operating system recovery is taking place. The application program gets control when those procedures are complete. If the procedures are successful, a session is established and the application program can resume communications. A file open or acquire operation generally fails if the WAITFILE time specified on the file is exceeded, or if second-level error recovery procedures are canceled either by a message or by a command.

Depending on the high-level language, application programs that do not monitor results of I/O operations may either be abnormally ended or allowed to continue operation. If the operations are allowed to continue, a looping condition may result. If several jobs go into a looping condition (for example, all the jobs on a failed line), severe system performance degradation may result.

In physically or logically switched environments, the WAITFILE parameter is not used for the following:

- SDLC switched lines
- X.25 switched virtual circuits
- Ethernet, DDI, frame relay, wireless, or token-ring logically switched networks
- Connections on ISDN switched channels

The open or acquire operation is suspended until either a successful connection is made or the connection attempts are canceled. This includes Contact Unsuccessful errors that are caused by a failure to successfully exchange XIDs after the physical or logical connection is made. (Note that the WAITFILE parameter *is* used when APPC devices are linked through APPN support.)

Many major and minor return codes can show that another open or acquire operation is required to continue communications. These return codes do not use the QCMNRCYLMT and CMNRCYLMT values, the second-level recovery information, or application program synchronization. That is, during such an error situation, another open or acquire operation can result in immediate notification to the application program that the file open or acquire failed. The application program should be aware of the potential for application program looping and system degradation if reopen or reacquire logic is unconditionally performed. The system does not control the number of opens attempted. See the *ICF Programming* book and the particular language reference manuals for details on major and minor return codes and programming considerations.

Target Program Considerations: The target program should test for errors, perform any error exit logic, and then end for any of the following intersystem communications functions:

- Advanced program-to-program communications (APPC)
- Asynchronous communications
- Binary synchronous communications (BSC)
- Finance communications
- Intrasystem communications
- Retail communications
- SNA upline facility (SNUF) communications

The source program may start *another* instance of the target program. Partial transactions are the responsibility of your application program.

Note: The source job *may not* connect itself again with the previous target job. An exception to this note is the SNUF subsystem, which uses a *protected session* concept.

Table 8 shows the major and minor return codes for second-level error recovery associated with an open or acquire operation and input and output operations in relationship to communications type.

Table 8. Major and Minor Return Codes for Second-Level Error Recovery

Communications type	Open/Acquire Operation	Output Operation	Input Operation
APPC	8291	8191	8191
Asynchronous	8191	8191	8191
BSCEL	8291	8191	8192
Finance	8281	8191	8191

Table 8. Major and Minor Return Codes for Second-Level Error Recovery (continued)

Communications type	Open/Acquire Operation	Output Operation	Input Operation
Intrasystem	8281	None	None
Remote Work Station	8291	8191	8191
Retail	8281	8191	8191
SNUF	8291	8191	8191

Using Binary Synchronous Communications

When using binary synchronous communications, a response is not sent to an EOT (end-of-transmission) control character. The sending station assumes that the EOT is received after the last data block is sent. If the EOT is not received, data integrity is not assured.

To ensure data integrity, it is recommended that you use user-implemented error detection and recovery capabilities. Some of these include:

- Sequential block numbering
- Appropriate checkpoint-restart capabilities
- Job numbering
- Message numbering
- Data format checking

Using SNA Upline Facility

SNUF may operate with protected sessions. Protected sessions can be started again after communications failures. You define a protected session by specifying MSGPTC(*YES) on the Add Intersystem Communications Function Device Entry (ADDICFDEVE) or Override Intersystem Communications Function Device Entry (OVRICFDEVE) command. When SNUF starts the session again, it exchanges information with the host system to decide whether any data must be sent again, and proceeds operating from that point.

Using a Display File

All work station application programs should test for error conditions, whether local or remote. A local device may be turned off in the middle of running an application program. However, because of the nature of remote communications, error conditions may occur more frequently in remote environments than in local environments. Therefore, you should analyze your application programs for sufficient error detection and handling logic when moving those application programs from local to remote environments. Generally, when the application program is notified that the communications resource failed, the application program should end as smoothly as possible. For example, a Sign Off command could be issued specifying that no job log be written. If the application program is running in a job as a result of signing on the system interactively, the sign-on can be recovered by the subsystem monitor. If the device is not managed by a subsystem, your application program is responsible for recovery.

Security becomes an application program consideration if application programs do not end, but continue after the communications resource recovers. This is because the time required for error recovery may be indefinite. For example, a second-level inquiry message is at QSYSOPR or the configured message queue, awaiting operator direction. In addition, the authorized operator at the original work station may no longer be at the device. Here, a different work station user could use the work station without having to sign on.

Using CPI Communications

Each Common Programming Interface (CPI) Communications call has a parameter called *return_code*. This *return_code* informs the application of the results of the CPI Communications call. Each application program that issues CPI Communications calls should monitor the *return_code* and take action that is based on the value returned.

Some *return_code* values (for example, CM_PROGRAMMING_ERROR_PURGING) indicate errors that can be corrected by the application program itself. Others indicate more serious errors. For example, a *return_code* of CM_PARAMETER_ERROR could mean that the *mode_name* that is specified by your program was not configured. In this case, your program must end and the error must be manually corrected. For another example, a *return_code* of CM_RESOURCE_FAILURE_RETRY could indicate a line failure. In this case, another route to the system could be available, or the operator or operating system may have successfully recovered the line which failed. So, for this *return_code* the application program could attempt to reestablish communications a limited number of times.

Some CPI Communications return codes end with RETRY or NO_RETRY. CPI Communications return codes that end with RETRY indicate that the error may be temporary, and that your program can attempt to take appropriate recovery action. The application program can attempt recovery action a limited number of times. CPI Communications return codes that end with NO_RETRY indicate that the condition is permanent, and that no recovery action should be attempted.

Recovery action is different between a source CPI Communications application program and a target CPI Communications application program. Source application programs can attempt to initialize and allocate a new conversation as a means of recovery. Target application programs should complete processing and end. A target application program can never be reconnected to the source program that started it.

When second-level error recovery is being performed, the operator and the operating system can be attempting recovery at the same time that your program is attempting application error recovery. When a source program is attempting to recover by allocating a new conversation, the allocate request is suspended by the system while the operator or operating system error recovery is taking place. The application program regains control when those procedures are complete. When those procedures are successful, a new conversation can be established and the application program can resume communications. When the procedures are canceled, either by a message or a command, control is returned to the application program with an error *return_code*. Care should be taken to attempt error recovery only a limited number of times to prevent looping conditions.

Some CPI Communications return codes can also indicate that the conversation has been ended, yet have nothing to do with second-level error recovery or with the QCMNRCYLMT and CMNRCYLMT values. If an application program attempts error recovery by initializing and allocating the conversation again in these cases, another failure may be reported immediately. The application must be designed to only attempt to allocate the conversation a limited number of times, or a serious system degradation could occur. For example, an application program that issues the

Allocate call while *return_control* is CM_IMMEDIATE would receive a *return_code* of CM_UNSUCCESSFUL if no sessions were available at that time. An application that unconditionally loops on the Allocate call until a session becomes available could severely degrade system performance.

See the *APPC Programming* book and the *Common Programming Interface Communications Reference* book for more information on CPI Communications and error return codes.

Using User-Defined Communications

User-defined communications support provides application program interfaces (APIs) that can be called from any high-level language that is supported on the AS/400 system. Each API has a return code parameter and a reason code parameter to indicate the results of the last operation. Application programs that use the user-defined communications APIs should monitor the return and reason code parameters and take action that is based on them.

Some return codes (such as 82 and 83) indicate errors that can be corrected without ending communications. Other return codes (80 and 81) indicate errors that require communications to end in order to recover. For example, an attempt to do output on a communications line after first-level error recovery was unsuccessful would result in a return code of 83 and a reason code of 4001. An attempt to do output on a communications line after second-level error recovery was canceled by the operator would result in a return code of 80 and a reason code of 4000.

See the *System API Reference* book for more information on user-defined communications APIs, return and reason codes, and programming considerations.

Using File Transfer

The file transfer support detects station or line inoperative conditions in the same way as an application program. When inoperative conditions are detected, the file transfer function sends a return code to the calling application program. For more information on file transfer support, see the *ICF Programming*.

Using a Printer File

Printer support is informed when either a controller or a line failure occurs on the current or next outstanding output operation.

The escape message that is sent on receipt of the error is shown in Table 9.

Table 9. Remote Printer Escape Messages

	Open	**Output**	**Close**	**Major or Minor Error Code**
Line	CPF4146	CPF5128	CPF4542	8191
Station	CPF4193	CPF5198	CPF4526	8181

When sending a file directly to a printer, without using a printer writer, exceptions are sent to the user’s application program. The recommended action is to close and open the printer file again when communications is reestablished with the printer.

Using Distributed Data Management

Communications failures are reported by DDM with OS/400 messages similar to those returned for errors occurring for local database files. For example, if a communication failure occurs during an I/O operation to a remote file, the following action occurs.

Communications support in DDM sends an OS/400 message in the CPF9100 range to the job log (for example, CPF9152, `Error occurred during DDM communications`). The database support in DDM puts a related message into another OS/400 message that is returned to the calling application program. The message depends on the type of function that is being requested at the time of failure. In this example, a CPF5169 message is sent saying, `Cannot complete input or output to DDM file....` The messages that are sent to the calling application program tell the user to look at the previous messages to decide the actual failure.

DDM provides no retry function except those provided by the lower communications support layers. After a communications failure, you cannot again connect the session between the DDM source job and the DDM target job, because the DDM target job ends when a communication failure is detected, closing any files that had been opened. You must restart the application program at or before the point where you refer to any remote files.

System-Supplied Applications

Error recovery is also an important consideration when using a system-supplied application program. Depending on the application that is being used, the error recovery can be quite different. The following sections contain general information concerning error recovery procedures for various system-supplied application programs.

Using 3270 Emulation and 5250 Display Station Pass-Through

3270 emulation for BSC and SNA, and for 5250 display station pass-through, is designed to return the physical device immediately to the user when an associated line or controller error is detected on the path to the system to which the device is logically connected.

Notice that if display station pass-through is driving a switched connection, and that connection is not successful, the display station operator must do one of the following:

- Cancel the pass-through request to regain control of the device
- Answer with a C (Cancel) the inquiry message associated with the failed connection

Using Remote Job Entry

BSC and SNA remote job entry (RJE) provide second-level error recovery through an automatic-restart function. If an RJE session ends abnormally and automatic restart is appropriately configured, RJE will attempt to restart the RJE session by using the Start RJE Session (STRRJESSN) command.

You should provide a user program that is called before each restart attempt. Use the program to perform installation-specific functions that are not part of automatic restart. These functions can include checking the status of the communications line and performing any necessary recovery (such as varying on the line), and resubmitting an interrupted job.

Using Distributed Systems Node Executive

The distributed systems node executive (DSNX) is an IBM application program that uses SNUF communications support. If the SNUF session is lost, the DSNX application program ends. The remote system NetView Distribution Manager user must again call the AS/400 DSNX application program after the communications resource has been recovered to continue communicating. See the *DSNX Support* book for more information on DSNX.

Using a Printer File

When the application program is the printer writer, it makes one recovery attempt. That is, for any open failure, the program attempts the open operation again. The printer writer then ends normally for any open exception.

If a failure occurs at output time, the printer writer closes and opens the file again. The printer writer ends normally for any new exception. If the next open is successful, the printer writer sends message CPA3301 or CPA3302 to the operator. The operator can then select how printing is started. When a close failure occurs, the close is attempted again. The printer writer then ends normally for any new exception. The exceptions are entered in the job log of the printer writer.

Using SNA Distribution Services Subsystem

The SNA distribution services (SNADS) subsystem is used by the object distribution function and the OfficeVision licensed program to distribute to another SNADS system. SNADS provides some application-level error recovery procedures in addition to the first- and second-level communications error recovery procedures. These additional procedures are also of a *retry* nature and, therefore, they relate to communications recovery.

If SNADS is not active, there is no recognition that any error situations have occurred or are occurring. However, when SNADS is activating or is already active and an inoperative condition occurs, the SNADS application program is informed at either its outstanding or next input or output operation. SNADS does not attempt an open or acquire operation again immediately, but puts itself into a WAIT status, and sends message CPI8805 (CPI3A31 for *SVDS) to the following:

- The job log
- The QSYSOPR message queue
- The configured message queue
- The device message queue
- The QHST message queue

After that WAIT status, SNADS attempts an open or acquire operation again. If the open or acquire operation attempt is successful, SNADS proceeds to resend the next distribution. If the communications resource is not recovered, similar to any application program that opens or acquires, the job is suspended. The job is suspended until communications recovery is successful. Communications recovery is canceled by one of the following:

- The operator answers C (Cancel) to an outstanding communications recovery inquiry message.

- The operator enters an End Network Interface Recovery (ENDNWIRCY), End Line Recovery (ENDLINRCY), End Controller Recovery (ENDCTLRCY), or End Device Recovery (ENDDEVRCY) command.
- The WAITFILE time for the file is exceeded (the default is 120 seconds for the SNADS file (QCSNADSC in library QSYS)).

If the session allocation fails again for any of the previous reasons, SNADS again goes into a WAIT status. This wait-try again-wait loop is tried again up to the specified count limit. If the number of SNADS-specified retries is exceeded, the SNADS sender job suspends, and sends message CPI8816 (CPI3A32 for *SVDS) to the following:

- The job log
- The QHST message queue
- The device message queue
- The QSYSOPR message queue
- The configured message queue

The queue status is set to Rty-Fail (to display the status, use the Work with Distribution Queue (WRKDSTQ) command). To continue with the distributions, the operator must restart the sender job by running the WRKDSTQ command for that distribution queue, or send, hold, or release the queue to clear the failed state. The send, hold, and release functions of the WRKDSTQ command can also be done within a CL program with the Send Distribution Queue (SNDDSTQ), Hold Distribution Queue (HLDDSTQ), and Release Distribution Queue (RLSDSTQ) commands.

Error Recovery Procedures Parameter Selection Flow Charts

The following flow charts identify the specific configuration parameters in the network interface descriptions, line descriptions and controller descriptions used with first-level error recovery procedures, based on each data link type and the configuration you are using.

Parameters for Asynchronous Communications Error Recovery Procedures

Figure 23 on page 159 shows the parameters that are used for first-level error recovery procedures with asynchronous communications.

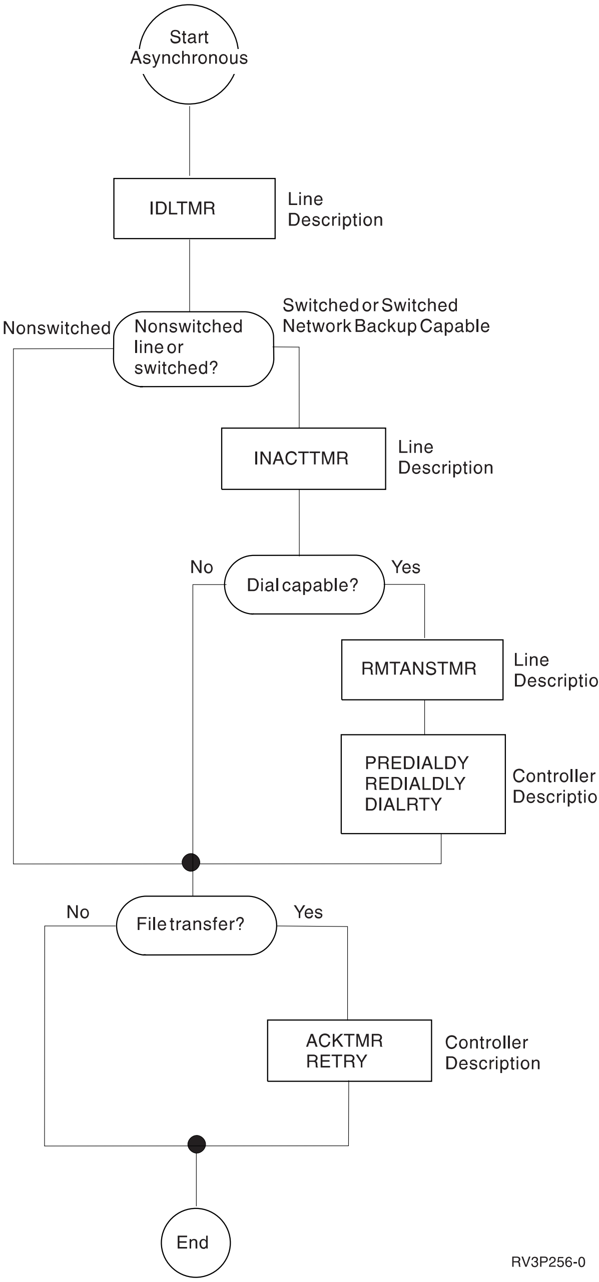
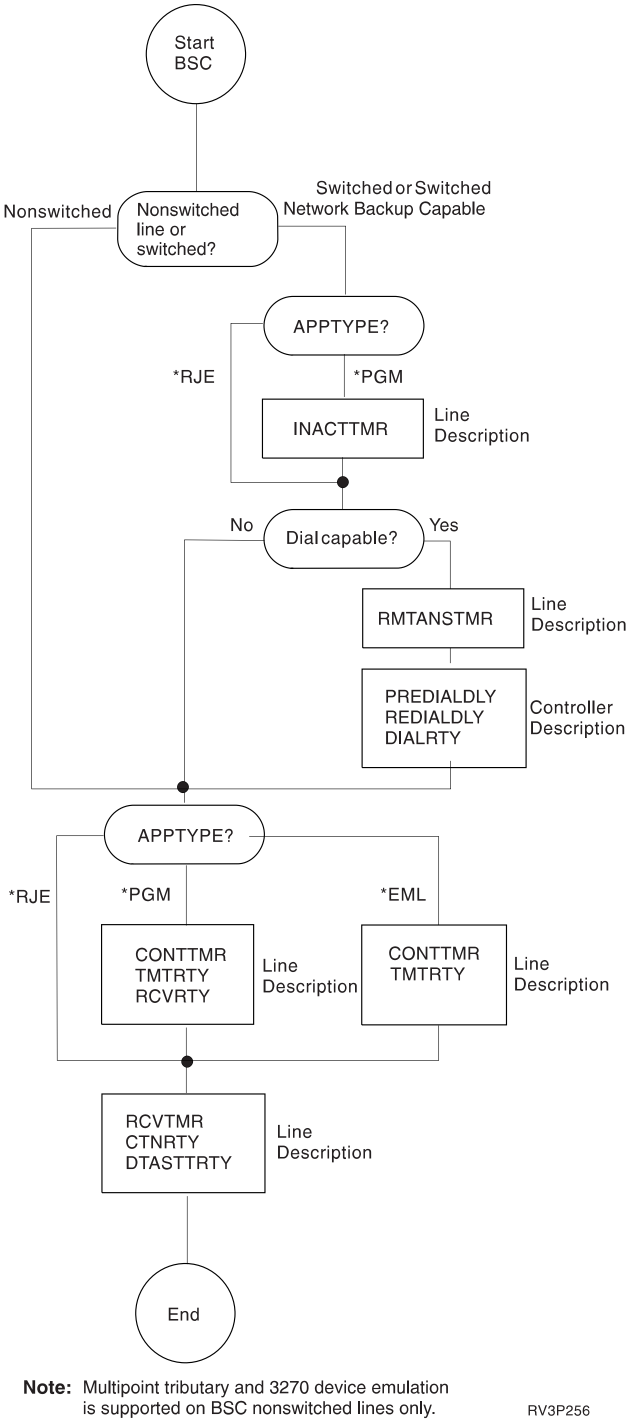


Figure 23. Parameter Selection for Asynchronous Communications Error Recovery Procedures

Parameters for Binary Synchronous Communications Error Recovery Procedures

Figure 24 on page 160 shows the parameters that are used for first-level error recovery procedures with binary synchronous communications.



Note: Multipoint tributary and 3270 device emulation is supported on BSC nonswitched lines only.

Figure 24. Parameter Selection for BSC Error Recovery Procedures

Parameters for Ethernet Network Error Recovery Procedures

Figure 25 on page 161 shows the parameters that are used for first-level error recovery procedures with an Ethernet network.

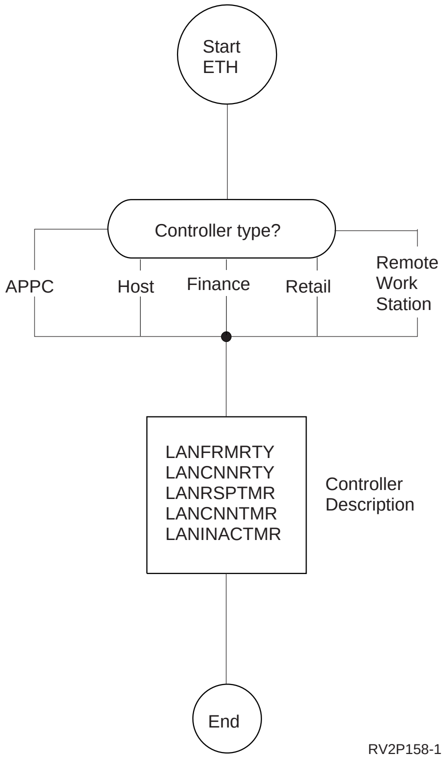


Figure 25. Parameter Selection for Ethernet Network Error Recovery Procedures

Parameters for ISDN (Integrated Services Digital Network) Error Recovery Procedures

Figure 26 shows the parameters that are used for first-level error recovery procedures with ISDN. Note that the NETTYPE parameter in the network interface description must match the network type you are using.

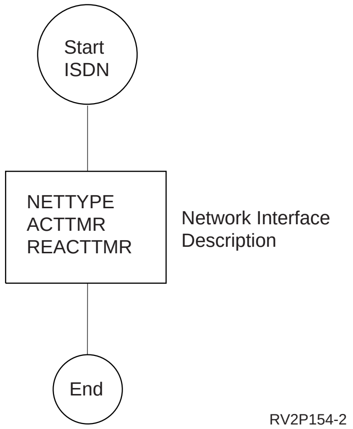


Figure 26. Parameter Selection for ISDN Error Recovery Procedures

Parameters for IDLC Error Recovery Procedures

Figure 27 shows the parameters that are used for error recovery procedures with an ISDN data link control.

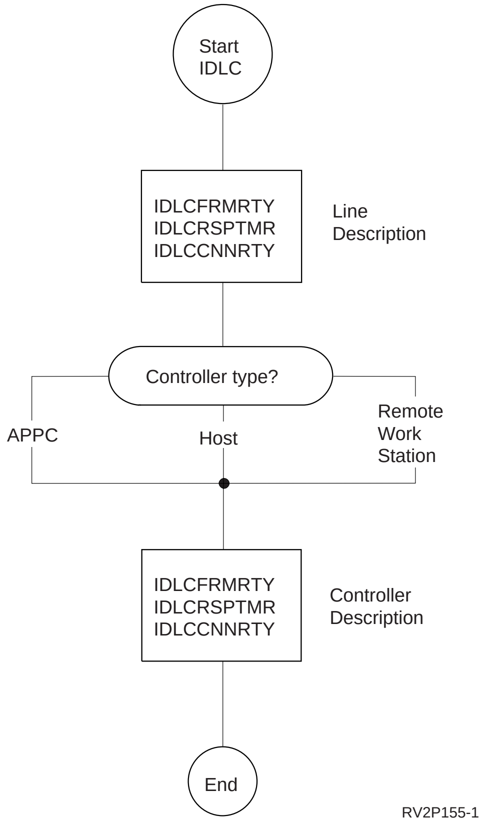


Figure 27. Parameter Selection for IDLC Error Recovery Procedures

Parameters for IBM Token-Ring Network Error Recovery Procedures

Figure 28 on page 163 shows the parameters that are used for first-level error recovery procedures with an IBM token-ring network.

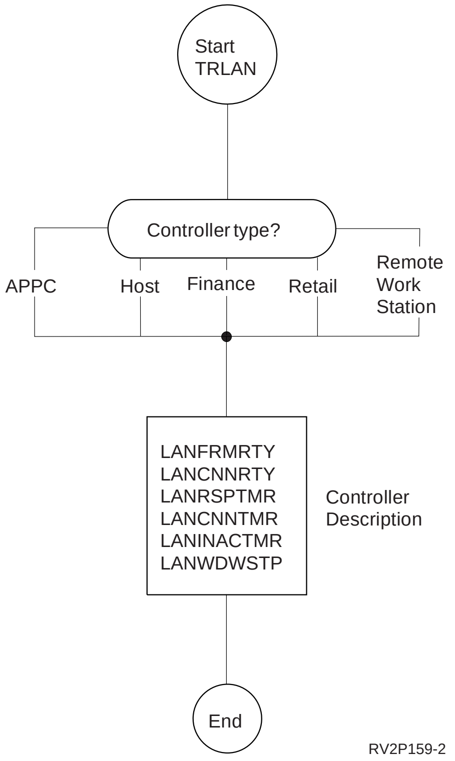


Figure 28. Parameter Selection for token-ring Network Error Recovery Procedures

Parameters for Synchronous Data Link Control Error Recovery Procedures

Figure 29 on page 164 shows the parameters that are used for first-level error recovery procedures with synchronous data link control.

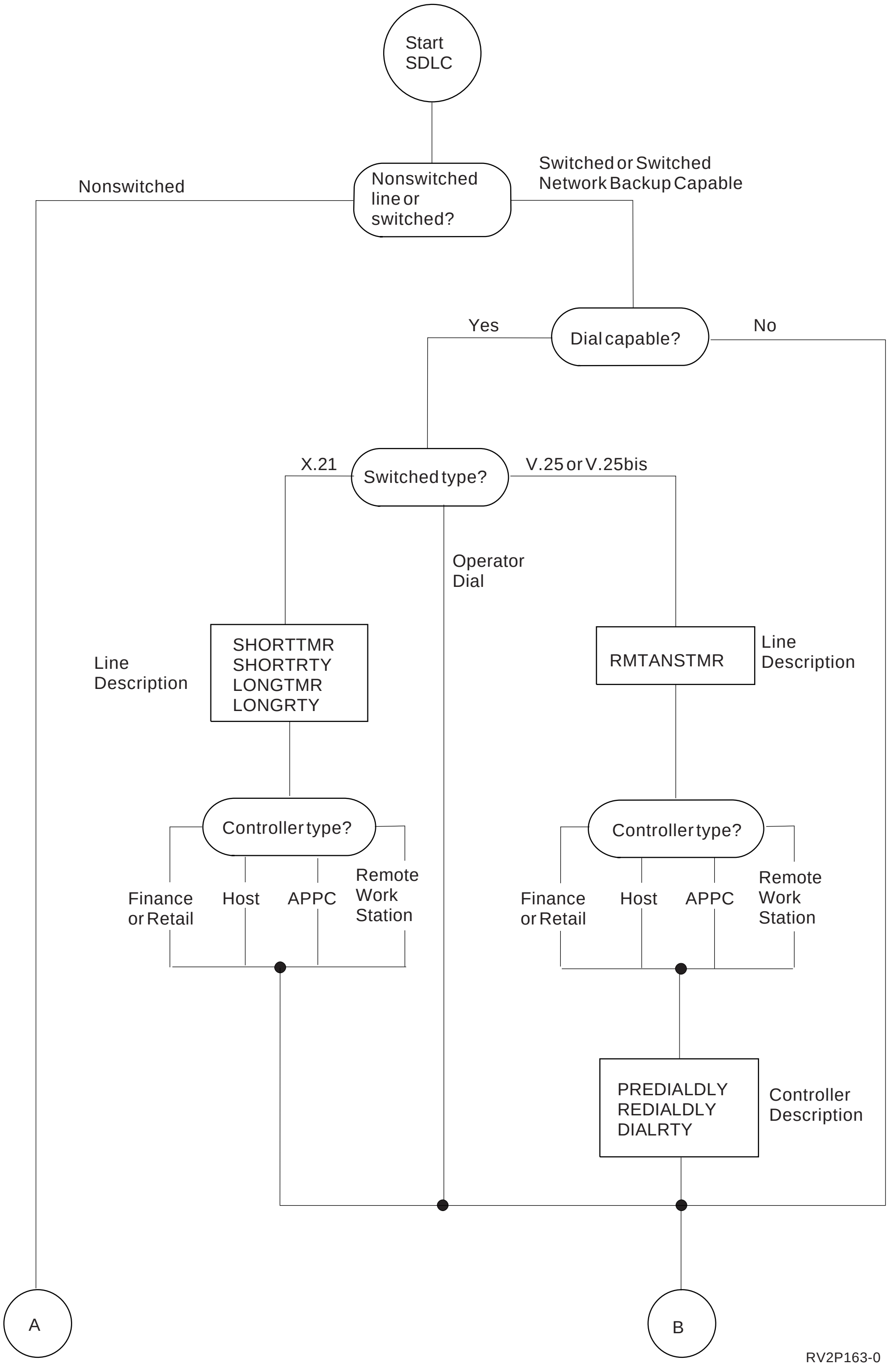


Figure 29. Parameter Selection for SDLC Error Recovery Procedures (Part 1 of 3)

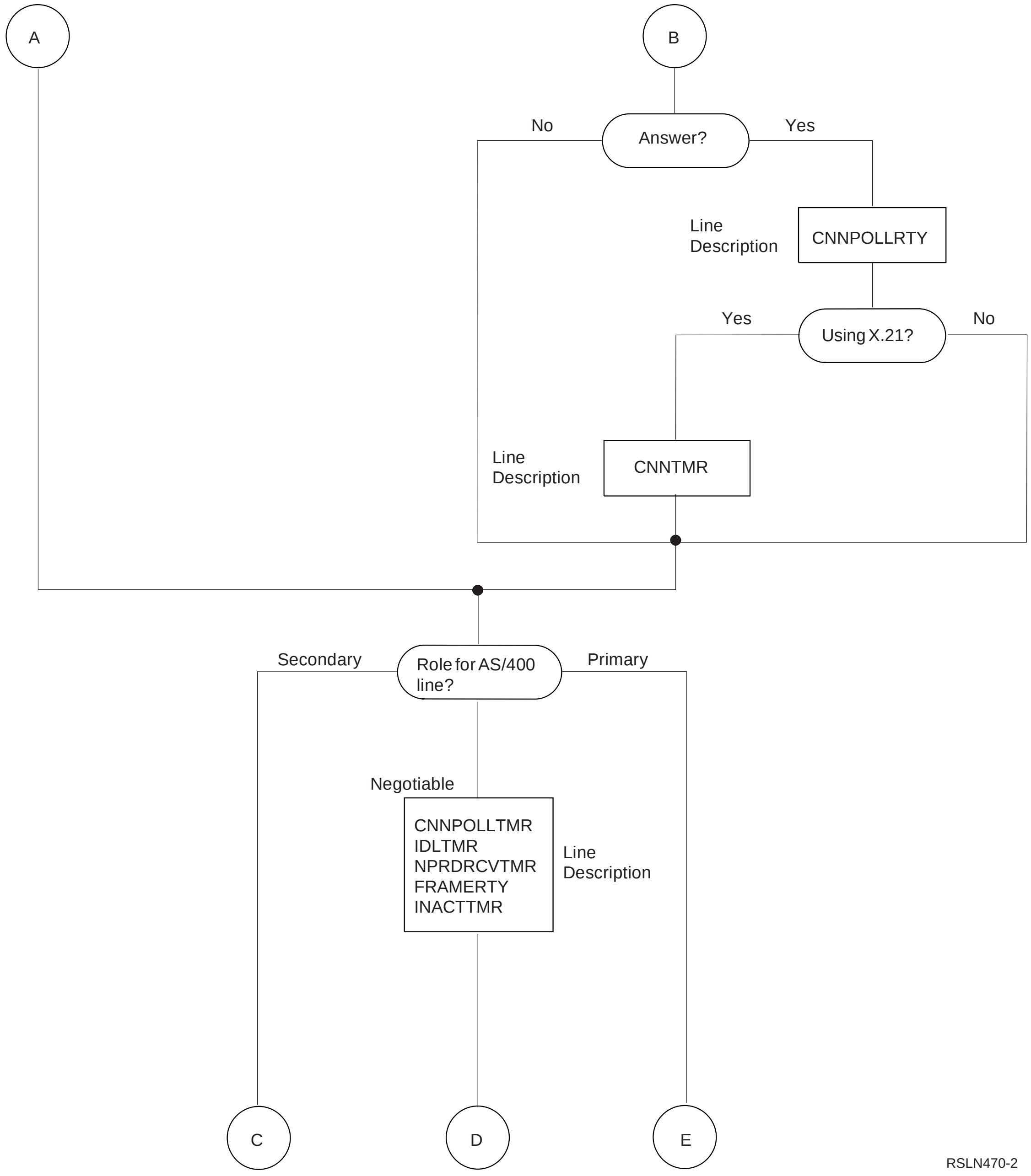


Figure 29. Parameter Selection for SDLC Error Recovery Procedures (Part 2 of 3)

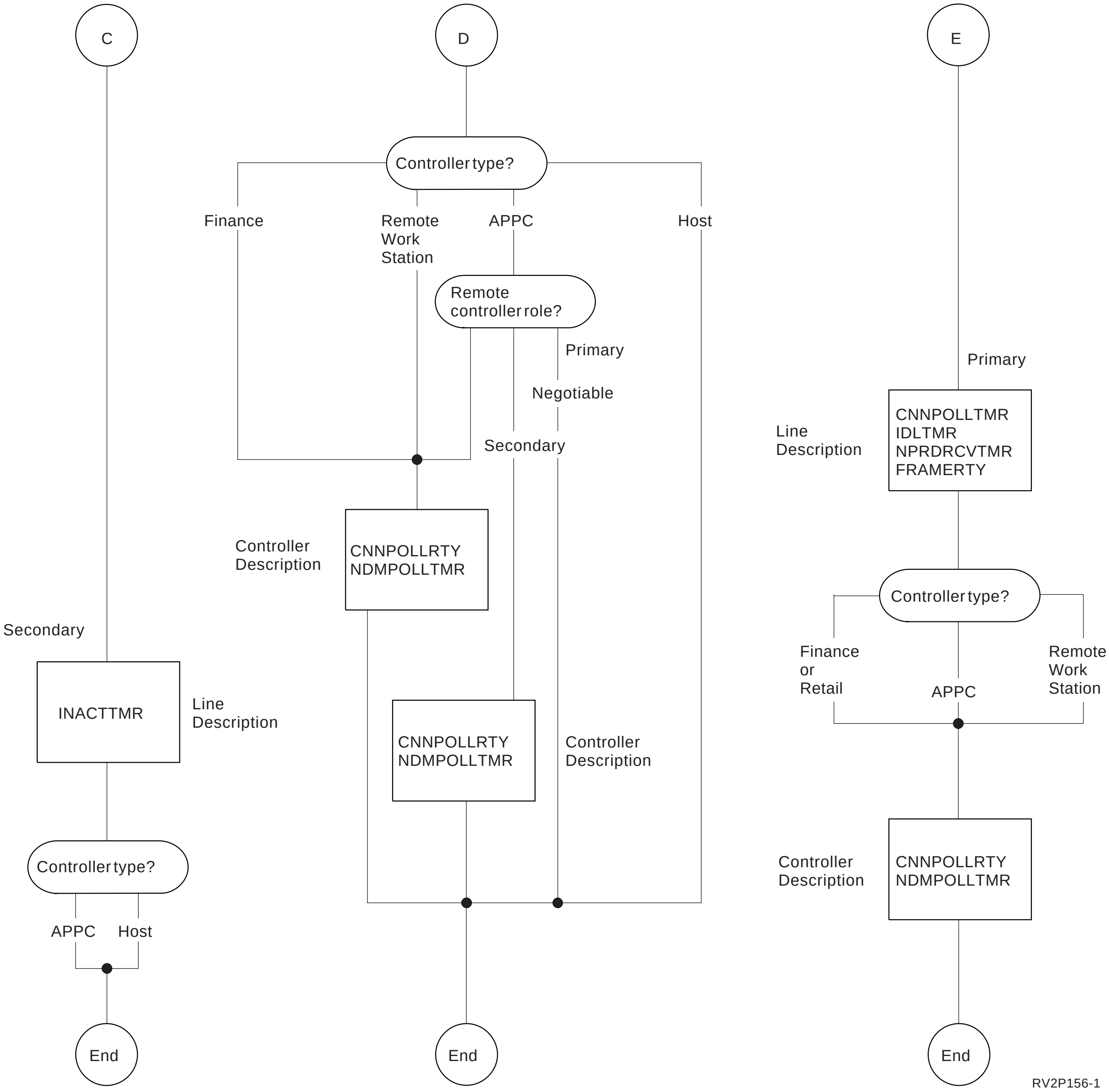


Figure 29. Parameter Selection for SDLC Error Recovery Procedures (Part 3 of 3)

Parameters for Frame Relay Network Error Recovery Procedures

Figure 30 on page 167 shows the parameters used for first-level error recovery procedures with a frame relay network.

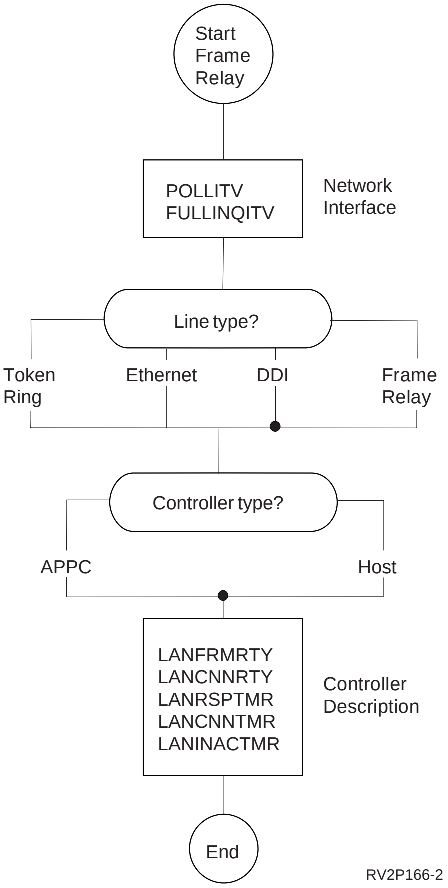


Figure 30. Parameter Selection for Frame Relay Network Error Recovery Procedures

Parameters for DDI Network Error Recovery Procedures

Figure 31 on page 168 shows the parameters used for first-level error recovery procedures with a frame relay network.

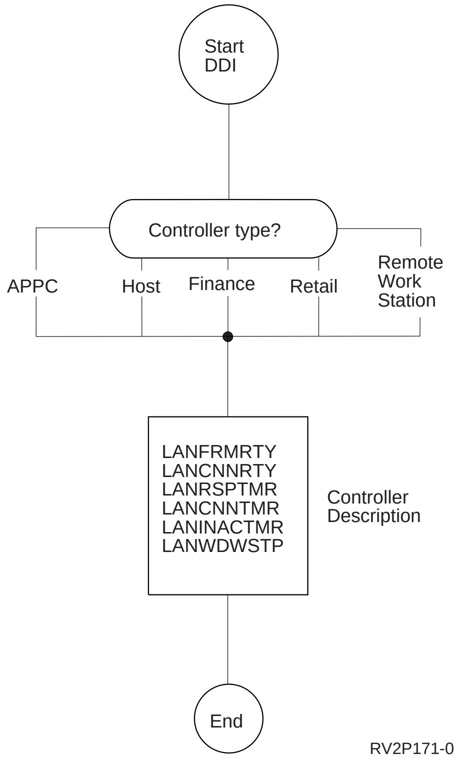


Figure 31. Parameter Selection for DDI Network Error Recovery Procedures

Parameters for Wireless Network Error Recovery Procedures

Figure 32 on page 169 shows the parameters used for first-level error recovery procedures with a wireless network.

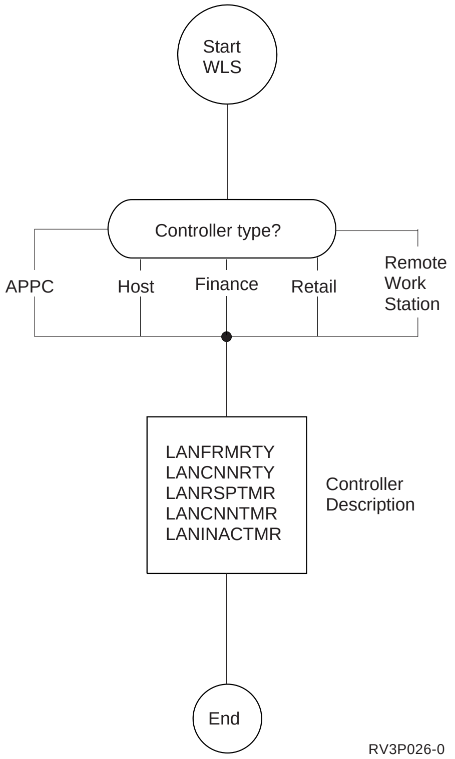


Figure 32. Parameter Selection for Wireless Network Error Recovery Procedures

Parameters for X.25 Error Recovery Procedures

Figure 33 on page 170 shows the parameters that are used for first-level error recovery procedures with X.25.

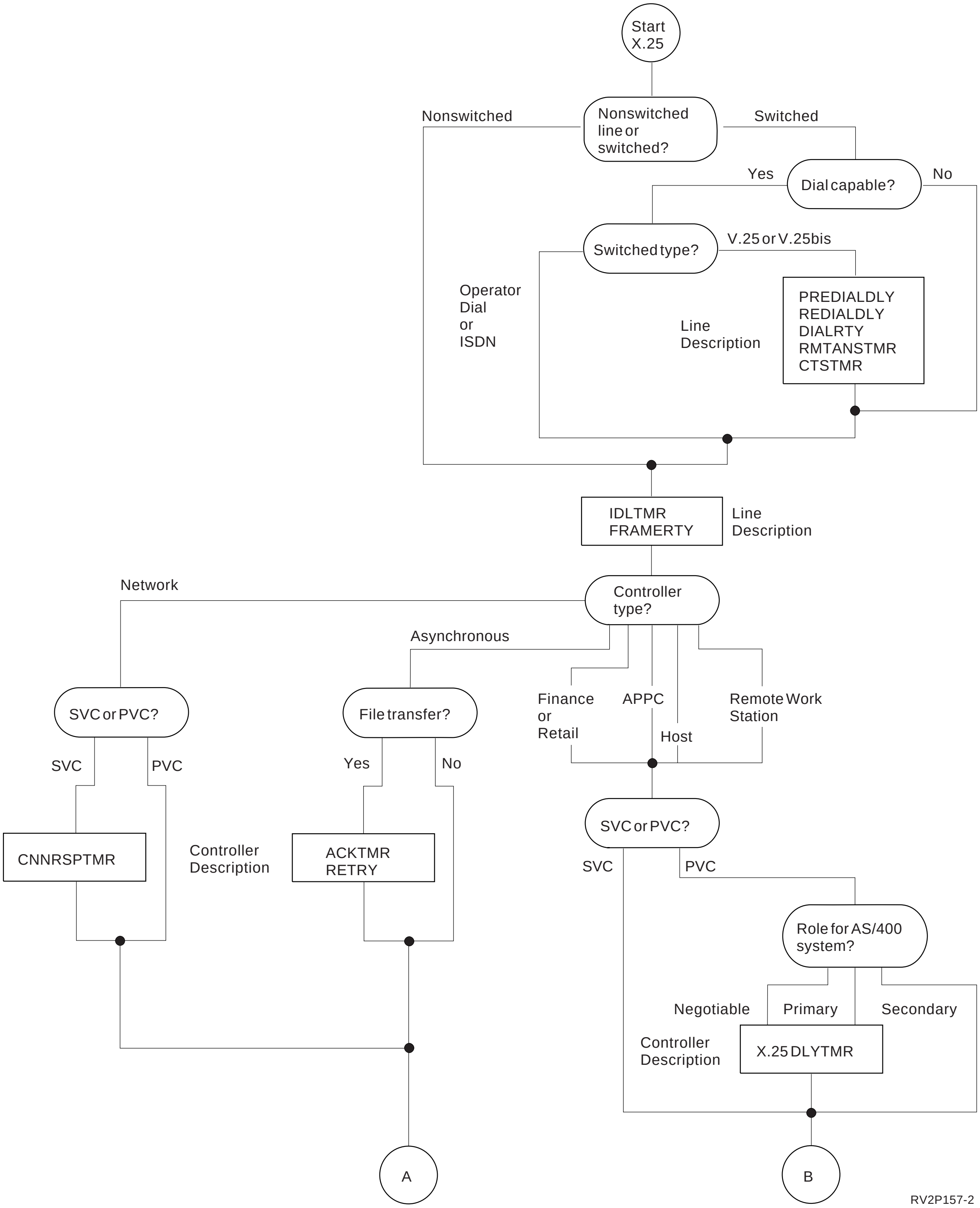


Figure 33. Parameter Selection for X.25 Error Recovery (Part 1 of 2)

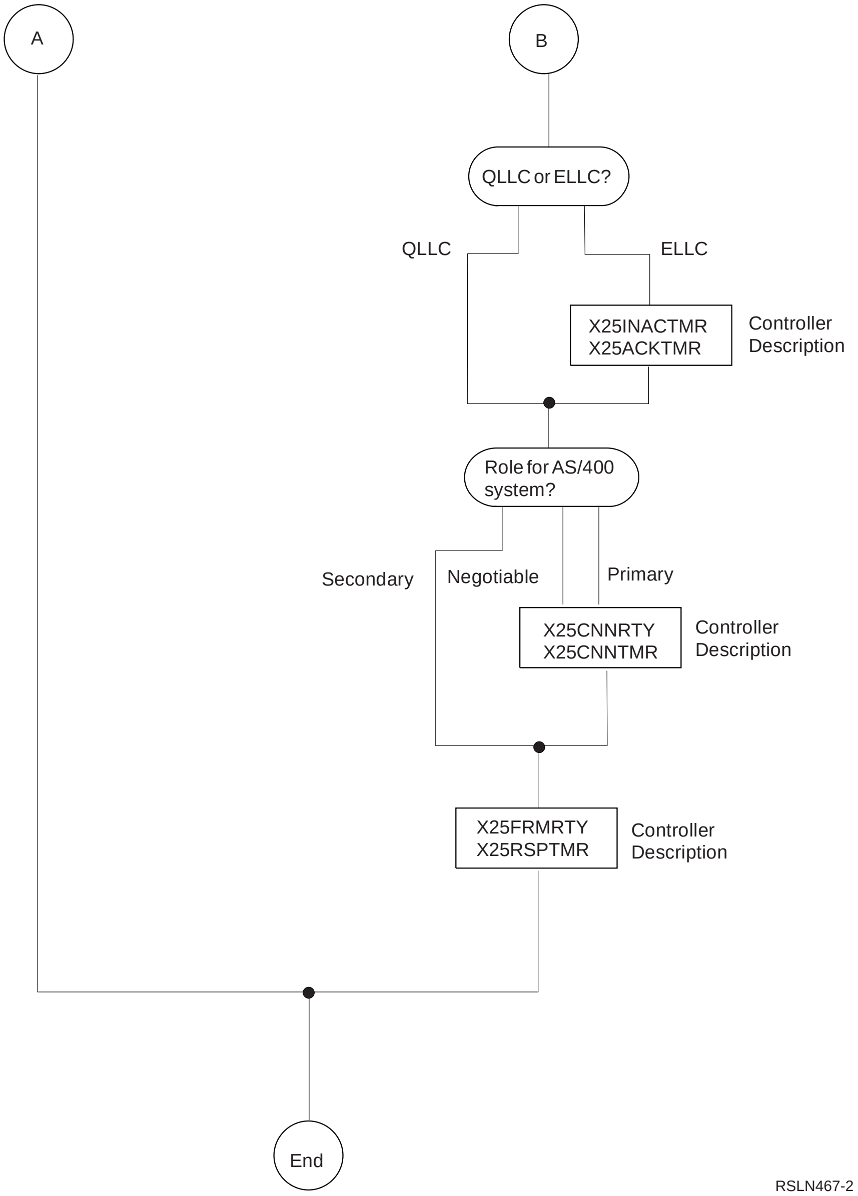


Figure 33. Parameter Selection for X.25 Error Recovery (Part 2 of 2)

Chapter 5. Communications Threshold Process

The threshold process is used in problem analyzing and predictive maintenance for remote communications on an AS/400 system. The communications controller use threshold measurements to report error conditions on a communications line that exceeds a specified rate. The threshold process is also used to decide if performance degradation is because of many recoverable errors, resulting in no *failed* messages to QSYSOPR or the configured message queue.

Note: Thresholds are no longer supported for LAN lines.

Thresholds are measured according to the number of occurrences of an error condition per:

- The number *(n)* data units transmitted or received, or,
- A set time interval *(t)* (ISDN only)

For ISDN, the measurement interval is based on time rather than the number of bytes transmitted. The values *n* or *t* are called the **threshold denominators**. There is one threshold denominator for both incoming and outgoing traffic. This threshold denominator actually defines a fixed-size measurement interval. Error conditions are measured interval by interval.

For each condition to be measured, a **threshold counter** is maintained that counts error conditions. The counter is reset once for every measurement interval.

Each time a threshold counter increments, it is compared to a corresponding **threshold limit** (the numerator). A message is reported to the operating system if the threshold counter value exceeds the threshold limit. The counter is then reset.

For example, for intervals based on data units, a numerator of 16 and a denominator of 256 mean that the AS/400 communications controller will, for each counter with these values, report a THRESHOLD EXCEEDED condition if more than 16 events occur in a 256 data-unit interval. At the end of 256 data units, the numerator is reset and counting begins again.

Selecting the Threshold Setting

You can use the Create Line Description (CRTLIN*xxx*, where *xxx* is the line type, for example, CRTLINIDLC) command to select one of the four threshold settings for a communications line. The system predefines several sets of values for the threshold denominator, and threshold limits for four threshold settings: Maximum, medium, minimum, and no thresholds. “List of Values for Threshold Settings” on page 175 defines maximum, medium, and minimum levels (This does not include ISDN). No threshold level means that the AS/400 communications controller maintain no threshold counters (threshold process is off).

You can set the thresholds with the error threshold-level (THRESHOLD) parameter of the create line description command.

Note: Threshold limits and the threshold denominator are set as a group. You cannot set the value for the threshold denominator or any one threshold limit without setting the values for the other threshold limits.

For ISDN only, a value of *SELECT is available on the THRESHOLD parameter which allows the individual threshold counters to change. This option is available on the Create Line (IDLC) (CRTLINIDLC) command only.

When you select, the threshold setting, the line description stores the values for the threshold denominator, and threshold limits. At vary on time, these values are sent to the AS/400 communications controller. The AS/400 communications controller uses the threshold denominator and the threshold limits to decide whether the occurrences of a condition exceed the defined rate.

Changing the Threshold Setting

You can use the Change Line Description (CHGLIN*xxx*) command to change the threshold setting for a remote communications line. In this instance, *xxx* is the line type, for example, CHGLINASC, CHGLINBSC, or CHGLINSDLC. There are four threshold settings to choose from on the error threshold-level (THRESHOLD) parameter of the change line description command. Changing threshold settings changes the values of the threshold limits and the threshold denominator that are stored in the line description. The threshold values are sent to the AS/400 communications controller only at vary on time. Therefore, you have to vary off the line description (if it is varied on), and vary it on again to make the communications controller aware of this change.

Exceeding a Threshold Limit

When a line or network interface description is varied on, the threshold counters in the communications controller begin counting error conditions. If one of the counters exceeds its corresponding threshold limit, the communications controller notifies the operating system. A message is sent to the QSYSOPR message queue or the configured message queue, that describes what counter exceeded the threshold limit and the action required by the user. The recovery may include changing values set in the line or network interface description (such as threshold settings, timer values, retry limits, buffer allocations, and so on). If the threshold limits need to be changed, you can use the change line description or change network interface description commands to select another set of threshold settings. Depending on what type of threshold was exceeded, the message may direct you to perform specific actions such as, `have the telephone company check the quality of the line`.

You cannot enter problem analysis [1] from threshold error messages. That is, no entry is made in the service activity log and problem analysis cannot be called from the message display. You can use the Verify Communications (VFYCMN) command to run link tests on your line. Online help is available to guide you through this function.

If the error condition continues and the retry limit in the line or network interface description is exceeded, the communications controller reports a failure for the line or network interface.

Note: If the retry limit is set very high, the system may become overloaded with threshold messages.

[1] For more information about problem analysis, see the *Basic System Operation, Administration, and Problem Handling* book.

This results in a message that is sent to QSYSOPR or the configured message queue, and an entry that is recorded in the service activity log. You can then run problem analysis for the failure from the displayed message or from the service activity log.

When a threshold is exceeded, an entry is always recorded in the error log. You can display these entries using the *Work with error log* prompt in the system service tools function. You can access this function by entering the Start System Service Tools (STRSST) command. The online help information explains how to use the STRSST command. The *Error type* prompt in the detailed display is set to `Threshold`. The display gives a short description of the error and its associated system reference code. It also gives detailed information about the line or network interface counters at the time the threshold is exceeded.

List of Values for Threshold Settings

Table 10 defines the values for each one of the six threshold settings for intervals based on the number of data units (non-ISDN). Notice that all the denominators are set to 256. The A and B settings indicate the threshold limit that pertains to the specific threshold error check.

Table 10. Threshold-Setting Values

Monitoring Level	Setting A	Setting B
Maximum	16/256	48/256
Medium	64/256	96/256
Minimum	128/256	160/256

Following is a list of the threshold limit error checks that are made and the associated threshold settings.

Threshold Error Checks

This topic lists the threshold error checks and the associated threshold settings that are made for each of the following network types:

- Asynchronous communications
- Binary synchronous communications
- SDLC non-X.21 communications
- SDLC X.21 communications
- X.25 communications

The values for threshold settings A and B are listed under the topic “List of Values for Threshold Settings”.

Asynchronous Communications Thresholds

The threshold error checks that are made on an asynchronous communications network and the associated threshold setting follow:

Table 11. Asynchronous Communications Network

Threshold Setting	Setting A	Setting B
Breaks received	x	

Table 11. Asynchronous Communications Network (continued)

Threshold Setting	Setting A	Setting B
Transmit adapter check	x	
Receive buffer overrun	x	
Incorrect stop bit	x	
Intercharacter time-out	x	
Characters discarded	x	
Incorrect parity bit	x	
Clear to send off error	x	
Data set ready glitch		x
Clear to send glitch		x
Received line signal detected glitch		x
Call attempted		x
Call completed with error	x	
Data link occupied error threshold for V.25 automatic call	x	
Abandon call retry error threshold for V.25 automatic call	x	
Present next digit error	x	
Distant station connected error	x	
Data set ready error	x	
Dialing digit length	x	

- Breaks received threshold: This is the number of breaks received allowed per denominator before sending a message to the system.
- Transmit adapter check threshold: This is the number of transmit adapter checks allowed per denominator before sending a message to the system.
- Receive buffer overrun: This is the allowed number of times that the number of bytes can exceed the buffer size per denominator before sending a message to the system.
- Incorrect stop bit threshold: This is the number of buffers received with incorrect stop bit errors allowed per denominator before sending a message to the system.
- Intercharacter time-out threshold: This is the number of intercharacter time-outs allowed per denominator before sending a message to the system.
- Characters discarded threshold: This is the number of receive characters discarded (buffer not available) that are allowed per denominator before sending a message to the system.
- Incorrect parity bit threshold: This is the number of buffers received with incorrect parity bit errors allowed per denominator before sending a message to the system.
- Clear to send off error threshold: This is the number of modem-caused clear to send off errors allowed per denominator before sending a message to the system.
- Data set ready **glitch** threshold: This is the number of modem-caused data set ready spikes in an electronic signal allowed per denominator before sending a message to the system.
- Clear to send glitch threshold: This is the number of modem-caused clear to send glitches allowed per denominator before sending a message to the system.

- Received line signal detected glitch threshold: This is the number of modem-caused received line signal detected glitches allowed per denominator before sending a message to the system.
- Call attempted threshold: This is the number of automatic calls attempted per denominator before sending a message to the system.
- Call completed with error: This is the number of automatic calls completed with errors per denominator before sending a message to the system.
- Data link occupied error threshold for V.25 automatic call: This is the number of data link occupied errors allowed per denominator before sending a message to the system.
- Abandon call retry error threshold for V.25 automatic call: This is the number of abandon call retry errors allowed per denominator before sending a message to the system.
- Present next digit error threshold: This counter contains a measurement of the total number of times the present next digit is ON when it should be OFF and is OFF when it should be ON per denominator before sending a message to the system.
- Distant station connected error threshold: This is the number of distant station connected errors allowed per denominator before sending a message to the system.
- Data set ready error threshold: This is a measurement of the total number of times data set ready or clear to send did not become active when expected per denominator before sending a message to the system.
- Dialing digit length threshold: This is a measurement of the total number of times the length of the telephone number was incorrect per denominator before sending a message to the system.

Binary Synchronous Communications Thresholds

The threshold error checks that are made on a binary synchronous communications network and the associated threshold setting follow:

Table 12. Binary Synchronous Communications Network

Threshold Setting	Setting A	Setting B
Vertical redundancy check/cyclic redundancy check error	x	
Data blocks received in error	x	
TTD error	x	
WACK error	x	
Hardware underrun	x	
Hardware overrun	x	
Receive time-out	x	
Continue synchronous characters time-out	x	
No synchronous character time-out	x	
Clear to send off error	x	
Data set ready glitch		x
Clear to send glitch		x
Received line signal detected glitch		x
Call attempted		x

Table 12. Binary Synchronous Communications Network (continued)

Threshold Setting	Setting A	Setting B
Call completed with error	x	
Data link occupied error threshold for V.25 automatic call	x	
Abandon call retry error threshold for V.25 automatic call	x	
Present next digit error	x	
Distant station connected error	x	
Data set ready error	x	
Dialing digit length	x	

- Vertical redundancy check/cyclic redundancy check error threshold: This is the allowed number of parity (vertical redundancy check/cyclic redundancy check) errors per denominator before sending a message to the system.
- Data blocks received in error threshold: This is the allowed number of data blocks received in error per denominator before sending a message to the system.
- TTD error threshold: This is the allowed number of temporary-text-delay (TTD) characters transmitted per denominator before sending a message to the system.
- WACK error threshold: This is the allowed number of wait-before-transmitting acknowledgment (WACK) characters transmitted per denominator before sending a message to the system.
- Hardware underrun threshold: This is the allowed number of hardware underruns per denominator before sending a message to the system.
- Hardware overrun threshold: This is the allowed number of hardware overruns per denominator before sending a message to the system.
- Receive time-out threshold (data mode): This is the allowed number of receive time-outs in data mode per denominator before sending a message to the system.
- Continue synchronous characters time-out threshold (data mode): This is the allowed number of continue synchronous time-outs in data mode per denominator before sending a message to the system.
- No synchronous character time-out threshold (data mode): This is the allowed number of times that no synchronous character received in 3 seconds time-out in data mode is allowed per denominator before sending a message to the system.
- Clear to send off error threshold: This is the allowed number of clear to send off errors per denominator before sending a message to the system.
- Data set ready glitch threshold: This is the allowed number of data set ready glitches per denominator before sending a message to the system.
- Clear to send glitch threshold: This is the allowed number of clear to send glitches per denominator before sending a message to the system.
- Received line signal detected glitch threshold: This is the allowed number of received line signal detected glitches per denominator before sending a message to the system.
- Call attempted threshold: This is the allowed number of automatic calls attempted per denominator before sending a message to the system.
- Call completed with error: This is the allowed number of automatic calls completed with errors per denominator before sending a message to the system.

- Data link occupied error threshold for V.25 automatic call: This is the allowed number of data link occupied errors per denominator before sending a message to the system.
- Abandon call retry error threshold for V.25 automatic call: This is the allowed number of abandon call retry errors per denominator before sending a message to the system.
- Present next digit error threshold: This is a measurement of the total allowed number of times present next digit is ON when it should be OFF and is OFF when it should be ON per denominator before sending a message to the system.
- Distant station connected error threshold: This is the allowed number of distant station connected errors per denominator before sending a message to the system.
- Data set ready error threshold: This is the total allowed number of times data set ready or clear to send did not become active when expected per denominator before sending a message to the system.
- Dialing digit length threshold: This counter contains a measurement of the total allowed number of times the length of the telephone number was incorrect per denominator before sending a message to the system.

SDLC Non-X.21 Communications Thresholds

The threshold error checks that are made on an SDLC non-X.21 communications network and the associated threshold setting follow:

Table 13. SDLC Non-X.21 Communications Network

Threshold Setting	Setting A	Setting B
Frame check sequence errors	x	
Overruns	x	
Received frames too short	x	
Received frames too long	x	
Abnormal ends received		x
Idle signals detected	x	
Clear to send off error	x	
Data set ready glitch		x
Clear to send glitch		x
Received line signal detected glitch		x
Call attempted		x
Call completed with error	x	
Data link occupied error threshold for V.25 automatic call	x	
Abandon call retry error threshold for V.25 automatic call	x	
Present next digit error	x	
Distant station connected error	x	
Data set ready error	x	
Dialing digit length	x	
Send count (Ns) errors	x	
Receive count (Nr) errors	x	

Table 13. SDLC Non-X.21 Communications Network (continued)

Threshold Setting	Setting A	Setting B
Response (T1) time-outs	x	

- Frame check sequence errors threshold: This is the allowed number of frame check sequence errors per denominator before sending a message to the system.
- Overruns threshold: This is the number of overruns per denominator before sending a message to the system.
- Received frames too short threshold: This is the allowed number of received frames too short per denominator before sending a message to the system.
- Received frames too long threshold: This is the allowed number of received frames too long per denominator before sending a message to the system.
- Abnormal ends received threshold: This is the allowed number of abnormal endings received per denominator before sending a message to the system.
- Idle signals detected threshold: This is the allowed number of idle signals detected per denominator before sending a message to the system.
- Clear to send off error threshold: This is the allowed number of clear to send off errors per denominator before sending a message to the system.
- Data set ready glitch threshold: This is the allowed number of data set ready glitches per denominator before sending a message to the system.
- Clear to send glitch threshold: This is the allowed number of clear to send glitches per denominator before sending a message to the system.
- Received line signal detected glitch threshold: This is the allowed number of received line signal detected glitches per denominator before sending a message to the system.
- Call attempted threshold: This is the allowed number of automatic calls attempted per denominator before sending a message to the system.
- Call completed with error: This is the allowed number of automatic calls completed with errors per denominator before sending a message to the system.
- Data link occupied error threshold for V.25 automatic call: This is the allowed number of data link occupied errors per denominator before sending a message to the system.
- Abandon call retry error threshold for V.25 automatic call: This is the allowed number of abandon call retry errors per denominator before sending a message to the system.
- Present next digit error threshold: This counter contains a measurement of the total allowed number of times the present next digit is ON when it should be OFF and is OFF when it should be ON per denominator before sending a message to the system.
- Distant station connected error threshold: This is the allowed number of distant station connected errors per denominator before sending a message to the system.
- Data set ready error threshold: This counter contains a measurement of the total allowed number of times data set ready or clear to send did not become active when expected per denominator before sending a message to the system.
- Dialing digit length threshold: This counter contains a measurement of the total allowed number of times the length of the telephone number is incorrect per denominator before sending a message to the system.
- Send count (Ns) errors threshold: This is the allowed number of send count errors per denominator before sending a message to the system.

- Receive count (Nr) errors threshold: This is the allowed number of receive count errors per denominator before sending a message to the system.
- Response (T1) time-outs threshold: This is the allowed number of response (T1) time-outs per denominator before sending a message to the system.

SDLC X.21 Switched Communications Thresholds

The threshold error checks that are made on an SDLC X.21 switched communications network and the associated threshold setting follow:

Table 14. SDLC X.21 Communications Network

Threshold Setting	Setting A	Setting B
Frame check sequence errors	x	
Overruns	x	
Received frames too short	x	
Received frames too long	x	
Abnormal ends received		x
Idle signals detected	x	
DCE controlled not ready time-out	x	
DCE uncontrolled not ready time-out	x	
DCE state unknown time-out	x	
DCE uncontrolled not ready	x	
DCE controlled not ready (quiescent phase)	x	
DCE state unknown (quiescent phase)	x	
DCE state overrun	x	
Incorrect character received	x	
DCE clear (call establishment)	x	
DCE controlled not ready (data transfer)	x	
Call collisions	x	
Missed incoming calls	x	
Missed clear indications	x	
Network provided information reception parity error	x	
Network provided information receiver overrun error	x	
Network provided information buffer overrun error	x	
ITU-T[1] time-out error	x	
Call progress signal reception	x	
New/changed call progress signal reception	x	
Send count (Ns) errors	x	
Receive count (Nr) errors	x	
Response (T1) time-outs	x	

Note:

1. This organization was known as the CCITT (International Telegraph and Telephone Consultative Committee.

- Frame check sequence errors threshold: This is the allowed number of frame check sequence errors per denominator before sending a message to the system.
- Overruns threshold: This is the number of overruns per denominator before sending a message to the system.
- Received frames too short threshold: This is the allowed number of received frames too short per denominator before sending a message to the system.
- Received frames too long threshold: This is the allowed number of received frames too long per denominator before sending a message to the system.
- Abnormal ends received threshold: This is the allowed number of abnormal ends received per denominator before sending a message to the system.
- Idle signals detected threshold: This is the allowed number of idle signals detected per denominator before sending a message to the system.
- DCE controlled not ready time-out threshold: This is the allowed number of data circuit-terminating equipment (DCE) controlled not ready time-out occurrences per denominator before sending a message to the system.
- DCE uncontrolled not ready time-out threshold: This is the number of DCE uncontrolled not ready time-out occurrences per denominator before sending a message to the system.
- DCE state unknown time-out threshold: This is the allowed number of DCE state unknown time-out occurrences or port monitor 2 detections of a missing DCE clock per denominator before sending a message to the system.
- DCE uncontrolled not ready threshold: This is the allowed number of DCE uncontrolled not ready transitions per denominator before sending a message to the system.
- DCE controlled not ready (quiescent phase) threshold: This is the allowed number of DCE controlled not ready transitions during the quiescent phase per denominator before sending a message to the system.
- DCE state unknown (quiescent phase) threshold: This is the number of DCE transitions to an incorrect or unknown state during the quiescent phase per denominator before sending a message to the system.
- DCE state overrun threshold: This is the allowed number of DCE transitions that are missed because of interrupt overruns per denominator before sending a message to the system.
- Incorrect character received threshold: This is the allowed number of characters received while monitoring in states 1 or 2 per denominator before sending a message to the system.
- DCE clear (call establishment) threshold: This is the allowed number of DCE clears during the call establishment phase per denominator before sending a message to the system.
- DCE controlled not ready (data transfer) threshold: This is the allowed number of DCE controlled not ready transitions during the data transfer phase per denominator before sending a message to the system.
- Call collisions threshold: This is the allowed number of call collisions detected by the port monitor per denominator before sending a message to the system.
- Missed incoming calls threshold: This is the allowed number of missed incoming calls per denominator before sending a message to the system.
- Missed clear indications threshold: This is the allowed number of missed clear indications per denominator before sending a message to the system.

- Network provided information reception parity error threshold: This is the allowed number of network provided information parity errors detected by the port monitor per denominator before sending a message to the system.
- Network provided information receiver overrun error threshold: This is the allowed number of receiver overruns detected by the port monitor while receiving network provided information data per denominator before sending a message to the system.
- Network provided information buffer overrun error threshold: This is the allowed number of receive buffer overruns detected by the port monitor while receiving network provided information data per denominator before sending a message to the system.
- ITU-T time-out error threshold: These are the allowed number of ITU-T DTE time-outs per denominator before sending a message to the system; 11 ITU-T time-out error limits are allowed.
- Call progress signal reception threshold: These are the number of call progress signals of the specified type received per denominator before sending a message to the system; 24 call progress signals are allowed.
- New/changed call progress signal reception threshold: This is the allowed number of new or changed call progress signals per denominator before sending a message to the system.
- Send count (Ns) errors threshold: This is the allowed number of send count errors per denominator before sending a message to the system.
- Receive count (Nr) errors threshold: This is the allowed number of receive count errors per denominator before sending a message to the system.
- Response (T1) time-outs threshold: This is the allowed number of response (T1) time-outs per denominator before sending a message to the system.

X.25 Communications Thresholds

The threshold error checks that are made on an X.25 communications network and the associated threshold setting follow:

Table 15. X.25 Communications Network

Threshold Setting	Setting A	Setting B
Checksum errors	x	
Extended logical link control rejects transmitted	x	
Extended logical link control rejects received	x	
Extended logical link control receive not ready received	x	
Restart request transmitted	x	
Restart indication received	x	
Reset indication received	x	
Frame check sequence errors	x	
Overruns	x	
Received frames too short	x	
Received frames too long	x	
Abnormal ends received		x
Idle Signals detected	x	
Clear to send off error	x	

Table 15. X.25 Communications Network (continued)

Threshold Setting	Setting A	Setting B
Data set ready glitch		x
Clear to send glitch		x
Received line signal detected glitch		x
Call attempted		x
Call completed with error	x	
Data link occupied error threshold for V.25 automatic call	x	
Abandon call retry error threshold for V.25 automatic call	x	
Present next digit error	x	
Distant station connected error	x	
Data set ready error	x	
Dialing digit length	x	
Send count errors	x	
Receive count errors	x	
Response (T1) time-outs	x	

- Checksum errors threshold: This is the allowed number of checksum errors per denominator before sending a message to the system.
- Extended logical link control rejects transmitted threshold: This is the allowed number of extended logical link control rejects transmitted per denominator before sending a message to the system.
- Extended logical link control rejects received threshold: This is the allowed number of extended logical link control rejects received per denominator before sending a message to the system.
- Extended logical link control receive not ready received threshold: This is the allowed number of extended logical link control receive not ready received per denominator before sending a message to the system.
- Restart request transmitted threshold: This is the allowed number of restart request transmitted per denominator before sending a message to the system.
- Restart indication received threshold: This is the allowed number of restart indication received per denominator before sending a message to the system.
- Reset indication received threshold: This is the allowed number of reset indication received per denominator before sending a message to the system.
- Frame check sequence errors threshold: This is the allowed number of frame check sequence errors per denominator before sending a message to the system.
- Overruns threshold: This is the number of overruns per denominator before sending a message to the system.
- Received frames too short threshold: This is the allowed number of received frames too short per denominator before sending a message to the system.
- Received frames too long threshold: This is the allowed number of received frames too long per denominator before sending a message to the system.
- Abnormal ends received threshold: This is the allowed number of abnormal ends received per denominator before sending a message to the system.
- Idle Signals detected threshold: This is the allowed number of idle signals detected per denominator before sending a message to the system.

- Clear to send off error threshold: This is the allowed number of clear to send off errors per denominator before sending a message to the system.
- Data set ready glitch threshold: This is the allowed number of data set ready glitches per denominator before sending a message to the system.
- Clear to send glitch threshold: This is the allowed number of clear to send glitches per denominator before sending a message to the system.
- Received line signal detected glitch threshold: This is the allowed number of received line signal detected glitches per denominator before sending a message to the system.
- Call attempted threshold: This is the allowed number of automatic calls attempted per denominator before sending a message to the system.
- Call completed with error: This is the allowed number of automatic calls completed with errors per denominator before sending a message to the system.
- Data link occupied error threshold for V.25 automatic call: This is the allowed number of data link occupied errors per denominator before sending a message to the system.
- Abandon call retry error threshold for V.25 automatic call: This is the allowed number of abandon call retry errors per denominator before sending a message to the system.
- Present next digit error threshold: This is a measurement of the total allowed number of times the present next digit is ON when it should be OFF and is OFF when it should be ON per denominator before sending a message to the system.
- Distant station connected error threshold: This is the allowed number of distant station connected errors per denominator before sending a message to the system.
- Data set ready error threshold: This counter contains a measurement of the total allowed number of times data set ready or clear to send did not become active when expected per denominator before sending a message to the system.
- Dialing digit length threshold: This counter contains a measurement of the total allowed number of times the length of the telephone number was not valid per denominator before sending a message to the system.
- Send count errors threshold: This is the allowed number of send count errors per denominator before sending a message to the system.
- Receive count errors threshold: This is the allowed number of receive count errors per denominator before sending a message to the system.
- Response (T1) time-outs threshold: This is the allowed number of response (T1) time-outs per denominator before sending a message to the system.

ISDN Communication thresholds

Threshold error checks for ISDN are handled differently on the system than other communications types. The threshold algorithm counts the number of errors in a given time interval rather than the number of errors in a number of data units. The threshold algorithm allows two time intervals to be specified for monitoring for errors. A shorter, more sensitive initial time interval, is used so that problems in short term jobs can be identified. The length of the shorter interval is dependent on the type of error.

A longer interval of 900 seconds (15 minutes) is also used for all errors. If the threshold for an error is reached within the shorter time interval, a message is sent to the system operator message queue. At this point, the second, longer time interval is used to monitor for errors. The threshold level for the longer interval must be reached before another threshold message is sent to the system operator message queue. When the longer time interval expires, both threshold intervals are reactivated. The user should receive no more than two threshold messages for the same error over the 900-second interval.

Threshold error checking for ISDN allows you to set the threshold values for the threshold counters on an individual basis. The value (*SELECT) on the THRESHOLD parameter for the Create Line Description (CRTLINIDLC), and the Change Line Description (CHGLINIDLC) for IDLC command allows the user to select values for each error counter. The user can specify a value of *MIN, *MED, *MAX, *OFF, or a numeric value within the range that is provided. The numeric values correspond to the number of errors that are allowed in a 900 second interval.

Chapter 6. Performance

Many factors can affect the performance of AS/400 application programs in a communications environment. This chapter discusses some of the more common factors and offers guidance on how to achieve the best performance with your particular application program. However, the performance of a particular system is dependent on the interaction of environment, system configuration, and overall system workload. Contact your IBM representative if you have questions about the performance and capacity of your system, or for performance recommendations tailored to your system.

More specific information regarding aggregate line speed and I/O processor storage considerations is in “Chapter 7. Communications Subsystem Controller Storage and Aggregate Line Speed” on page 225. For additional performance information related to error recovery, see “Chapter 4. Handling Communications Errors” on page 107.

The organization of the AS/400 support for communications can be divided into the following categories:

- Work management
- Physical network or line
  - Line speeds
  - Nonswitched vs. switched
  - Point-to-point vs. multipoint
- Data link protocol
  - Asynchronous communications
  - Asynchronous transfer mode (ATM)
  - Binary synchronous communications (BSC)
  - Distributed data interface (DDI)
  - Ethernet network
  - Frame relay
  - ISDN data link control (IDLC)
  - Synchronous data link control (SDLC)
  - Token-ring network
  - Wireless network
  - X.25
- Programming support functions

Note: Other factors, not listed here, may also apply.

The following topics discuss factors from each of these categories that can affect performance. These factors should be considered when you design and configure a communications environment.

The performance of your communications applications is affected by your subsystem description and storage pool definitions. For more information on subsystems storage, see Chapter 1. Work Management and the *Work Management* book.

Network and Line Considerations

The selection of the appropriate telecommunications line or network is one of the most important factors affecting the performance of a communications application program. It requires advanced planning and represents a significant long-term expense. Also, it is a significant factor in overall response time in a communications environment.

The following factors should be considered when selecting your line or network. For more specific information on aggregate line speed, refer to “Maximum Aggregate Line Speeds” on page 225.

Line Speed

An application program’s ability to transfer data or information is limited by the speed of the line or network. For this reason, it is important to ask a few questions about the nature of the information being transferred. For example:

- How much information must be moved?
  - What is the size of the file for a large file transfer?
  - What is a typical transaction for an interactive application program and how much data is sent and received for each transaction? A thorough understanding of the application program is required to answer these questions. Most interactive application programs only transmit and receive certain input and output fields from a user’s display instead of the whole display. These input and output fields, together with a small amount of control information, make up the data that is in the individual transactions.
  - How many application programs or users will be using the line at the same time?
- Is this a large transfer or interactive environment?
  - A good practice for interactive application programs is to make sure that the average line use does not exceed 50 percent. This ensures consistent response time for all users of the line.

    Note: Switched LANs and ATM networks, with their increased bandwidth, are capable of handling line use well in excess of 50 percent.
  - Line use for large transfer application programs typically can approach 100 percent with no adverse performance effects.
  - Mixing both interactive and large transfer application programs on the same line can reduce performance, and is not advised. If you choose to combine these programs, careful planning is required to ensure both programs are provided sufficient opportunity to use the line.
- What protocol is used?

  This topic is covered in more detail in the “Data Link Protocol Considerations” on page 241, but it is important to know that some protocols create more non-data traffic on a line than others.
- What block or frame size is used?
  - The block or frame size determines the maximum amount of data transmitted over the line or network in one operation. All of the protocols supported by the AS/400 system provide the capability to use more than one block size. Larger block sizes usually are more efficient for the line or network, and for the

  AS/400. They result in less protocol cost and fewer changes on the line or network between transmitting and receiving.
  - Larger block sizes may have little or no effect on certain types of interactive application programs. These programs contain many small transactions that cannot wait to be assembled into a larger block or frame, and cannot use the line or network as efficiently as application programs that can use larger blocks or frames. This difference in efficiency is less noticeable as the speed of the line increases.
  - Larger block sizes may not work well for error-prone lines or networks. Larger blocks have a higher probability for errors in this environment, and take longer to transmit again than smaller blocks do.
- Are line speeds greater than 19200 bps used?
  - For line speeds of 19200 bps (bits per second) or less, most application programs can fully use the bandwidth (data capacity) of the line. The performance in this environment depends largely on the ability of the line to transfer the data.
  - For line speeds of greater than 19200 bps, some application programs cannot fully use the bandwidth of the line. The performance in this environment depends much more on the performance of the application program itself. For example, for a batch program reading several database records, compressing or translating data before requesting the output operation may not deliver data to the line fast enough to take full advantage of the line speed. Multiple jobs such as these running at the same time can take advantage of the greater line speed.
  - Local area networks represent a special case. They are intended to be shared by many stations on the same network. Also, because the token-ring network has 4 Mbps or 16 Mbps (million bits per second) of bandwidth, and the Ethernet network has 10 Mbps of bandwidth, in most cases, a single hardware adapter cannot use the total network bandwidth. In all cases, the performance in these environments depends on the performance of the application program. The hardware adapter can be the factor limiting performance in cases where multiple application programs are using the same adapter.
- When should data compression be used for APPC?

  This topic is covered in more detail in “APPC Data Compression” on page 201. It is important to understand how data compression works and when it can be used to your advantage.

For line speeds of 19200 bps and less, answering these questions helps you to estimate the performance that can be expected in a given environment.

X.21 SHM Port Sharing Performance

Multiple port sharing is an arrangement for short-hold mode operation in which both the first call and a reconnection call (recall) for a population of data terminal equipment (DTE) are directed to any available port within a port group. Remote controllers in both single and multiple port sharing configurations may appear to hang while awaiting an available port if:

- No SHM disconnection occurs because a single, busy controller is monopolizing the port. Other controllers trying to use the port group will be unable to call in, and calls out will not be made. This situation can be caused by a controller with a fast printer or large number of devices attached.

- SHM disconnections do occur, but heavy outbound traffic to several remote controllers prevents other remote controllers from calling in.

These problems are most likely to occur in single port sharing configurations, but can also occur with multiple port sharing if the number of busy controllers exceeds the number of ports in the port group.

X.21 SHM support for the OS/400 licensed program includes two timers, the *maximum connect timer* and the *delay-for-answer timer*. These timers are designed to improve performance of port sharing connections where the number of remote controllers is greater than the number of available ports in the port group. In these situations, the timers ensure that calls to and from remote controllers will be completed, despite heavy outbound traffic on the port or port group by one or more other remote controllers. For more information on correctly configuring your controllers for port sharing, see the *Communications Configuration* book.

Line Disconnection on the AS/400 System

Switched lines are a limited resource on the system. It is important that these lines do not remain connected for a longer time than necessary. Line expenses also can be more costly than necessary if connections do not disconnect correctly.

Switched Line Disconnection

You can manually disconnect the switched line, or the system can automatically disconnect the switched line. For example, if you communicate with a host using SNA upline facility (SNUF) or the 3270 device emulation, the host should end the connection. The last file that closes is usually associated with the side causing the disconnection. The system records device usage, and uses this information to determine when to disconnect the line.

The Inactivity Timer, or connection profile (if using Operations Navigator), controls switched line disconnection on PPP line description (INACTTMR). If there is no TCP/IP data that is transferred for the length of time that is specified for the Inactivity Timer, the line is disconnected.

The information is used in different ways depending on the line protocol. To help make this function more understandable, example protocols and the way they are used are discussed separately in the following topics.

Manually Disconnecting Switched Lines

The system operator performs the following steps to manually disconnect a switched line:

1. Ensures all previously active jobs on the line are finished
2. Cancels all jobs using devices on the line that are not finished
3. Varies off all of the devices
4. Varies off active controllers

The switched line is disconnected after the controller is varied off. For switched X.25 lines, the line is not disconnected if SWTSDC(*NO) is specified in the line description, unless disconnected remotely.

Finance or Retail Controller Line Disconnection

If a switched line is connected to a finance controller that supports switched lines (3694, 4701, 4702, or FBSS) or a retail controller (3651, 3684, 4680, or 4684), the AS/400 system determines when to automatically disconnect the line. To make this determination, one of the following must occur.

For a finance controller:

- A session to a finance device TYPE(*FNCICF) attached to a 3694 controller ends.
- A session to a finance device TYPE(*FNCICF) attached to a 4701, 4702, or FBSS controller ends in which the device did not send an INIT-SELF command before the session was started.
- An UNBIND command to a finance device TYPE(*FNCICF) is sent in response to a TERM-SELF received from the device.
- A file to a finance device TYPE(4704, 3624, or 3694) is closed.
- A file to an attached 3270 device is closed.
- A finance device TYPE(*FNCICF) is varied off.

For a retail controller:

- A session to a retail device attached to a retail controller ends.
- A file to an attached 3270 device closes.
- A retail device is varied off.

When one of the above events occurs, the AS/400 system drops the line only if all of the following are true:

- There are no jobs associated with any of the attached retail or finance devices.
- There are no finance devices in an ACTIVE status. This status can be displayed on the Work with Configuration Status display.
- The Sign On display appears on all of the attached 3270 displays.
- No user has an outstanding Allocate Object (ALCOBJ) command to any of the attached devices.
- The controller description, the 3270 device description, or the SIGNOFF command indicates the line should be dropped.

A retail or finance controller description indicates the line should be dropped if the switched disconnect (SWTDSC) parameter is specified as *YES.

The corresponding parameter on the 3270 device description and on the SIGNOFF command is the DROP parameter. If this parameter is specified as *YES, the line will drop if all other conditions are met. The DROP parameter is used only in the case where closing a file to an attached 3270 device caused the AS/400 system to determine whether to drop the line. If the device description DROP parameter is specified as *NO, the AS/400 system will not drop the line regardless of the value on the controller SWTDSC parameter for the first sign-off. If the next user signs off with DROP(*YES) specified, the line will disconnect.

SDLC Primary-to-Remote Work Station Line Disconnection

The AS/400 system controls a synchronous data link control (SDLC) primary line connected to a work station controller. The system controls the starting and ending of communications, as well as the disconnecting of the switched line. A remote work station controller can still dial, but that call may be considered premature by the AS/400 system and, therefore, can be disconnected. The AS/400 system controls the connection in this situation.

If a switched line is connected to a 5250 controller (5251 Model 12, 5294, 5394, or an emulation of one of these products), the line is automatically disconnected by the AS/400 system if all of the following are true:

- At least one job with one of the attached devices was active and ended or closed a file to one of the devices.
- Only a subsystem monitor has an open file to any of the attached display devices. The Sign On display shows on all display devices allocated to a subsystem monitor.
- No user has an outstanding Allocate Object (ALCOBJ) command to one of the attached devices.
- All users sign off and the last person to sign off specifies DROP(*YES).

Premature Calls for Primary Lines

A premature call can occur for primary line types if the remote location attempts to make a switched connection with the system before a user does one of the following:

- Opens a file to the switched device
- Allocates the device with the ALCOBJ command
- Starts a subsystem using the switched device

The system disconnects the switched line and notifies the system operator that a premature connection occurred. This helps prevent inefficient use of the systems resources. You can prevent premature calls by ensuring that at least one of the three points listed above is true for at least one of the devices attached to the controller at the remote location.

SDLC Secondary Lines Using Host Controller-to-System/370 Line Disconnection

The secondary SDLC host controller indicates that the remote primary system (System/370 host or equivalent) is responsible for controlling the communications line. The remote primary system controls the starting of the communications, the ending of the communications, and the disconnecting of the switched line. However, the AS/400 system can cause the line to disconnect with the SWTDSC parameter on the controller description.

Refer to the host manuals for more information concerning host parameters that affect line disconnection.

A secondary switched line should be disconnected if all the following are true:

- The remote system correctly ended all communications sessions on the line.
- All files opened to a device attached to the controller are closed.
- The remote system sent a disconnect command to the AS/400 system or the AS/400 system caused the line to disconnect after the disconnect timer ended.

The system operator performs the following steps to manually disconnect a switched line:

1. Cancels all jobs using switched devices on the line
2. Varies off all the devices

3. Varies off the controller

The switched line is disconnected after the controller is varied off.

Depending on the setting of the disconnect timer (DSCTMR parameter), multiple disconnections and connections can occur. If the value is set to an adequate amount of time (the default is 170 seconds), the system can complete the processing of a command without a disconnection. The setting of DSCTMR is valid only for connections with the SWTDSC value set to *YES.

Premature Calls for Secondary Lines

Premature calls cannot occur for secondary lines. The host system controls the establishment of the data link; therefore, the call is never considered premature.

APPC/APPN Line Disconnection

The *APPC Programming* book, and the APPC, APPN, and HPR articles in the *AS/400e Information Center* contain detailed information and examples for creating configurations. Review the SWTDSC parameter on the Create Controller Description (APPC) (CRTCTLAPPC) command to understand the conditions of a switched line disconnection. Using the Start Mode (STRMOD), End Mode (ENDMOD), or Change Session Maximum (CHGSSNMAX) command with SWTDSC (*YES) for switched connections can degrade the performance of a line. Depending on the setting of the disconnect timer (DSCTMR), multiple disconnections and connections can occur. If the value is set to an adequate amount of time (the default is 170 seconds), the system can complete processing of a command without a disconnection. The setting of DSCTMR is valid only for connections with the SWTDSC value set to *YES.

Premature Calls for APPC and APPN Connections

Premature calls cannot occur for APPC or APPN connections.

BSC APPTYPE (*PGM) Line Disconnection

When APPTYPE(*BSC38) or APPTYPE(*RPGT) is specified for CRTDEVBSC

The BSC APPTYPE(*PGM) line indicates that either the AS/400 system or the remote system can control the communications line. The system can control the starting and ending of communications as well as the disconnecting of the switched line. The remote location can dial the AS/400 system; however, that call may be considered premature and, therefore, can become disconnected. A BSC APPTYPE(*PGM) switched line is disconnected if the file for a BSC device is closed or the program ends.

The system operator can manually disconnect a switched line by canceling the job that is using the switched device on the line, which causes the file to close.

A BSC switched line also disconnects if there is no activity for the time specified in the inactivity timer (INACTTMR) parameter.

When APPTYPE(*BSCEL) is specified for CRTDEVBSC

The BSC APPTYPE(*PGM) line indicates that either the AS/400 system or the remote system can control the communications line. The system can control the starting of communications, the ending of communications, and the disconnecting of the switched line. If the remote location dials the AS/400 system, the remote location is responsible for disconnecting the line. A BSC APPTYPE(*PGM) switched line is disconnected when the system that dialed ends the communications session.

The AS/400 system operator can force a switched line to disconnect by canceling the job that is using the switched device on the line, which causes the communications session to end abnormally.

A BSC switched line disconnects if there is no activity for the amount of time specified in the inactivity timer (INACTTMR) parameter on the Create Line (BSC) (CRTLINBSC) command. It disconnects if an abnormal end-of-transmission (EOT) control character is received and the value *NO is specified in the RMTBSCEL parameter in the device description, or on program device entry commands, such as ADDICFDEVE, CHGICFDEVE, and OVRICFDEVE.

Premature Calls for BSC APPTYPE(*PGM)

When APPTYPE(*BSC38) or APPTYPE(*RPGT) is specified for CRTDEVBSC

A premature call can occur for a BSC line if the remote location attempts to make a switched line connection with the system before a program opens a file to the device and performs a write or a read operation to the file. The AS/400 system disconnects the switched line and notifies the system operator if a premature call occurred to help prevent inefficient use of the system.

Premature calls can be prevented by ensuring that a file is open and a write or a read operation is performed on the file before attempting to establish the connection.

When APPTYPE(*BSCEL) is specified for CRTDEVBSC

A premature call cannot occur for BSC lines. The system that dials controls the establishment of the data link. Therefore, the call is never considered premature.

BSC APPTYPE(*RJE) Line Disconnection

The BSC remote job entry line type indicates that the remote system controls the disconnection of the switched line. The BSC remote job entry line type supports the System/370 host. A BSC remote job entry switched line should be disconnected if the following points are true:

- The remote system correctly ended all the communication sessions on the line.
- The remote system sent a disconnect command to the system.
- All files opened to a device attached to the controller are closed.

The system operator can manually disconnect a switched line by canceling the job that is using the switched device on the line, which causes the file to close.

A BSC switched line also disconnects if there is no activity for the amount of time specified in the inactivity timer (INACTTMR) parameter.

Premature Calls for BSC APPTYPE(*RJE): Premature calls cannot occur for BSC remote job entry lines. The remote system controls the establishment of the data link; therefore, the call is never considered premature.

X.25 Considerations

The link between the AS/400 system and an X.25 packet-switching data network can be a voice-grade telephone line, or an ISDN B-Channel. That line can be either non-switched or switched. There can be two levels of switched disconnection associated with X.25. If the line is a switched line, then the high-level data link control (HDLC) connection to the DCE is ended when the line is disconnected. If switched virtual circuits (SVCs) are defined in the line description, then a virtual circuit connection to the remote DTE is ended when a controller is disconnected. Switched disconnection can only occur at the controller level on a nonswitched line. Switched disconnection can occur on a switched line at both the controller and the line levels.

A switched controller description will be disconnected and the SVC connection will end when the following are true:

- The last SNA session using the controller description is unbound.
- SWTDSC(*YES) is specified in the controller description.
- The DSCTMR has ended.

If the line is switched, you can specify parameters on the line description to control disconnection of the line. These parameters are SWTDSC and DSCLINTMR. You can also specify the X.25 switched line selection (SWTLINSLCT) parameter in the controller description to tell the system how to select a line for calling the X.25 network. The system will either select a line from the SWTLINLST in the order that they are entered, or will use an algorithm to select the line that would incur the least cost when used. A switched line will be disconnected and returned to connect pending state when the following are true:

- SWTDSC(*YES) is specified in the line description.
- The last controller using the line description has been disconnected.
- The DSCLINTMR has ended.

For more information on X.25 switched connections, see the *X.25 Network Support* book. For more information on X.25 on ISDN, see the *ISDN Support* book.

Nonswitched vs. Switched Lines

Communications lines must be either nonswitched (the line is always available) or switched (the connection is established by dialing, and only made available when needed). A switched line should probably be used for short and infrequent transmission of batch files. Considerations include:

- Cost of the line (switched lines are usually less expensive)
- Distance between stations
- Frequency of use
- Duration of use
- Time to establish a connection
- Use of the line (batch or interactive transfer)
- Point-to-point (switched lines only) vs. multipoint

Telephone number lists

The AS/400 system supports telephone number lists for V.25, V.25bis, X.21, and X.25 switched lines. Each controller, created through the create controller or change controller commands, specifies 1 telephone number, and up to 256 controllers can be specified by an X.25 line description (LGLCHLE parameter). The system will try to make a switched connection, trying one telephone number after another until a connection is made or the list is completed.

Maximum Throughput

The maximum throughput on a line can depend on any of the following:

- Application program (how it is written and what task it is performing)
- Line quality (number of errors on the line)
- Line speed
- Modem considerations
- Overall system performance and capacity

Duplex vs. Half-Duplex Networks

A **duplex** line is one that can send and receive data at the same time. **Half-duplex** refers to communications that can be sent in only one direction at a time.

The most important factor when considering switched lines vs. nonswitched lines is the modem turnaround time. This is the amount of time required for a station on the line to stop receiving data and to begin transmitting data. Nonswitched lines generally have little modem turnaround time because they normally have four wires. These four wires can be used so that two wires are always ready to transmit data and the other two wires are always ready to receive. Switched lines can have a significant amount of modem turnaround time because they have only two wires.

Typical times for a switched line range from 0.1 second to 0.5 second, depending on the modem and the quality of the line connection. This is especially important for interactive application programs if a station alternates frequently between sending and receiving data, in some cases, many times for the same transaction. Large transfer application programs experience some degradation as well because most line protocols require multiple transitions from sending to receiving during the course of a transfer to ensure data integrity.

Point-to-Point vs. Multipoint Lines

It is not electrically possible for more than one station to transmit on a set of wires at one time. For this reason, secondary or tributary stations on a multipoint line must wait until they are allowed to transmit by the primary station. The amount of time to wait depends on the number of stations on the line and the particular protocol used. Some protocols provide a way to give higher priority to individual stations on the line. Point-to-point lines, however, do not have any wait time because they connect two systems to each other without the possibility of additional transmissions from another system.

Line Speed Examples

The following examples show how you can use this information to understand the effect that the line or network has on an application program. These examples apply to the line or network only and do not attempt to describe any of the other factors that can influence the performance of an application program. Also consider the other factors, such as the number of users changing the same database record within a file.

Interactive Application Program Example

The application program receives an average of 50 bytes of information from the user for each transaction. The application program returns an average of 750 bytes of information to the user for each transaction. The AS/400 system (for this example) takes an average of 0.25 second to process the transaction after it receives the data from the user. The user spends an average of 15 seconds between each transaction to think about the next transaction and to enter data.

The line or network is configured as follows:

- Line speed = 9600 bps
- Modem turnaround time = 0.1 second
- Line protocol = SDLC
- Frame size = 256 bytes
- Maximum number of outstanding frames = 7

The transaction proceeds as follows:

1. The user presses the Enter key to begin the transaction. There are 50 bytes of data, 9 bytes of SNA control information, and 6 bytes of SDLC control information, which results in 65 bytes of information. SDLC adds an additional 10 percent (6.5 bytes) for zero-bit-insertion. The total is now 71.5 bytes of information to be transmitted.
2. The user's station must wait 0.01 second before it is polled by the primary station.
3. When polled, the user's station takes 0.1 second of modem turnaround time.
4. The user's station transmits the data. There are 71.5 bytes or 572 bits of information to be transmitted (8 bits per byte). At 9600 bps this takes 0.06 second.
5. The AS/400 system receives the information and takes 0.25 second to process the response. The modem for the AS/400 system turns the line around during the 0.25 second of processing so the modem turnaround time is not included.
6. The AS/400 system is ready to transmit a 750-byte response to the user. The frame size is 256 bytes, so there are two frames of 256 bytes and one frame of 238 bytes (256 + 256 + 238 = 750). Each frame has an additional 9 bytes of SNA control information and 6 bytes of SDLC control information. SDLC adds an additional 10 percent for zero-bit-insertion.

   The first frame has 298 bytes or 2384 bits of information to be transmitted. This takes 0.25 second at 9600 bps. The second frame takes an additional 0.25 second to transmit. The third frame has 278 bytes of information and takes 0.23 second to transmit.
7. The user's station must now transmit an acknowledgment back to the primary station for the 750 bytes of information, but this occurs within the user-acceptable response time.

The entire transaction can be summarized as follows:

```
0.01 second + 0.1 second + 0.06 second + 0.25 second
+ 0.25 second + 0.25 second + 0.23 second
= 1.15 seconds total
```

If the line speed had been 2400 bps instead of 9600 bps, the transaction would have taken:

```
0.01 second + 0.1 second + 0.24 second + 0.25 second
+ 1.0 second + 1.0 second + 0.92 second
= 3.52 seconds total
```

Large File Transfer Example

The user wants to transfer a file with 51200 bytes of information.

The line or network is configured as follows:

- Line speed = 9600 bps
- Modem turnaround time = 0.1 second
- Line protocol = SDLC
- Frame size = 256 bytes
- Maximum number of outstanding frames = 7

The transaction proceeds as follows:

1. The AS/400 system segments the 51200 bytes of information into 200 frames of 256 bytes each. Each frame has an additional 9 bytes of SNA control information and 6 bytes of SDLC control information. This results in 271 bytes of information. SDLC adds an additional 10 percent (27 bytes) for zero-bit-insertion. The total is now 298 bytes of information to be transmitted. The first 7 frames are transmitted. Each frame takes 0.25 second.
2. The receiving station must now acknowledge the first seven frames because the maximum number of outstanding frames has been specified as 7. The receiving station’s modem takes 0.1 second to turn the line around so that the receiving station can transmit the acknowledgment.
3. A 6-byte acknowledgment is sent. SDLC zero-bit-insertion has no effect on this acknowledgment. The acknowledgment takes 0.005 second to transmit.
4. The modem for the AS/400 system now turns the line around so that the AS/400 system can transmit again. This takes 0.1 second.
5. This sequence repeats until all 200 frames have been transmitted and acknowledged.

The entire transaction is summarized as follows:

```
((0.25 second x 7) + 0.1 second + 0.005 second + 0.1 second)
x 28 + ((0.25 second x 4) + 0.1 second + 0.005 second)
= 55.8 seconds total
```

If the file had contained 1048576 bytes of information instead of 51200 bytes of information, the transaction would have taken:

```
((0.25 second x 7) + 0.1 second + 0.005 second + 0.1 second)
x 585 + (0.25 second + 0.1 second + 0.005 second)
= 19 minutes 4.3 seconds total
```

Again, these examples show only the line or network contribution to a transaction.

Estimating the Effects of Lines and Networks

The Performance Tools/400 licensed program (5763-PT1) provides a convenient means to estimate the effect that the range of application programs has on the line or network using the remote work stations.

Process Management

When doing system performance analysis, you must have the ability to observe the progress of batch work and the effect of heavy transaction loads on background batch processing. If the batch processing is falling behind schedule, you must make necessary adjustments in batch job priority, storage allocation, and possibly scheduling.

To measure the progress of a batch job in relationship to the other transaction loads on the system, use the batch job trace. The performance analysis functions measure the resource use for jobs going in and out of the long wait state during processing, such as a transaction processing job. However, for batch processing, the process does not usually enter the wait state unless there is a lock conflict or when a time slice value ends.

You should create trace records when a batch job begins, at user-defined trace intervals, and when the job ends. The trace record should be created at the start of a job and at intervals specified by a system variable. The interval time should be in seconds with a minimum value of 10. Only type A (automatic start) and type B (batch) are traced. To start the trace, the job checks for a nonzero value in the system variable.

Modem Considerations

This section uses the following terms:

- **Data terminal equipment (DTE)** is the part of a data link that sends data, receives data, and provides data communications control function according to protocols. This is usually a product such as a 5394 controller, a personal computer, or an AS/400 system.
- **Data circuit-terminating equipment (DCE)** is the equipment that provides all the functions required to establish, maintain, and end a connection. This equipment is installed at the premises of the user. It also includes the signal conversion and coding between the DTE and the communications line. The term DCE is synonymous with modem.
- **EIA-232** is a specification of the Electronic Industries Association (EIA) that defines the interface between DTE and DCE using serial binary data interchange. EIA-232 is roughly equivalent to V.24.
- **EIA-449** is a specification of the Electronic Industries Association (EIA) that defines the interface between DTE and DCE using serial binary data interchange. EIA-449 is roughly equivalent to V.36.
- **V.24**, **V.35**, and **V.36** are specifications of the International Telecommunications Union-Telecommunications (ITU-T) that define the list of definitions for interchange circuits between a DTE and a DCE.
- **X.21** is a specification of the ITU-T that defines the connection of a DTE to an X.21 (public digital data) network.
- **X.25** is a specification of the ITU-T that defines the interface to an X.25 (packet-switching) data network.

Following are considerations for choosing a modem. These considerations may also affect your choice of a communications line.

Aggregate Line Speeds

The **aggregate line speed** is the maximum possible speed that data can be transmitted using a communications I/O adapter on the AS/400 system. Aggregate line speed is determined using the sum of the speeds of the communications lines attached to the communications I/O adapter on the AS/400 system. You should consider the aggregate line speed allowed on a communications I/O adapter on the AS/400 system when you plan your communications configuration.

The environments you can consider are:

Table 16. Aggregate Line Speed

	IOP/IOA Types		
Protocols	**2623 IOP**	**2666 IOP**	**2699/2720/2721 IOA**
V.24 Async.	19.2K	N/A	115.2K
V.24 other protocols	19.2K	N/A	19.2K
V.35 SDLC	640K	2048K	2048K
V.35 X.25	64K	640K	640K
V.35 Frame Relay	N/A	2048K	2048K
V.35 Bisync.	64K	N/A	64K
V.36[1,2]	N/A	2048K	2048K
X.21[1]	64K	2048K	2048K

Notes:

1. X.25 is limited to 640K on 2666 and 2669/2720/2721.
2. Bisync. support, where available, is limited to 64K.

With X.25 the line speed should be doubled to determine the aggregate line speed.

Be certain that your modems support the aggregate line speed for the environment that you choose.

For more information on aggregate line speed, see Chapter 7. Communications Subsystem Controller Storage and Aggregate Line Speed, and for speed calculation examples, see “Line Speed Examples” on page 197.

Nonswitched vs. Switched Lines

When deciding between switched lines and nonswitched lines, consider the modem turnaround time. This is the amount of time required for a station on the line to stop receiving data and to begin transmitting data. Nonswitched lines generally have little modem turnaround time because they normally have four wires. These four wires can be used so two wires are always ready to transmit data and the other two wires are always ready to receive. Switched lines, however, can have a significant amount of modem turnaround time because they usually have two wires.

Typical modem turnaround times for switched lines range from 0.1 second to 0.5 second, depending on the modem and the quality of the line connection. This is especially important for interactive application programs if a station alternates frequently between sending and receiving data. In some cases, this can be many times for the same transaction. Batch application programs slow down because most line protocols require multiple transitions from sending to receiving during the transfer to ensure data integrity.

Duplex vs. Half-Duplex Support

Duplex support is usually applicable for nonswitched lines or networks because they have four wires available. It takes advantage of the two-way nature of these lines. One set of wires is always conditioned to receive while the other set of wires is always conditioned to transmit. There is almost no modem turnaround time with duplex support. Modem turnaround time is also related to the use of nonswitched or switched modems.

Note: Synchronous data link control (SDLC) can alternately receive or transmit on duplex lines. SDLC does not support duplex operation in which both receiving and transmitting can occur simultaneously.

Switched lines usually use half-duplex support except for the following:

- V.34 networks
- X.21 networks
- X.25 networks
- Modems that use the V.32bis modulation scheme

Be certain that your modems support the type of duplex operations you want to use.

Diagnostic Capability

Many modems have diagnostic capabilities. The diagnostic capabilities of a modem are important when doing problem analysis in a communications network. Diagnostic capabilities include **Link Problem Determination Aid** (LPDA-1 and LPDA-2 for IBM modems), a set of commands used for operating modems and diagnosing problems. For information on the specific diagnostic capabilities for a modem, consult the documentation for the mode.

APPC Data Compression

Data compression at the session level reduces the amount of data sent across a communications line. It can increase the throughput on slower lines. It can reduce the cost per bit on expensive lines. However, data compression also uses processing unit cycles. It can actually reduce throughput on very fast lines, which can send the data faster than the processing unit can compress it. Data compression varies in its effectiveness depending on the content of the data. For example, data compression is more effective on text than on binary data.

You can use APPC data compression between any two systems that support APPC and data compression, including APPC over TCP/IP configurations.

You can set up APPC to do compression in various ways. APPC can compress the outbound data, the inbound data, or both. You can select from two different compression algorithms and three variations of one of the compression algorithms.

The compression algorithms are:

- Run-length encoding (RLE)
- Adaptive dictionary-based compression

  Note: Adaptive dictionary-based compression is a dynamic compression algorithm, similar to Lempel-Ziv, that compresses previously seen strings to 9-, 10-, and 12-bit codes. This algorithm is referred to as LZ.

  Three levels of LZ compression are supported:

  – LZ9
  – LZ10
  – LZ12

You can specify the data compression (DTACPR) network attribute to set the system strategy for APPC data compression. The system can do the following data compression operations for APPC sessions:

- Require
- Request based on line speed
- Request regardless of line speed
- Allow
- Disallow

The DTACPR network attribute may be overridden by its corresponding parameter in a mode description.

APPN network nodes serving as an intermediate node can request data compression. The request is based on line speed.

When deciding how to configure your system for APPC session-level data compression, you should consider the following:

- Line speed
- Processing unit use
- Line charges
- Line use
- Intermediate node compression requests
- Specialized modes
- Type of data

Additional information and examples of data compression can be found in the *APPC Programming* book.

Protocol Considerations

Synchronous Data Link Control (SDLC) Considerations

SDLC is the most commonly used AS/400 data link protocol. It is supported by a number of devices and application programs and is compatible with the Systems Network Architecture (SNA). The following are performance considerations when using SDLC.

Frame Size

The AS/400 support for SDLC can use a range of frame sizes up to 2057 bytes. The frame size is specified with the maximum frame size (MAXFRAME) parameter in the line and controller descriptions. Usually, the larger the frame size used, the better the performance.

In general, the jobs that most frequently send data from the AS/400 system achieve better performance. Application design, job priority, and work station operator activity influence the frequency of data being available to send to the remote controller. Also, the system will favor output over polling for new input. For more information on polling, see “Polling” on page 205.

Given a specific line speed, and that data is being transmitted or received, the following have the greatest effect on performance for a particular controller:

- Controller frame size (MAXFRAME parameter on the controller description)
- Number of frames that can be sent to a controller without turning the line around (MODULUS and MAXOUT parameters on the line description)
- Number of frames that can be sent to a controller before communicating with another controller on the same line (POLLLMT parameter on the controller description)

Maximum Length of Request/Response Unit: Device description or mode description request/response unit (RU) length (MAXLENRU parameter) and pacing values are important when considering performance. Examples of such pacing values are INPACING and OUTPACING parameters on the CRTMODD command and the PACING parameter on the CRTDEVPRT command. These parameters are at the SNA-attached device or session level rather than at the controller SDLC and link level.

Pacing determines the number of RUs (windows) that can be sent or received before a pacing response indicates that additional RUs can be sent. Small RU length and pacing values may cause short frames or fewer frames to be sent than were specified.

When communicating with a host System/370 controller, NCP/VTAM parameters generally override AS/400 SNA RU length and pacing values for all device descriptions except APPC descriptions that specify LOCADR(00).

If short amounts of data are exchanged with devices or sessions on a controller, the effects of frame size, the number of frames sent, RU length, and pacing values on performance are minimal.

Make the maximum SNA RU size as close as possible to a multiple of the frame size, less the length of the transmission header (TH) and the response header (RH). If you do choose to make the RU size larger than the frame size, use an RU size that is slightly less than a multiple of the frame size. Avoid the situation where the RU size divided by the frame size results in an integer and a small remainder. This small remainder requires an additional small frame to be sent for each RU, which results in additional overhead.

The *CALC value for the MAXLENRU parameter automatically selects an efficient size for the SNA RU that is compatible with the frame size that you choose. Valid optional values for MAXLENRU are from 241 through 32768 except for Finance.

For Finance, MAXLENRU has an upper limit of 4096, but can be as low as 8. For Retail, valid values for MAXLENRU range from 247 to 1024. *CALC is the default.

For more information on pacing and RUs, see “Pacing” on page 222.

Large frame sizes may not work well for error-prone lines or networks. The effect of large frame sizes on different controllers on the error-free multipoint line must also be considered. If multiple controllers have sessions active concurrently, consideration must be given to the controllers with the smaller frame size.

Communicating with a controller with large frames, (2057 bytes, for example) can significantly improve performance if large amounts of data are exchanged. However, concurrent communications with controllers that support smaller frames sizes, such as 265 or 512 bytes, may be significantly degraded while the large frames are exchanged. The controller with the larger frames gets a larger portion of the available line capacity. For example, while sending a batch file to a controller using 2057-byte frames, interactive response time on a remote work station controller using 265-byte frames can be significantly degraded.

The effect of larger frames can be increased further if the maximum number of outstanding frames is increased beyond 7 and the controller POLLLMT value is set to greater than 0.

See “Maximum Number of Outstanding Frames” for more information on polling and frame size.

Maximum Number of Outstanding Frames

The maximum outstanding frames parameter determines how many SDLC frames can be sent before the receiving station must send an acknowledgment. The maximum for this parameter depends on the modulus (MODULUS) and maximum outstanding frames (MAXOUT) parameters that are part of the line description. If MODULUS is set to 8, the maximum that MAXOUT can be set to is 7. If MODULUS is set to 128, the maximum that MAXOUT can be set to is 28. MODULUS should usually be set to 8 except in special situations such as networks using satellite links. If MODULUS is 128, then MAXOUT must be greater than 7.

If you want to set MODULUS to 128, be certain that the remote system or device is also capable of supporting MODULUS set to 128.

As with larger frame sizes, a large maximum number of outstanding frames may not work well for error-prone lines or networks.

Duplex vs. Half-Duplex Support

Duplex support is applicable for both switched and nonswitched lines or networks. Duplex support takes advantage of the two-way nature of four-wire lines so that one set of wires is always conditioned to receive while the other set of wires is always conditioned to transmit. Although data is never sent and received simultaneously with SDLC, there is almost no modem turnaround time with duplex support. This is discussed further in the topic “Duplex vs. Half-Duplex Networks” on page 196. The DUPLEX parameter is part of the line description. Compatible modems are required.

Be certain that your modems support the type of duplex operations you want to use.

Polling

SDLC has primary and secondary stations. A **primary station** is the station responsible for controlling the data link. There must be only one primary station on a data link. All traffic over the data link is between the primary station and a secondary station. A **secondary station** is a data station that runs data link control functions as instructed by the primary station, interprets received commands, and generates responses for transmission. The primary station polls the secondary station. **Polling** is the process of sending data or control information to determine whether the secondary station is ready to send data. When a primary station polls a secondary station, the secondary station interprets the received commands and responds with data of its own; the secondary controller needs this poll before it can send back any data. Besides polling, a primary station is also responsible for starting recovery from most temporary line errors.

All SDLC polling is handled by the communications controller and is governed by parameters in both the line and controller descriptions.

The following are primary line description parameters that control SDLC polling:

- Fair polling timer (FAIRPLLTMR)
- Idle timer (IDLTMR)
- Connect poll timer (CNNPOLLTMR)*
- Frame retry limit (FRAMERTY)
- Poll cycle pause (POLLPAUSE)

where * indicates normal disconnect mode (NDM) polling only.

The following are primary controller description parameters that control SDLC polling:

- SDLC poll limit (POLLLMT)
- SDLC poll priority (POLLPRTY)
- SDLC connect poll retry (CNNPOLLRTY)*
- SDLC normal disconnect mode poll timer (NDMPOLLTMR)*

where * indicates normal disconnect mode (NDM) polling only.

Note: Connect poll retry is also specified on the line description for switched lines, but the value is only taken from the line description when the AS/400 system answers a call.

The following is a secondary line description parameter that contributes to SDLC polling performance:

- Poll response delay (POLLRSPDLY)

Because a negotiable (*NEG) station eventually takes either a primary or secondary station role, for this type of line you must configure all line and controller parameters for both primary and secondary roles.

As controllers are varied on, they are placed at the end of the **poll list**. This is a list of the controllers to which the primary station will transmit. “Polling the Next Station” on page 207 describes how the primary station chooses which controller to send to next.

A controller may be either in normal response mode (NRM), the mode for the sending and receiving of user data, or in normal disconnect mode (NDM), the mode a remote controller takes when it is first varied on and is offline or is turned off. Parameters that only affect polling in NDM are identified in the lists above.

Polling a Station That Does Not Respond

The idle timer (IDLTMR parameter) and the connect poll timer (CNNPOLLTMR parameter), used in error recovery and station vary-on processing, can affect response times for all controllers on the line. These parameters control how long the primary station waits after it polls a secondary controller and receives no response. While the primary station is waiting for a response, the line is idle, and the primary station cannot poll other stations on the line. Setting either parameter too high can result in poor performance for all controllers on the line.

The SDLC primary station uses 3 different combinations of timers and retry limits to poll the secondary station, depending on what it has received. Before the first valid response is received from the secondary, the connect poll timer and connect poll retry limit are used. Once a valid response is received from the secondary, the frame retry limit and connect poll timer are used even though the secondary may not have sent an unnumbered acknowledge (UA) frame and the connection is not in normal response mode (NRM) yet. When the UA frame is received from the secondary (the connection is in NRM), the frame retry limit and the idle timer are used.

- ***Idle timer:*** This parameter is used while in normal response mode and determines how fast the line recovers after a temporary error. If an error occurs after a station is polled and no response (or no final frame of a response) is received, the primary station waits for the idle timer to end (an idle time-out). This parameter is important if the line is noisy or has frequent temporary frame errors such as frame checks. Too low a value for the idle timer can cause unnecessary time-outs and responses to be transmitted again.

  The frame retry (FRAMERTY) parameter is used in conjunction with the idle timer in normal response mode. The frame retry limit determines the number of times the primary station will try a transmission to a remote station if consecutive temporary errors, such as idle time-outs, occur. Between retries to a station, a primary station continues to cycle through its poll list, transmitting to other stations. The frame retry limit should be large enough to allow the system to recover from temporary errors, such as those caused by line noise. You should note that too large a frame retry value can cause unnecessary delays in reporting permanent errors (such as inoperable remote system or link) and degrade performance for other stations while SDLC is busy retrying.
- ***Connect poll timer:*** This timer is used only in normal disconnect mode, when a controller is first varied on and is offline. Whenever a primary station polls a controller in this state and receives no response, it waits the duration of the connect poll timer before it resumes polling other stations.

  The connect poll retry (CNNPOLLRTY) parameter is used in conjunction with the connect poll timer in normal disconnect mode and determines if the primary station should poll an offline, normal disconnect mode station periodically or if it should stop polling the controller after a limited number of polls. The system default value (*CALC) allows seven retries on a switched line and unlimited (*NOMAX) retries for a nonswitched line.

Normal Disconnect Mode Polling Considerations

In addition to the CNNPOLLTMR and CNNPOLLRTY parameters discussed in “Polling a Station That Does Not Respond”, the SDLC normal disconnect mode poll timer (NDMPOLLTMR parameter) indicates how long the primary station must wait before sending a poll to a station in normal disconnect mode. While the timer is running, a controller that is offline, is not polled, even if its turn comes in the poll list. After the timer completes and the primary station finally reaches that station in the poll list, the station is polled. After the poll, the primary station waits for a response for the length of time specified by the connect poll timer.

In most cases if this timer is not large, response time problems will occur for other, active stations. If the normal disconnect mode poll timer is less than the time it takes to make one complete pass through the poll list, the inactive station is polled on every pass through the poll list. This can severely degrade performance of active normal response mode stations as the primary station polls controllers that are offline and then waits for responses. A large value for the NDM poll timer parameter only affects how long it takes a single, offline controller to become active after it is turned on. But a small value will have continuing, adverse effects for all active stations on the line until the offline stations are turned on.

For example, consider a multipoint line with eight controllers, a connect poll timer of 1.0 second, and a normal disconnect mode poll timer of 5.0 seconds. If all stations but one are active (in normal response mode), every 5.0 seconds there will be a 1.0 second wait while the primary station polls the single inactive station and waits for a response. During this 1.0 second period there is no activity on the line and all the active controllers wait.

An even worse situation, using the above example, is when only one of the eight stations is active. The line would be inactive for 7.0 seconds while the primary station polls the seven inactive stations and waits 1.0 second for each to respond. Then the primary station would be able to send only one productive poll to the single active station before the nonproductive poll sequence repeats. The result is that for every poll to the only active station, the system waits 7.0 seconds for the inactive stations.

In both examples, setting the normal disconnect mode poll timer to 90.0 seconds and decreasing the connect poll timer to 0.5 seconds would improve response time with only a small increase in the time to complete the vary on of the inactive controllers when they finally come online.

For large multipoint configurations in which many stations are not responding to their connection polls, it is necessary to coordinate the vary on of remote controllers. When many stations are not responding, the time for one pass through the poll list may grow large enough so that every NDM station is polled on every pass through the poll list. If this occurs, it is necessary to vary off some of the inactive stations to maintain adequate response time for active stations.

Polling the Next Station

If the primary station has no data for any of its stations, it sends supervisory polls to each station in the poll list in the order in which the stations were varied on. As soon as the primary station has output data for any one station and after it receives a final response from the last polled station, it interrupts sequential polling and starts sending information frames to that station. As long as it has data to send to that station, the primary station continues to transmit this data without polling other stations. If the primary station has data for more than one station, the out limit (OUTLIMIT parameter) value controls how many sequences of information frames (frames with user data) SDLC sends to a station before sending data to the next. For a complete description of how to use the out limit value, see “Repolling the Same Station” on page 209.

Thus, SDLC gives priority to the sending of output over polling for input. For overall system performance, it is important that SDLC send output data to remote stations as soon as possible after it becomes available. This sending of data reduces the number of short wait and short wait extended times. For more information on wait times, refer to the *Performance Tools for AS/400* .

The **fair poll timer** (FAIRPLLTMR) parameter specifies the length of time SDLC allows a few busy stations to monopolize the line before it sends polls to all stations. When the fair poll timer completes, and after a final response has been received at the end of the next set of frames determined by the OUTLIMIT parameter, the primary station stops sending output data. The primary station then determines which stations had not been polled since the last time the fair poll timer completed, or since the last time all stations were polled. The primary station then polls those stations. Thus, the fair poll timer ensures that no station will be denied polls because SDLC continuously sent data to other, busier stations. The fair poll timer is restarted when all stations in the poll list have been sent a poll.

Lowering the fair poll timer value makes polling more equitable by increasing the number of times every station is polled, but it also degrades performance for busy stations and can increase the length of time it takes to send data to a remote controller after a poll response. If some controllers are unusually busy and appear to be monopolizing the line at the expense of all other controllers, lowering the fair poll timer may improve the response times for the other controllers. (You can also put an unusually busy controller on a separate line to improve performance.) The system default for the fair poll timer is 15 seconds.

Priority Stations

If any station is configured with POLLPRTY(*YES), SDLC will send extra polls to that station in situations where a poll might otherwise be denied, as follows:

- When the primary station has data for stations other than the priority station, one extra poll will be sent to a priority station.
- When the fair poll timer completes, a priority station also receives an extra poll.
- When there is no data to send and the primary station is polling stations sequentially, it will poll a priority station twice as often as nonpriority stations.

Therefore, the POLLPRTY parameter can be used to give better response to some stations. This parameter should be used with caution. Designating one station as priority is almost equivalent, from a nonpriority station viewpoint, to adding another controller to the line. Nonpriority stations may experience increased response times.

Note: Except for the one extra poll allowed, priority stations do not take precedence in the poll list over stations with output data.

Polling after an Error

When the primary station polls a station that does not respond, it will not repoll that station until the fair poll timer completes, or until there are no stations with data to send. Thus, a station with output to send loses the priority it otherwise would have when it enters error recovery. Likewise, if a priority station does not respond to a poll it also loses its priority, until either the fair poll timer ends or until the primary station has no data for any other station.

Repolling the Same Station

When SDLC starts transmitting to a station, two values control how long SDLC sends polls or data to that station before polling the next station. These two values are the following:

- **Poll limit** (POLLLMT) parameter controls the number of additional polls SDLC will send to a station when that station responds with a number of frames equal to the maximum outstanding frames (MAXOUT) count. If a remote controller returns MAXOUT information frames to the primary station, it is possible the remote has other frames it could not send on its last response. By polling again, the primary station allows the secondary to send these frames.

  The following examples illustrate how the poll limit affects polling:

  – If the poll limit value is 1, the maximum outstanding frames count (MAXOUT) is 7, and a poll to a station results in receiving 7 information frames, the primary station sends one additional poll to that station. This allows the remote station to send more data to the system before SDLC goes to the next station in the poll list.

  – If the poll limit value is 1, MAXOUT is 7, and a poll to a station results in receiving 6 information frames, the primary station does not poll the same station again. Instead the primary station polls the next station in the poll list.

  – If poll limit is zero, MAXOUT is 7, and a poll to a station results in receiving 7 information frames, SDLC will poll the next station in the poll list.

  The value chosen for the poll limit in the controller description is also used to set the out limit value.

- **Out limit** (OUTLIMIT) parameter controls how many sequences of information frames (frames with user data) SDLC sends to a remote station before polling the next station. This is illustrated in the following examples:

  – If the out limit value is zero, the primary station sends one sequence of frames to a remote station. This sequence of frames consists of whatever data is available to send, from one to seven frames, up to the maximum outstanding frames allowable (MAXOUT parameter). The primary station waits for the response from the remote station, then sends data (if available) to the next station.

  – If the out limit value is one, the primary station sends one frame sequence (from one to seven information frames) to the remote station and receives the response as before. The primary station sends one extra frame sequence (if the output is ready) to the same remote station and waits for a second response. After the second response is received, the primary station sends data to the next station in the poll list.

    If the data is not available to the communications controller when the primary station is ready to send the second poll sequence, the primary station polls the next station in the poll list.

  The out limit value indicates whether SDLC should send extra sequences of information frames before going on to poll other stations in the poll list.

  An out limit value greater than zero should be considered if remote displays are being updated in segments, with long pauses between updates, or if remote printers are printing in short bursts.

  If SDLC has data for only one station, it will continue to transmit to that station regardless of what value the out limit parameter is set. If data becomes available for other stations or if the fair poll timer completes, SDLC starts transmitting to other stations.

Other Polling Parameters

The **poll cycle pause timer** (POLLPAUSE) parameter in the line description tells the primary station to delay after one complete pass through the poll list before cycling through the poll list again. This timer is only used when SDLC is not sending data to any station. Small values are recommended. If there are a number of lines physically attached to the same communications controller, however, and other lines are experiencing performance problems, increasing the poll cycle pause timer may give other tasks in the controller more time to run. The more stations on a line, the less effect this timer has on overall performance.

The **poll response delay timer** (POLLRSPDLY) parameter in the line description tells the secondary station how long to wait after it has been polled before it returns a response. A nonzero value should be used only if:

- Required by the modem
- The primary station cannot handle a fast response, or
- There is some other unusual requirement

Specifying a nonzero value will increase the time all other controllers on a multipoint line must wait before being serviced and should be avoided unless required. The value of the POLLRSPDLY parameter along with the internal delays and the time needed for increasing must be less than the idle timer on the remote system.

SDLC Overhead

SDLC causes three types of overhead:

- Zero-bit-insertion is the technique that allows any type of data to be sent over an SDLC line without interfering with the SDLC control information. The exact amount of overhead depends on the data that is being sent. Generally, zero-bit-insertion adds approximately 10 percent to the total number of bits that are transmitted.
- Each SDLC frame includes 6 SDLC control characters in addition to the data that is carried in the frame.
- Data sent on an SDLC line must be acknowledged. The frequency of these acknowledgments is determined by the maximum number of outstanding frames and the polling parameters.

IDLC Considerations

ISDN data link control (IDLC) is a protocol used for general data communications over an ISDN data channel. IDLC is compatible with SNA applications. IDLC is a balanced protocol, meaning it can send and receive data at the same time. Additional performance considerations can be found in the *ISDN Support* book.

Frame Size

The frame size is specified with the MAXFRAME parameter in the line and controller descriptions. The AS/400 support for IDLC allows a range of frame sizes up to 2064 bytes. In general, large frame sizes will provide better performance. However, large frame sizes may not work well for error-prone lines or networks, due to the longer time required to retransmit large frames.

Note: When running high-performance routing (HPR), the maximum frame size must be at least 768. If it is less than 768, the link will run APPN instead of HPR.

Maximum Length of Request/Response Unit: The maximum length of an SNA request/response unit (RU) can be specified with the MAXLENRU parameter in a mode description (APPC), or in some device descriptions. If you specify *CALC on the MAXLENRU parameter, an SNA RU size will automatically be selected that is compatible with the frame size chosen.

If you specify a value other than *CALC, choose an RU size that is slightly less than the frame size or a multiple of the frame size. The reason it should be slightly less is that there is some additional overhead added by the protocol being used. For example, SNA adds an additional 9 bytes of overhead by the time an RU is transmitted in a frame (3 bytes for the RH and 6 bytes for the TH). Therefore, to maximize performance for SNA, an RU size should be chosen so that the RU size plus 9 equals the frame size or a multiple of the frame size.

An RU size should not be chosen that is slightly greater than a multiple of the frame size, since this could result in an extra frame carrying a small amount of data being sent for each RU.

Window Size

Window size is the maximum number of IDLC frames that can be sent before an acknowledgment is required. Window size is specified with the IDLCWDWSIZ parameter in the line and controller descriptions. The AS/400 support for IDLC allows maximum window size of 31. In general, a large window size will provide better performance. However, as with large frame sizes, a large window size may not work well for error-prone lines or networks.

IDLC Overhead

IDLC uses zero-bit-insertion, similar to the SDLC protocol. Zero-bit-insertion adds approximately 10 percent to the total number of bits that are transmitted.

Each IDLC frame includes 6 or 7 IDLC control characters in addition to the data that is carried in the frame.

ATM Network Considerations

For LAN-attached lines, see “Local Area Network Considerations” on page 213, and *LAN, Frame-Relay and ATM Support* .

Frame Relay Network Considerations

Frame relay is a fast packet-switching protocol that can be used for wide area network (WAN)[2] interconnection of token-ring, Ethernet, or DDI networks. It can also be used as an alternative to X.25. Frame relay provides users with higher transmission speeds than X.25 and allows for bursts of data. These characteristics make frame relay well suited as a WAN backbone for carrying both SNA and TCP/IP LAN traffic from an AS/400 system to a bridge. Frame relay is compatible with SNA and is another data link that may be used to carry SNA data, similar to X.25.

[2] A data communications network designed to serve an area of hundreds or thousands of miles–for example, public and private packet-switching networks and national telephone networks.

The following are performance considerations if you are using frame relay support. Additional performance considerations are contained in the *LAN, Frame-Relay and ATM Support* book.

Frame Size

The frame size is specified in the MAXFRAME parameter in the line and controller descriptions. Larger frame sizes generally supply better performance. Large frame sizes may not work well for error-prone lines or networks because longer times are required to retransmit large frames when errors are encountered. When frame relay networks become congested, frames are discarded to alleviate the congestion. Therefore, large frame sizes may not work well on congested frame relay networks where many retries for transmissions are necessary.

The frame size is usually determined by the particular frame relay network to which the system is attached. The AS/400 system allows a maximum frame size up to 8182 bytes. Other frame sizes (265 through 8182 bytes) may be supported.

Note: When running high-performance routing (HPR), the maximum frame size must be at least 768. If it is less than 768, the link will run APPN instead of HPR.

See the *LAN, Frame-Relay and ATM Support* book for more information on selecting frame sizes.

Maximum Length of Request/Response Unit: The maximum length of an SNA request/response unit (RU) can be specified with the MAXLENRU parameter in a mode description (APPC). It may also be specified in the host device description. If you specify *CALC for the value of the MAXLENRU parameter, which is the suggested value, an SNA RU size automatically selected that is compatible with the frame size chosen.

When specifying values other than *CALC, use RU sizes that, when combined with your packet size and protocol, minimize communications costs. The RU size should be a multiple of the frame size, less the length of the transmission header (TH), the response header (RH), and the frame relay headers and route fields that are needed in the frame.

Avoid the situation where the RU size divided by the frame size supplies an integer and a small remainder. This small remainder requires an additional frame to be sent for each RU, which results in additional overhead. If the network charges are calculated based on the number of frames transmitted, additional costs may result.

Frame Relay Used for Bridging

Frame relay can be used to attach to a bridge, making the AS/400 system appear as if it were locally attached to the LANs that are connected to the bridge. The AS/400 system uses the IEEE 802.2 logical link control protocol for frame relay flow control. Therefore, the same controller description timer and retry parameters that are used for the LAN protocols also apply to frame relay. These parameters may need to be adjusted for frame relay. The speed of the frame relay medium is likely to be slower than typical LAN speeds. The frame relay throughput depends on the speed of the frame relay connection.

Token-ring and DDI Network: When using frame relay to bridge to a token-ring local area network, the maximum frame size is 8148 bytes. When using frame relay to bridge to a DDI local area network, the maximum frame size is 4444 bytes.

These frame sizes do not include frame relay headers or routing information. The frame sizes need to be reduced to allow for the frame relay and routing information fields. See the *LAN, Frame-Relay and ATM Support* book for more information on frame sizes.

Ethernet Network: When using frame relay to bridge to an Ethernet network, the maximum frame sizes are equivalent to those specified in the *LAN, Frame-Relay and ATM Support* .

Local Area Network Considerations

Local area network support on the AS/400 system offers a flexible means of connection for token-ring and Ethernet network users with much higher transmission speeds than are available from traditional telecommunications support. The following are performance considerations if you are using the local area network support. Additional performance considerations are contained in *LAN, Frame-Relay and ATM Support* book. For specific performance information regarding your configuration and work load, contact your account representative.

Frame Size

The frame size is specified with the MAXFRAME parameter in the network interface (NWI) line and controller descriptions. As with SDLC, the larger frame sizes generally supply better performance. The improved performance caused by larger frame sizes tends to be more noticeable with the local area network because of the faster media speed.

Note: When running high-performance routing (HPR), the maximum frame size must be at least 768. If it is less than 768, the link will run APPN instead of HPR.

Maximum Length of Request/Response Unit: The *CALC value for the MAXLENRU parameter automatically selects an efficient size for the SNA request/response unit (RU) that is compatible with the frame size you choose.

If you choose not to use the *CALC value, use a large RU size that is slightly less than a multiple of the frame size. Avoid the situation where the RU size divided by the frame size supplies an integer and a small remainder. This small remainder requires an additional frame to be sent for each RU, which results in additional overhead.

The frequency of errors on the local area network media tends to be much less than with traditional telecommunications lines. For this reason, there is much less chance of performance degradation due to error recovery when large frame sizes are used.

| ***Token-ring Network:*** The AS/400 support for token-ring networks, using an
| adapter that supports either 4 million bits per second (Mbps) or 16 Mbps, uses a
| range of frame sizes. These can vary from 4472 bytes for the 4 Mbps Token-Ring
| network, and 16393 bytes for the 16 Mbps Token-Ring network. The AS/400 support
| for token-ring networks, using an adapter that supports 4 Mbps only, uses a range
| of frame sizes up to 1994 bytes.

Ethernet Network: AS/400 Ethernet network support allows a frame size ranging from 265 to 1496 bytes for IEEE 802.3. For TCP/IP running over Ethernet Version

2, the maximum frame size is 1502. For SNA running over Ethernet Version 2, the maximum frame size is 1493. The maximum value provides the best performance.

Wireless Network: AS/400 wireless network support allows a frame size of 1496 bytes for IEEE 802.3. For SNA running over Ethernet Version 2, the maximum frame size is 1493. The maximum value provides the best performance.

Acknowledgment Frequency and Maximum Number of Outstanding Frames

The **acknowledgment frequency** is the number of incoming frames that a station can receive from another station on the local area network before an acknowledgment for those frames is sent. It is specified in the LANACKFRQ parameter in the controller description.

The maximum number of outstanding frames is the number of outgoing frames that a station is allowed to send to another station before an acknowledgment is required. This value is specified in the LANMAXOUT parameter on the controller description. Due to the nature of most local area networks, it is recommended that no more than seven frames be sent before an acknowledgment is required.

See the *LAN, Frame-Relay and ATM Support* book for more information about the values for LANACKFRQ and LANMAXOUT that offer the best performance.

Ethernet and Wireless Adapter: SNA uses the flow control of the IEEE 802.2 LLC protocol. TCP/IP also uses a form of flow control. When using these interfaces, frame discards only occur on heavily loaded networks.

Because those programs that rely on the User Datagram Protocol (UDP) interface use an unacknowledged service, it is more likely that discarding of frames can occur. This means that a single application with a single user can cause a burst of frames to the adapter that is great enough to cause discarding of frames. Applications being developed for this interface should keep this in mind.

If IEEE 802.3 support is used and SNA sessions experience excessive delay, the traffic directed at the overloaded adapter should be reduced. If SNA sessions are disconnected due to an excessive number of frames being discarded, you should reduce the amount of traffic or increase the LANRSPTMR.

Link Speed

The link speed (LINKSPEED) is a characteristic specified on the create line description commands. The link speed characteristic in a high-performance routing (HPR) network should accurately specify what the link is capable of running.

Wireless APPC Controller: When creating a wireless APPC controller (using the CRTLINWLS command) that is intended for use in running HPR, the recommended LINKSPEED parameter setting is 280000.

Local Area Network Overhead

The local area network overhead tends to be less noticeable because of the higher media speed. Local area network overhead includes the use of acknowledgments. This overhead can be minimized with the proper settings of timer parameters when configuring. Several other types of media limitations must also be considered in local area network environments. Because it is a shared media, it is important to manage the number of stations and the amount of transmission so that the local area network does not become overused or congested. For transmissions that use bridges to go from one network to another, throughput levels will be further reduced based on the capacity of the slowest bridge in the link.

Specifying *YES in the *Enable for TCP/IP only* field of the Ethernet Line description now results in improved TCP/IP performance in the input/output processor (IOP). When the *YES option is chosen, and the hardware requirements are met, special IOP microcode is loaded in the IOP.

Note: This mode only works only for the 2810 and 2809 IOPs when a 2838 (100 Mbps Ethernet LAN) is the only input/output adapter (IOA) in the IOP. For all other IOPs, this option has no affect. In this mode, protocols other than TCP/IP cannot be used with the IOP. This special microcode is optimized for maximum TCP/IP performance through the adapter.

X.25 Considerations

X.25 is an alternative protocol to SDLC and is compatible with SNA. X.25 is also used for general data communications over an ISDN B-channel. X.25 is capable of sending and receiving data simultaneously, which provides a significant performance advantage for those application programs that can make use of this feature. The following are performance considerations when using X.25. Additional performance considerations are contained in *X.25 Network Support* book.

Packet Size

The AS/400 support for X.25 allows a range of packet sizes up to 4096 bytes. The packet size is specified using the DFTPKTSIZE parameter in the line and controller descriptions and the MAXPKTSIZE parameter in the line description. As with SDLC and token-ring networks, larger packet sizes provide better performance. In general, it is preferable to run with the largest packet size supported by the X.25 network.

Make the maximum SNA request/response unit (RU) (MAXLENRU) size a multiple of the packet size, less the length of the SNA headers. Specifying *CALC as the value for the MAXLENRU parameter causes the system to select an efficient RU size. Avoid the situation where the RU size divided by the packet size supplies an integer and a small remainder. This small remainder requires an additional packet to be sent, which results in additional overhead.

Frame Size

The frame size is specified in the MAXFRAME parameter in the line and controller descriptions. The AS/400 support for X.25 allows frame sizes up to 4096 bytes. This frame size represents the maximum link protocol data unit size that can be sent or received.

Notes:

1. This parameter is independent from the high-level data link control (HDLC) frame size. In general, large frame sizes provide better performance, but require more of the resources available in the communications processor.
2. When running high-performance routing (HPR), the maximum frame size must be at least 768. If it is less than 768, the link will run APPN instead of HPR.

Maximum Length of Request Unit: If you specify the value *CALC (the default) for the MAXLENRU parameter in a mode (APPC) description or in device descriptions having this parameter, the AS/400 system then determines the most appropriate size for the request unit (RU) to be used for devices attached to an X.25 controller.

The *CALC value for the MAXLENRU parameter automatically selects an efficient size for the SNA RU that is compatible with the packet size that you choose.

For Finance and Retail, the value calculated by the *CALC value for the MAXLENRU parameter depends on whether there is a frame size indicated on the Exchange ID (XID) received. If a frame size is indicated by the XID, the value is calculated from whichever frame size is smaller, the frame size from the XID or the frame size on the controller description. If no frame size is indicated on the XID, the value is calculated from whichever frame size is smaller, the frame size specified in the controller description, or the frame size specified in the line description.

When specifying values other than *CALC, use RU sizes that, when combined with your packet size and protocol, minimize communications costs. The RU size should be a multiple of the packet size, less the length of the transmission header (TH), the response header (RH), and the logical link header (LLH).

For example, assume that your subscription requires 128-byte packets and the protocol to be used between the DTEs is enhanced logical link control (ELLC), which requires 6 bytes for the logical link control (LLC) header. If you specify an RU value of 241 (241 bytes of data), this length of data combined with 9 bytes (a common value) of SNA headers and the 6 bytes for the LLC header fills exactly two packets. Specifying a larger RU size creates the need for a third packet to accommodate the excess amount over 256 bytes contained in two packets.

Note: Valid optional values for X.25 (ELLC protocol) are 241, 497, 1009, 2033, 4081. Valid optional values for X.25 (QLLC protocol) are 247, 503, 1015, 2039, and 4087. The chart of common maximum RU sizes under the MAXLENRU parameter of the Create Mode Description (CRTMODD) command appears in the *CL Reference (Abridged)* book.

Large packet sizes may not work well for error-prone lines or networks. The large packets have a higher probability for errors in this environment and take longer to transmit again.

Window Size

The X.25 window size is similar to the maximum number of outstanding frames parameter used by SDLC and the token-ring networks. The maximum that this parameter can be set to depends on the MODULUS and DFTWDWSIZE parameters that are part of the line description. If MODULUS is set to 8, the largest size that DFTWDWSIZE can be set to is 7. If MODULUS is set to 128, the largest size that DFTWDWSIZE can be set to is 15. Usually, making this number as large as possible results in the best performance.

As with packet size, this must match the configuration of the X.25 network.

X.25 Overhead

Like SDLC, X.25 uses zero-bit-insertion and framing characters around the data. X.25 sends 9 control characters in Modulo-8 and 10 control characters in Modulo-128 in addition to the data. Although X.25 is not a polled protocol like SDLC, it requires acknowledgment frames, which result in a similar cost.

Enhanced logical link control (ELLC) support also adds an additional 6 control characters to each frame for end-to-end data integrity. This support causes additional acknowledgment frames to be sent.

LLC2 support adds 23 control characters to each frame. Also for LLC2, additional padding characters are inserted into the frame prior to transmission for control character escape formatting. The number of padding characters is variable and is based on the user data.

X.25 offers a significant performance advantage for those application programs that can use the capability of sending and receiving data at the same time.

X.25 Throughput Overhead

ELLC support adds overhead to the IOP processing in that it must calculate a FCS for each transmitted and received frame.

LLC2 adds even more overhead to IOP processing. In addition to FCS, LLC2 must examine each transmitted and received byte for the possibility of control escape sequences.

Binary Synchronous Communications Considerations

The AS/400 system provides support for the many devices that continue to communicate with BSC. It is not compatible with IBM SNA. The following are performance considerations when using BSC.

Buffer Size

The AS/400 support for BSC uses a range of buffer sizes up to 8192 bytes. The buffer size is determined by the MAXBUFFER parameter in the line description. Usually, the larger the buffer size you use, the more the performance improves.

Large buffer sizes may not work well for error-prone lines or networks. The large buffers have a higher probability for errors in this environment and take longer to transmit again.

Data Compression and Blank Truncation

BSC data compression and blank truncation significantly reduce the amount of data transmitted on the line when the data contains large numbers of blanks. This reduction can be significant for text and forms data that usually contain many blanks. Data compression has little effect if the data contains very few or no blanks. BSC data compression is controlled with the DTACPR parameter in the device description. BSC blank truncation (all blanks are to the right of the data) is controlled by the truncation (TRUNC) parameter of the device description.

Duplex vs. Half-Duplex Support

Duplex support is usually applicable for nonswitched line or networks because they have four wires available. It takes advantage of the two-way nature of these lines so that one set of wires is always conditioned to receive while the other set of wires is always conditioned to transmit. Although data is never sent and received simultaneously with BSC, there is almost no modem turnaround time with duplex support. This is discussed further in the topic “Duplex vs. Half-Duplex Networks” on page 196.

Switched lines have only two wires available and usually use half-duplex support unless a special duplex switched modem is used.

Selection of duplex or half-duplex is made with the DUPLEX parameter in the line description.

BSC Overhead

BSC causes three types of overhead:

- BSC uses control characters for marking the beginning and ending of buffers, blocks, and records within a block. It also uses control characters to ensure data integrity. The exact number of control characters varies depending on how the support is used.
- Each data buffer causes an acknowledgment to be sent to the sending station from the receiving station.
- Transparency is an option used if the data contains values that are the same as BSC control characters, which is possible if the data is below hexadecimal 40. Additional control characters are inserted in the data to identify which of the data looks like BSC control characters. Transparency is controlled by the TRNSPY parameter in the device description.

Asynchronous Communications Considerations

The AS/400 system provides support for many devices, application programs, and services using asynchronous communications support. Asynchronous communications is not compatible with Systems Network Architecture (SNA). The performance of this support depends on the application program or service with which it is used and the speed of the line or network used. Consider the following when using asynchronous communications:

Logical Record Size

A logical record must not exceed 4096 bytes and is determined by one of the following:

Output
An output request will be one logical record.

Input
If your application program makes use of end-of-record (EOR) processing, a logical record will consist of all data received up to and including the EOR character and any trailing characters. The EOR table is specified during the line configuration.

All data received before the idle timer ends is considered a record.

The input data buffer in the communications adapter is full, and the entire input buffer is considered a record.

Buffer Size

The OS/400 asynchronous communications support allows you to specify a buffer ranging in size from 128 to 4096 bytes. The MAXBUFFER parameter of the line description determines the buffer size. The buffer size you configure should be the largest logical record your program uses.

Note: If you use file transfer support (FTS), the MAXBUFFER value must be a minimum of 896.

Asynchronous Communications Overhead

The amount of overhead to send each character on the line can be reduced by changing the definition of a character in the line description. For example, if you have configured 8 bits even parity and 2 stop bits, the total number of bits sent on the line would be 12. One start bit and at least one stop bit are always sent. If instead you configured 8 data bits, 1 stop bit, and no parity, the total number of bits sent on the line would be 10. This would reduce the overhead by 17 percent. The remote system must accept the character format sent by the system.

Duplex vs. Half-Duplex Support

See “Duplex vs. Half-Duplex Networks” on page 196 for more information about modem turnaround. Duplex transmission is the ability to send and receive data simultaneously. Half-duplex transmission allows both send and receive operations but not simultaneously. Selection of duplex and half-duplex transmission is made with the DUPLEX parameter of the line description. Asynchronous communications has two types of overhead:

- Using a start and stop bit for each character sent.
- Using a parity bit when configured. If not configured, no parity bit is sent for the high end, and a null bit is sent for the low end.

PPP Considerations

The Point-to-Point Protocol (PPP) provides a standard method for transporting multi-protocol datagrams over point-to-point links. It is described by Internet Engineering Task Force (IETF) Request for Comments (RFC) 1661, and other related RFCs. On the AS/400, PPP is used with TCP/IP applications.

Frame Size

The frame size is specified with the MAXFRAME parameter in the line description (or in the connection profile if using Operations Navigator). The AS/400 support for PPP allows a range of frame sizes from 1500 to 4096 bytes. In general, large frame sizes will provide better performance. However, due to the longer time required to retransmit, large frame sizes may not work well for error prone lines or networks.

General Programming Support Considerations

The AS/400 system has a wide range of programming support for application programs in a communications environment. This topic discusses some of the general performance considerations when using this support. Refer to the product documentation for the specific support you are using for more detail on how to apply these considerations.

Blocking

Usually, the AS/400 system works more efficiently with a few large blocks of data than it does with a large number of small blocks. The total number of blocks that must be processed affects the AS/400 system more than does the size of the blocks. Therefore, work with larger blocks whenever possible.

Large blocks are ideal for batch file transfer of all the records in a file. In some application programs, transmitting journal entries at different time intervals can result in records being placed in the buffer but not transmitted to the remote system. If this happens, a read operation is necessary to clear the buffer.

For application programs using SNA support, the block size is specified as the request or response unit (RU) maximum length.

When communicating to a host System/370 using SNA 3270 device emulation, SNA remote job entry, SNA distributed host command facility, SNA distributed systems node executive, or SNA upline facility support, the RU length is controlled by the host system parameters.

When communicating to a remote work station, that particular device determines the size of the RU. When communicating to a remote system using APPC, the RU value is negotiated between the two systems. The MAXLENRU parameter on the CRTMODD command specifies the RU length requested by the AS/400 system. The lesser of each system’s corresponding values is used. For SNA to a host system, the RU value is determined by the APPC parameters of the host system.

The block size is specified with the BLKLEN parameter in the device description of BSC application programs. Refer to the specific product manuals to determine any unique restrictions of the product.

Blocking can be applied to application program design as well. For example, when sequentially processing a series of records from a remote system, it is more efficient to transfer many records from the remote system in one transfer and then process them than it is to transfer each record individually and process it as it arrives. As discussed in the topic on “Network and Line Considerations” on page 188, larger block sizes also reduce protocol overhead and the number of times the line must be turned around.

While blocking improves batch file transfer, some application designs can result in an unfilled block of records that remain in the sending system. If entire files are transmitted, normal application program ending causes a “short block” to be transmitted. However, if subsets of records are transmitted (for example, sending database journal entries to a remote system), the subset of records may be small enough to cause a short block of records not to be transmitted. Such an application must make an allowance for such a condition by starting an operation that forces the short block to be sent. For example, an APPC program can force the short block to be sent by using the following DDS keyword functions:

- CONFIRM
- FRCDTA
- INVITE
- ALWWRT

Refer to the appropriate communications programmer’s book for additional information. The titles and associated order numbers and a brief description are listed in the “AS/400 Books” on page 253.

File Placement within a Network

When sharing files between systems and users in a network, it is important to place the files on the system where they will be used the most.

Identify which system has primary responsibility for file maintenance. In all cases of application programs that use multiple systems, it is best if only one system is responsible for file maintenance. If an application program maintains a file through exclusive (nonshared) processing, best performance is seen if that file resides on the same system as the application program.

If file maintenance is shared across systems, the best performance is seen if the file is placed on the system with the largest percentage of file update, add, and delete operations.

In interactive application programs, display station pass-through should be considered if the amount of data to be transferred to the remote display station is significantly less than the amount of file data that would be transferred to run the application program on the remote system.

Printer Performance

The speed at which a remotely attached printer runs is a function with many variables. The printer cannot run faster than its rated speed or faster than the line can provide data. For example, assuming 132 characters (bytes) per line and an 8-bit byte, a printer rated at 1100 lines per minute can print at 19360 bits per second. Therefore, a data link rate greater than this value is needed to keep up with the printer. Contrast this with an 80-character-per-second printer that has a rate of approximately 640 bits per second. The slower of these two considerations, either the line speed and or printer speed, establishes an upper limit of speed that the printer can possibly run.

However, that upper limit is never reached. Data link quality, the communications protocol used, the type of controller used, the speed of the link between the controller and printer, and the number of devices and controllers competing for resources can all contribute to keep you from reaching the theoretical upper limit. Some of these limits can be eliminated by using a line that is fast enough and by using a dedicated environment. If lines are not error free, data will be sent repeatedly, which decreases line capacity and slows the printer.

At the data link level, the AS/400 system can use SDLC, X.25, token-ring network, frame relay, IDLC, DDI, or Ethernet network protocols to send the data on the communications line to the remote work station controller. These protocols provide data accuracy and error recovery. The cost of these protocols can be considered relatively insignificant on high quality lines, but can become very significant with poor quality lines.

The AS/400 system uses SNA, along with the data link protocol, to transmit and receive application data and control information to and from a work station controller. All data is transmitted in path information units (PIUs). A PIU is 261 bytes for 5294 controllers, 517 bytes for the 5394 controller, and 521 bytes for the 5494 controllers. On each of these controllers, each PIU contains up to 256, 512, or 503 bytes of application program data or control information, respectively. SNA also provides support for data control of the speed and timing.

How the controller performs its buffer management directly affects the printer throughput. If the controller waits until all the data is sent to the printer before sending a pacing response to the host, the printer may have no work for a significant amount of time. This becomes even more apparent with a high-speed printer on a slow communications line. If the controller allocates twice as many buffers as the window size, it may send a pacing response whenever it receives the first frame of the window. However, even if you use this method, a high-speed printer may have to wait if the printer can process all the buffered data faster than the host or line can supply it.

The 5294 controller sends a pacing response after two or three frames if the window size is accepted by the printer; the 3274 sends its pacing response whenever it receives the first frame in the window. The 5394 and 5494 controllers send a pacing response after the number of frames specified by the AS/400 device description (in the range of 1 through 7) is sent. If more than three printers are attached to the 5394 and the pacing value was specified at the AS/400 system as higher than three, a pacing value of three is used for the fourth and higher printers. Both printers respond with `receiver not ready` at the data link control layer if the printer cannot print the data faster than the host can supply it. Therefore, data link rate and printer speed factors are important if printing in a dedicated environment. Pacing and system expense become significant if printer and data link speeds approach the internal processing speed of the host or work station controller.

Pacing

Pacing is required if there is a possibility of overflowing data buffers internal to the controller or the host system. This usually occurs if the controller or host must pass the data to a device operating at a slow speed. If the host system receives a pacing response, it sends more data frames up to the window size to the controller.

Pacing applies only to application programs using SNA support. APPC over TCP/IP applications use a fixed pacing mechanism between APPC and TCP/IP. It specifies the number of RUs that can be sent before a response or acknowledgment is required from the receiving device. A large pacing size generally provides better performance for a single application program on a line. Sometimes the pacing size must be balanced between interactive and batch application programs on the same line or network. If the pacing size is too large for the batch application program, it can cause the line to be busy for long periods of time, and the interactive application programs experience long and inconsistent response times. Interactive response time improves as the pacing value used by the batch job approaches one.

When communicating to the host System/370 using the following, the pacing values used are controlled by the host system parameters:

- SNA 3270 device emulation
- SNA remote job entry
- SNA distributed host command facility
- SNA distributed systems node executive
- SNA upline facility support
- SPLS
- Device descriptions with LOCADR specified as nonzero

For remote 5250, 3270-attached devices, display data is not paced and printer data defaults to a pacing value of seven. If printers are attached to a 5294 controller or 5251 Model 11 device, pacing is three. For up to three printers attached to a 5394 or 5494 controller, allow the pacing value of seven; each additional printer has a pacing value of three. In cases where a high-speed printer is used or satellite communications is involved, the default value of seven for the printer pacing value (PACING on the printer device description) may be required to achieve acceptable throughput but may affect interactive response time for the display device. Pacing to retail devices defaults to a value of seven.

If communicating to a remote system using APPC, the pacing values used are sometimes negotiated between the two systems. The INPACING and OUTPACING parameters on the CRTMODD command specify the pacing values requested by the AS/400 system. The lesser of each system's corresponding values is used. If these values are not negotiated, both APPC and APPN support on the AS/400 system provide adaptive pacing, which automatically controls how data is arranged and managed in the network. For more information on pacing and the APPN support, see the APPC, APPN, and HPR articles in the *AS/400e Information Center*.

Chapter 7. Communications Subsystem Controller Storage and Aggregate Line Speed

When configuring a communications controller, it is important to consider both subsystem storage and aggregate line speed. Subsystem storage is the amount of storage available on the communications controller. Aggregate line speed is the sum of the speeds of individual lines attached to the communications controller. These factors affect the performance of the communications controller.

Another factor to consider is the maximum line speed accommodated by the communications interface. Selecting line speeds for individual lines based on that maximum enables you to maximize controller performance. The communications controllers discussed in this chapter support the following non-LAN interface types:

- ISDN channels
- V.24 or EIA-232 communications lines
- V.35 communications lines
- V.36 or EIA-449 lines
- X.21 communications lines

Note: The discussion of aggregate line speed calculation in this chapter does not apply to DDI, frame relay, or Ethernet network adapters. You cannot change the line speeds on DDI and Ethernet network adapters.

Even though a communications controller has the storage and capacity to support a given aggregate line speed, the system unit may not have the capacity to utilize that controller configuration. See your IBM service representative to discuss overall performance capacity and workload characteristics.

Maximum Aggregate Line Speeds

The 9406 System Unit, the 9404 System Unit, the 9402 System Unit, and the 9401 System Unit have different communications limitations.

2623/4xx 5xx MFIOP Controllers

The communications controllers in your system are a limited resource. For effective communications, it is important to ensure that you have sufficient communications controller storage and processing capability. Figure 34 on page 226 shows possible combinations for two communications subsystems. The aggregate line speeds shown in Figure 34 on page 226 should not be exceeded.

Note: Figure 34 on page 226, Table 17 on page 226, and Table 18 on page 227 refer to 4xx and 5xx, as well as multiline communications controllers.

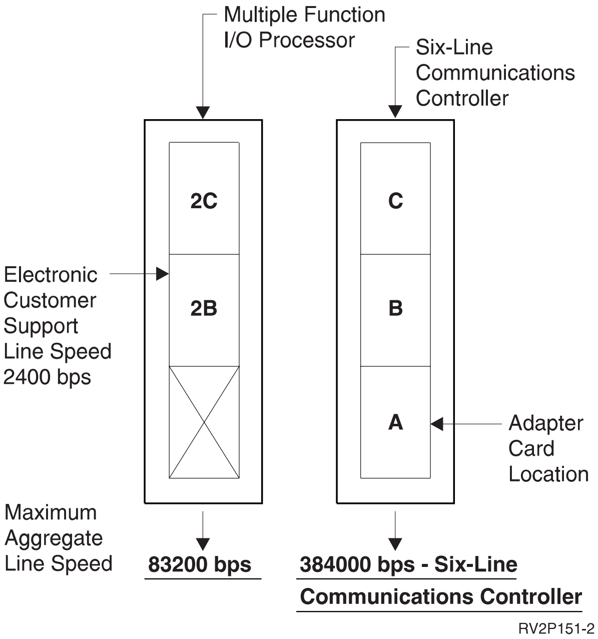


Figure 34. Controllers Possible Combination

Table 17 shows the capacity of each controller. It does not show all combinations, but can be used as an example.

Table 17. Line Speed Examples for the 9406 System Unit

Protocol	**Example of Line Speed in bps**	**Number of Lines per Six-Line Communications Controller**	**Number of Lines per 4xx and 5xx Multiple Function I/O Processor**[1]
Asynchronous	9600 or less	6	1
Asynchronous (half-duplex)	19200	6	1
Asynchronous (duplex)	19200	6	1
BSC	19200	6	1
BSC	>19200 through 64000	3	0
SDLC	19200	6	1
ISDN	64000	4[2]	0
SDLC	>19200 through 64000	6	1
SDLC	>64000 through 384000	3[3]	0
SDLC	>384000 through 512000	2[3]	0
SDLC	>512000 through 640000	1[3]	0
X.25	9600 or less	6	1
X.25	19200	6	1
X.25	48000	3	0

Table 17. Line Speed Examples for the 9406 System Unit (continued)

Protocol	Example of Line Speed in bps	Number of Lines per Six-Line Communications Controller	Number of Lines per 4xx and 5xx Multiple Function I/O Processor[1]
X.25	64000	3	0
X.25[4]	256000	1	0

Notes:

[1] This is the number of lines in addition to the SDLC line for electronic customer support.

[2] For a Northern Telecom DMS100 service, only one B-channel can be used for each ISDN adapter. Therefore, a six-line communications controller with two ISDN adapters supports two B-channels.

[3] This is the number of lines allowed if all lines on a controller are SDLC.

[4] If X.25 is run over 64000, no other lines are allowed on the IOP.

Line speeds are partially determined by the interface that is selected. Some protocols run only on particular interfaces. Table 18 shows the possible controller and protocol combinations by interface:

Table 18. Protocol and Interface Combinations for the 9406 System Unit

Interface	Communications Controller, Processor, or Adapter	Protocol	Maximum Line Speed (bps)
X.21	Six-Line	SDLC or X.25	64000
X.21	Multiple Function IOP	X.25	19200
X.21	Multiple Function IOP	SDLC	64000
X.21	2666	SDLC, Frame Relay	2048000
X.21	2666	X.25	640000
V.35	Six-Line	BSC	64000
V.35	Six-Line	X.25	256000
V.35	Six-Line	SDLC	640000
V.35	Multiple Function IOP	SDLC	64000
V.35	2666	SDLC, Frame Relay	20480001[1]
V.35	2666	X.25	640000[1]
EIA-232/V.24	Six-Line	SDLC, X.25, BSC, or Asynchronous	19200
EIA-232/V.24	Multiple Function IOP	SDLC, X.25, BSC, or Asynchronous	19200
EIA-449/V.36	2666	SDLC, Frame Relay	2048000[2]
EIA-449/V.36	2666	X.25	6400002
ISDN	Six-Line	IDLC or X.25	64000

Table 18. Protocol and Interface Combinations for the 9406 System Unit (continued)

Interface	Communications Controller, Processor, or Adapter	Protocol	Maximum Line Speed (bps)
Notes: 1. This is the maximum line speed supported only on the 20-foot (6-meter) cable. The line speed is restricted to 64000 on longer cables. 2. This is the maximum line speed supported on the 20-foot (6-meter) cable. The line speed is restricted to 64000 on longer cables unless the DCE supports a looped clock option and clock is set to *LOOP.			

Calculating Aggregate Line Speed for the 9406 System Unit

During your planning phase, you completed a form in the *Physical Planning Reference* book that lists your communications hardware. To calculate the aggregate line speed correctly, use the form and complete the following steps:

1. Write down the line speed for each line.
2. Calculate the aggregate line speed for the controller by summing the speeds of all the lines attached to the controller. Add the speeds of duplex asynchronous or X.25 lines twice.

   See the *Communications Configuration* book to determine which lines run on which controller.
3. Make sure the total per 2623/MFIOP 4xx or 5xx controller does not exceed the maximum aggregate line speed for that controller type.

   Determine the maximum aggregate line speed for a 2623 controller dedicated to SDLC lines greater than 64000 bps from Table 17 on page 226.

For general system performance considerations, the aggregate communications speed across the system must be considered.

Calculating Communications Subsystem Storage (2623 and 4xx/5xx MFIOP)

The communications subsystem provides flexible support for a number of adapters running various protocols. Each line on a controller uses controller storage. If you are using only the communications subsystem in the multifunction input/output processor (IOP) or ISDN, you do not need to calculate the subsystem storage discussed in this chapter.

In addition to the storage used in the controller for protocol code and operating system code, the following configuration parameters cause additional storage to be used:

MAXFRAME
The maximum frame size

MAXOUT
The maximum number of outstanding or unacknowledged frames

MAXBUFFER
The maximum buffer or message size

MAXCTL
The maximum number of controllers

See the *Communications Configuration* book to determine which lines are attached to each controller.

Use the following tables to calculate the storage per subsystem:

1. Table 20 on page 232 should always be used.

   Select the controller (Multiline Communications Controller, Three-Line Communications Controller, or Six-Line Communications Controller) and the protocol or protocols you use for the subsystem.

2. Table 19 on page 230 always should be used.

   Select the configuration parameters for the line or lines you use for the subsystem.

3. Table 21 on page 232 should be used only with an SDLC port group and short-hold mode.

   Add 51KB if you use an SDLC port group with short-hold mode for the subsystem.

4. Table 22 on page 232 should be used with SDLC controllers.

   Select the configuration parameters for the line or lines you use for the subsystem.

5. Table 23 on page 232 should always be used.

   Add the totals from the previous tables to determine the total storage used for the subsystem.

Following is an example to calculate the subsystem storage for a 9404 or 9402 Three-Line Communications Controller with:

- An X.25 line with 16 virtual circuits
- An SDLC line configured with the following parameters:
  - MAXFRAME=521
  - MAXOUT=7
  - MAXCTL=2

1. Add the following from Table 20 on page 232:

   ```
   345 (9402 or 9404 Three-Line Communications Controller) +
   188 (X.25) + 90 (SDLC) = 623
   ```

2. Add the following from Table 19 on page 230:

   ```
   59 (SDLC MAXFRAME=521, MAXOUT=7) +
   152 (X.25, 16 virtual circuits) = 211
   ```

3. Add the following from Table 22 on page 232:

   ```
   1.5 (SDLC MAXOUT=7) * 2 (SDLC MAXCTL=2) = 3
   ```

4. Use Table 23 on page 232 to add the totals for the previous tables:

   ```
   623 + 211 + 3 = 837KB
   ```

Following is an example to calculate the subsystem storage for a 9406 Multiline Communications Controller with:

- An SDLC port group using short-hold mode (LINE1) with:
  - 1 port
  - MAXFRAME=521
  - MAXCTL=2
- An SDLC port group using short-hold mode (LINE2) with:
  - 3 ports

  - MAXFRAME=521
  - MAXCTL=6

1. Add the following from Table 20 on page 232:

   ```
   280 (9406 Multiline Communications Controller) +
   125 (SDLC short-hold mode [once per subsystem]) +
   47 (X.21 switched [once per subsystem]) = 452
   ```

2. Get the following from Table 19 on page 230 for LINE1:

   ```
   14 (SDLC short-hold mode MAXFRAME=521) *
   1 (port per port group) = 14
   ```

3. Calculate the following from Table 19 on page 230 for LINE2:

   ```
   14 (SDLC short-hold mode MAXFRAME=521) *
   3 (ports per port group) = 42
   ```

4. Add the total for Table 19 on page 230:

   ```
   14 (LINE1) + 42 (LINE2) = 56
   ```

5. Calculate the following from Table 21 on page 232:

   ```
   51 * 2 (one port group for each line) = 102
   ```

6. Calculate the following from Table 22 on page 232 for LINE1:

   ```
   2.8 (SDLC short-hold mode) * 2 (MAXCTL=2) = 5.6
   ```

7. Calculate the following from Table 22 on page 232 for LINE2:

   ```
   2.8 (SDLC short-hold mode) *
   6 (MAXCTL=2) = 16.8
   ```

8. Add the total for Table 22 on page 232:

   ```
   5.6 (LINE1) + 16.8 (LINE2) = 22.4
   ```

9. Use Table 23 on page 232 to add the totals for the previous tables:

   ```
   405 + 56 + 102 + 22.4 + 47 = 632.4KB
   ```

Table 19 describes the storage requirements for each line or ISDN channel selected (varied on).

Table 19. Storage Requirements per Line or Port Group

Storage Requirements per Line or Port Group	Size (KB)	Selected
SDLC		
MAXFRAME=265, MAXOUT=7	56	
MAXFRAME=521, MAXOUT=7	59	
MAXFRAME=1033, MAXOUT=7	66	
MAXFRAME=2057, MAXOUT=7	81	
MAXFRAME=265, MAXOUT=28	72	
MAXFRAME=521, MAXOUT=28	86	
MAXFRAME=1033, MAXOUT=28	115	
MAXFRAME=2057, MAXOUT=28	172	
SDLC Short-Hold Mode	10	
MAXFRAME=265, MAXOUT=7	14	
MAXFRAME=521, MAXOUT=7	21	
MAXFRAME=1033, MAXOUT=7	36	

Table 19. Storage Requirements per Line or Port Group (continued)

Storage Requirements per Line or Port Group	Size (KB)	Selected
X.25 (Six-Line Communications Controller)	200	
8 virtual circuits, MAXFRAME=1024	240	
8 virtual circuits, MAXFRAME=2048, 4096	225	
16 virtual circuits, MAXFRAME=1024	260	
16 virtual circuits, MAXFRAME=2048, 4096	350	
32 virtual circuits, MAXFRAME=1024	415	
32 virtual circuits, MAXFRAME=2048, 4096	480	
48 virtual circuits, MAXFRAME=1024	575	
48 virtual circuits, MAXFRAME=2048, 4096	605	
64 virtual circuits, MAXFRAME=1024	735	
64 virtual circuits, MAXFRAME=2048, 4096		
IDSN B-Channel ISDN Data Link Control (IDLC)	125	
MAXFRAME=1033	160	
MAXFRAME=2057	175	
MAXFRAME=4105	195	
MAXFRAME=4105		
	135	
IDSN D-Channel (one for each adapter)		
Bisynchronous	30	
MAXBUFFER=256	30	
MAXBUFFER=512	31	
MAXBUFFER=1024	32	
MAXBUFFER=2048	35	
MAXBUFFER=4096	40	
MAXBUFFER=8192		
Asynchronous	39	
MAXBUFFER=256	39	
MAXBUFFER=512	40	
MAXBUFFER=1048	41	
MAXBUFFER=2024	43	
MAXBUFFER=4096		
Total Selected	**XXXXX**	

Table 20 describes the requirements for storage for each protocol.

Table 21 on page 232 describes the storage requirements for each SDLC port group using short-hold mode.

Table 22 describes the storage requirements for each controller.

Table 20. Storage Requirements per Protocol

Storage Requirements per Protocol	**Size (KB)**	**Selected**
9402, 9404, 9406 Six-Line Communications Controller SDLC SDLC short-hold mode 6-Line Controller (SDLC, SDLC SHM, X.25, BSC, or ASYNC) ISDN BSC Asynchronous X.21 switched Total selected	350 90 125 60 465 62 57 47 XXXXX	

Table 21. SDLC Port Group Using Short-Hold Mode Storage Requirements

Storage Requirements per Port Group	**Size (KB)**	**Selected**
Port group	51	
Total selected	XXXXX	

Table 22. Controller Storage Requirements

Storage Requirements per Controller	**Size (KB)**	**Selected**

SDLC:
MAXOUT=7
MAXOUT=28
	
	1.5
	4.2	

SDLC short-hold mode MAXOUT=7	 2.8	
Total selected	XXXXX	

Table 23. Total Storage Requirements

Totals	**Selected**
Total selected from Table 19 Total selected from Table 20 Total selected from Table 21 Total selected from Table 22	
Totals Total should be <950000 bytes for the 9406 Multiline and Three-Line Communications Controller. Total should be <1900000 bytes for the Six-Line Communications Controller.	

Table 23 allows you to combine information from the indicated tables.

For the Six-Line Communications Controller, the total should be <1900000 bytes. If the system configuration exceeds the above volumes in the same communications controller, message CPA58C4 may be sent to the QSYSOPR message queue or the configured message queue. To correct this problem, consider the following options:

- Reduce some of the configuration parameters (frame size, number of outstanding frames, number of remote stations or remote controllers) for the lines and calculate the total again.

- Ensure that all lines are not running, or varied on at the same time.
- Contact your IBM service representative to get an additional communications subsystem.

Table 24. 9401 Model 150 Line Speeds

Line Speeds Per Protocol	**Speed (KB)**	**Selected**
Async running TCP/IP SLIP	115.2K	
Async other applications	28.8K	
Bisync, SDLC, X.25	64K	
Note: Storage Limits • 1 LAN card (Ethernet or Token-Ring) • 16 Virtual Circuits per X.25 line • 64 Stations of SDLC on the MFIOP		

For additional IOPs, see the *AS/400e Handbook* .

Appendix A. AS/400 VTAM Node Support

VTAM and NCP provide support for the AS/400 system as shown in Table 25. It shows the various types of support available for an AS/400 system connected to NCP through a switched or nonswitched line.

Table 25. AS/400 System Connected to NCP through a Switched or Nonswitched Line

	VTAM V3R2	**VTAM V3R3**	**VTAM V3R4.x**	**VTAM V4R1**
NCP VSE V4R1	2.0	2.0	2.0	2.0
NCP MVS/VM V4R2	2.0	2.0	2.0	2.0
NCP V4R2 Feature	2.0	2.0	2.0	2.0
NCP V4R3.1	2.1	2.1	2.1	2.1
NCP V5R3 VSE only	2.1	2.1	2.1	2.1
NCP V5R4	2.1	2.1	2.1	2.1
NCP V6R1	2.1	2.1	2.1	2.1
NCP V6R2	2.1	2.1	2.1	APPN

Notes:

2.0 indicates the AS/400 system is supported as a physical unit (PU) type 2.

2.1 indicates the AS/400 system is supported as a type 2.1 node.

APPN indicates full APPN connectivity support.

The following figure shows the various types of support available for a locally attached AS/400 system.

Table 26. AS/400 System Locally Attached to VTAM (not through a NCP

	VTAM V3R2	**VTAM V3R3 (VM only)**	**VTAM V3R4 (MVS/VSE only)**	**VTAM V4R1**
AS/400 switched	2.0	2.1	2.1	APPN
AS/400 nonswitched	2.0	2.1	2.1	APPN

Notes:

2.0 indicates the AS/400 system is supported as a PU type 2.

2.1 indicates the AS/400 system is supported as a type 2.1 node.

APPN indicates full APPN connectivity support.

Appendix B. Planning for Coexistence

Use this appendix when different types of systems are in use on the same network.

Coexistence includes the following:

- Data coexistence: data transmitted from one system can be received and used on another system in the network.
- Security coexistence: data transmitted from one system will be secure on another system in the network.

Some types of coexistence, including display station pass-through, APPN, and Client Access may require a program temporary fix (PTF).

Data Coexistence

Simply transferring data to another system does not ensure that the data can be processed and used on the other system.

- Table 27 and Table 28 on page 238 show how objects transmitted from an AS/400 system are handled by another AS/400 system, a System/36, and a System/38.
- Table 29 on page 238 shows how objects transmitted from a System/36 are handled by an AS/400 system, another System/36, and a System/38.
- Table 30 on page 239 shows how objects transmitted from System/38 are handled by an AS/400 system, a System/36, and another System/38.

Table 27. Distribution of AS/400 System Objects (See Note 1), Part 1

From AS/400 System	Object Distribution			File Transfer Support			Distributed Data Management		
	To System/36	**To System/38**	**To AS/400 System**	**To System/36**	**To System/38**	**To AS/400 System**	**To System/36**	**To System/38**	**To AS/400 System**
File Physical Logical	 Yes[1] No	 Yes[1] No	 Yes[1] No	 Yes No	 N/A N/A	 Yes No	 Yes No	 Yes No	 Yes No
Save file	Yes[5]	No[4]	Yes	No	N/A	No	No	No	No
Spooled file	Yes	Yes	Yes	Yes[2]	N/A	Yes[2]	Yes[2]	Yes[2]	Yes[2]
Input stream	Yes	Yes	Yes	No	N/A	No	No	Yes[3]	Yes[3]

Notes:

[1] The transmitted file becomes a physical file in arrival sequence on the target system (consecutive file on System/36).

[2] The object is converted into a user file before it is transferred to the target system.

[3] The Submit Remote Command (SBMRMTCMD) command can be used.

[4] If the save file was originally received from a System/38, this save file can then be sent to another System/38 and be processed there. The AS/400 system does not convert the file.

[5] Save files can be sent to a System/36, stored on the System/36, and later sent back to the AS/400 system and restored.

Table 28. Distribution of AS/400 System Objects, Part 2

	File Transfer Protocol		
From AS/400 System	**To System/36**	**To System/38**	**To AS/400 System**
File Physical Logical	 No No	 No No	 Yes Yes
Save file	No	No	No
Spooled file	No	No	No
Input stream	No	No	Yes[1]

Notes:

[1] Binary files can be sent between AS/400 systems.

Table 29. Distribution of System/36 Objects

	Object Distribution			File Transfer Support			Distributed Data Management		
From System/36	**To System/36**	**To System/38**	**To AS/400 System**	**To System/36**	**To System/38**	**To AS/400 System**	**To System/36**	**To System/38**	**To AS/400 System**
File Sequential Direct Indexed Alternative	 Yes Yes Yes Yes	 Physical[4] Physical[4] Physical[4] Physical[4]	 Physical[45] Physical[45] Physical[45] Physical[45]	 Yes Yes Yes No	 N/A N/A N/A N/A	 Physical Physical Physical No	 Yes Yes Yes No	 Physical Physical Physical No	 Physical Physical Physical No
System file SMF.LOG History	 Yes[2] Yes[2]	 Yes[2] Yes[2]	 Yes[2] Yes[2]	 Yes[2] Yes[2]	 N/A N/A	 Yes[2] Yes[2]	 Yes[2] Yes[2]	 Yes[2] Yes[2]	 Yes[2] Yes[2]
Spooled file entry	Yes	Yes	Yes	Yes[2]	N/A	Yes[2]	Yes[2]	Yes[2]	Yes[2]
Folder	Yes	No	Yes[5]	Yes[2]	N/A	No	Yes[2]	No	No
Document	Yes[2]	Yes[2]	Yes[2]	Yes[2]	N/A	No	Yes[2]	No	No
Input stream	Yes	Yes	Yes	No	N/A	No	No	No	No
Library	Yes[1]	No	Yes[2,3,4,5]	Yes[2]	N/A	Yes[2]	Yes[2]	No	Yes[2]
Library member Source Procedure Load Subroutine	 Yes[1] Yes[1] Yes[1] Yes[1]	 Physical[1,4] Physical[1,4] No No	 Physical[1,4] Physical[1,4] No No	 Yes Yes Yes Yes	 N/A N/A N/A N/A	 Yes[3] Yes[3] Yes[3] Yes[3]	 Yes[2] Yes[2] Yes[2] Yes[2]	 Yes[2] Yes[2] No No	 Yes[2] Yes[2] No Yes[2]

Table 29. Distribution of System/36 Objects (continued)

	Object Distribution			File Transfer Support			Distributed Data Management		
From System/36	**To System/36**	**To System/38**	**To AS/400 System**	**To System/36**	**To System/38**	**To AS/400 System**	**To System/36**	**To System/38**	**To AS/400 System**

Notes:

[1] Object distribution can send a single member, multiple members that match a selection value, or all members of a library.

[2] The object is converted into a nonsystem file before being transferred.

[3] When System/36 library members are received into the System/36 processing environment, they are converted, due to the different library concept, as follows:

System/36 Member	**AS/400 File Member**
SOURCE	QS36SRC
PROC	QS36PRC
LOAD	QS36LOD
SUBR	QS36SBR

[4] A file or library member transmitted from a System/36 with object distribution as TYPE=DATA is always a physical file in arrival sequence.

[5] System/36 objects sent in System/36 format can be received into the System/36 processing environment.

Transmitting an object to another system does not mean that the object can be used on that system. This table assumes that the target system is able to process the object.

Table 30. Distribution of System/38 Objects

	Object Distribution			Distributed Data Management		
From System/38	**To System/36**	**To System/38**	**To AS/400 System**	**To System/36**	**To System/38**	**To AS/400 System**
File Physical Logical	 Yes[4] No	 Yes[1] No	 Yes[1] No	 Yes Yes[4]	 Yes Yes[1]	 Yes Yes[1]
Save file	Yes[5]	Yes	Yes	No	No	No
Spooled file	Yes	Yes	Yes	Yes[2]	Yes[3]	Yes[2]
Input stream	Yes	Yes	Yes	No	Yes[3]	Yes[3]

Notes:

[1] The transmitted file becomes a physical file in arrival sequence on the target system.

[2] The object is converted into a user file before being transferred to the target system.

[3] The Submit Remote Command (SBMRMTCMD) command can be used.

[4] The transmitted file becomes a consecutive file on a System/36.

[5] Save files can be sent to a System/36, stored on the System/36, and later sent back to the System/38 and restored.

Security Coexistence

All the systems in a network should be secure in the following areas:

- Location security, which verifies the identity of other systems in the network
- User ID security, which verifies the identity and authorization of users
- Resource security, which controls user access to particular resources, such as confidential databases.

See the *APPC Programming* book for security considerations when using the AS/400 system with advanced program-to-program communications (APPC) and APPN. For more information about general AS/400 security, see the *Security - Reference* book.

Appendix C. Communications Functions

This appendix provides information to help you determine which communications functions can be used with the various data link protocols.

A number of communications functions are compatible between the AS/400 system and the following systems:

- Other AS/400 systems
- System/36
- System/38
- System/370
- System/390
- Programmable work stations
- Nonprogrammable work stations

The communications functions you select will depend on the needs of your business and your network environment.

See Appendix A. AS/400 VTAM Node Support for information about System/370-type host program levels in relation to AS/400 nodes in the network.

After determining the communications functions that you want to use, go to Table 31 on page 242 to find out which communications functions are used with which data link protocols.

Data Link Protocol Considerations

The data link protocol that is chosen for a particular environment can have an effect on the performance of application programs in that environment. This is further discussed in “Network and Line Considerations” on page 188. This topic discusses some performance-related differences between the data link protocols supported on the AS/400 system. It also provides configuration considerations to help provide the best performance for a given protocol.

Note: These guidelines apply to the support that is offered on the AS/400 system. Other remote systems or devices may not provide the same level of support, or number of configuration options. Devices such as the 5294 Remote Control Unit support only a small subset of the configuration options that are discussed in this topic.

Use Table 31 on page 242, Table 32 on page 243, and Table 33 on page 244 to determine which communications functions can be used with various data link protocols.

Using IBM-Supplied Communications Functions

If you want to use IBM-supplied communications functions, refer to Table 31 on page 242.

For example, if you want to use distributed data management (DDM), notice that it can be used with Systems Network Architecture (SNA). Using Table 33 on page 244, notice that SNA and, therefore, DDM can be used with the following:

- DDI network
- Ethernet network
- Frame relay
- integrated services digital network (ISDN)
  - Using ISDN with (ISDN data link control (IDLC)), or the X.31 interface for X.25 packets
  - Using the 7820 ISDN adaptor and synchronous data link control (SDLC)
- SDLC
- Token-ring network
- twinaxial data link control (TDLC)
- X.25
- 8209 or 8229 local area network (LAN) Bridge

Writing Your Own Application Programs

If you want to write your own application programs, refer to Table 32 on page 243.

For example, if you want to write a BSCEL program using intersystem communications function (ICF), notice that binary synchronous communications equivalence link (BSCEL) can be used with binary synchronous communications (BSC). Using Table 33 on page 244, notice that BSC is supported with the BSC data link protocol. Therefore, a BSCEL program using ICF can be used with the BSC protocol.

Table 31. IBM-Supplied Communications Functions

	Network Protocols				
Communications Function	**Asynchronous Communications**	**BSC**	**SNA**	**TCP/IP OS/400**	**IPX OS/400**
CallPath/400			X	X	X
DDM			X	X	X
DHCF			X		
Display station pass-through			X	X	X
DSNX			X		
FTP			X	X	X
FTS	X	X	X	X	X
ITF	X				
LPD			X	X	X
LPR			X	X	X
NRF			X		
Object distribution			X	X	
Point-of-Sale Utility			X		
Retail pass-through			X		
RJE		X	X		

Table 31. IBM-Supplied Communications Functions (continued)

	Network Protocols				
Communications Function	**Asynchronous Communications**	**BSC**	**SNA**	**TCP/IP OS/400**	**IPX OS/400**
SMTP			X	X	X
SNADS			X	X	X
SNA pass-through			X	X	X
SPLS			X		
TELNET				X	
VM/MVS bridge		X	X	X	
3270 device emulation		X	X		

Table 32. User-Written Programs with IBM-Supplied Communications Functions

	Type of Application Program[1]					
Network Protocols	**Asynchronous Communications**	**BSC**	**SNA**	**TCP/IP Utilities**	**TCP/IP OS/400**	**IPX OS/400**
APPC programs using ICF or CPI Communications			X		X	X
Asynchronous programs using ICF	X					
BSCEL programs using ICF		X				
CallPath/400			X		X	X
Finance programs using ICF			X			
Finance programs using non-ICF			X			
Network management programs using SNA management services	X		X		X	X
PC pgragrams using Client Access API[2]			X		X	
Programs using TCP/IP programming interface			X	X		X
Programs using user-defined communications APIs	X	X	X	X		
Programs using virtual teminal APIs			X	X		

Table 32. User-Written Programs with IBM-Supplied Communications Functions (continued)

	Type of Application Program[1]					
Network Protocols	**Asynchronous Communications**	**BSC**	**SNA**	**TCP/IP Utilities**	**TCP/IP OS/400**	**IPX OS/400**
Programs using 3270 data stream APIs		X	X			
Programs using 3270 program interface		X	X			
Retail programs using ICF			X			
SNUF programs using ICF			X			
System/38 environment APPC programs			X		X	X
System/38 environment programs using BSC		X				
System/38 environment programs using LU 1			X			

Note:

1. Programs you write using the described interface.
2. Programs you write on the personal computer, not the AS/400 system.

Table 33. Compatible Data Links and Protocols

	Network Protocols					
Data Links	**Asynchronous**	**BSC**	**CallPath/400**	**SNA**	**TCP/IP OS/400**	**IPX OS/400**
Asynchronous communications	X					
BSC		X				
DDI network[4]			X	X	X	
Ethernet network[4]			X	X	X	X
Frame relay				X	X	X
IDLC			X	X		
ISDN internal adapter[2]				X		
LAN bridge[1]				X	X	X
SDLC			X	X		
TDLC[3]				X		
Token-ring network[4]			X	X	X	X[6]
Wireless network				X	X	X
X.25[5]	X		X	X	X	X
Point-to-Point (PPP)				X		

Table 33. Compatible Data Links and Protocols (continued)

	Network Protocols					
Data Links	**Asynchronous**	**BSC**	**CallPath/400**	**SNA**	**TCP/IP OS/400**	**IPX OS/400**
Note: 1. The 8209 LAN Bridge converts the token-ring protocol to Ethernet. 2. The 7820 Integrated Services Digital Network (ISDN) Terminal Adapter allows the SDLC protocol to operate on an ISDN interface. 3. Twinaxial data link control (TDLC) is supported only for Client Access. 4. This includes token-ring, Ethernet, or DDI network connections bridged over a frame relay network. 5. This includes X.25 over ISDN using the AS/400 integrated ISDN adapter or ISDN terminal adapter. 6. IPX does not support DDI or DDI bridge over a frame relay network.						

Appendix D. Communications Restrictions

Communications restrictions apply if any of the following communication functions are required when using the PCI WAN/Twinaxial IOA #2720, the PCI Two-line WAN IOA(#2721), or the Two-line WAN IOA (#2699); or the IPX protocol (when over LAN adapters or over frame relay):

- X.25, frame-relay, or IPX protocol.
- SDLC protocol if used to connect to more then 64 remote sites.
- Communications line speeds greater than 64 Kbps and up to 2.048 Mbps for synchronous data link control (SDLC) or frame-relay protocols (bisync always is limited to 64 Kbps maximum).
- Communications line speeds greater than 64 Kbps and up to 640 Kbps for X.25.

If you require any of the above protocols or configurations, then see the appropriate following subsection for additional rules and restrictions.

Note: The Marketing Configurator does not enforce these additional rules and restrictions. It is the responsibility of the marketing team and customer to enforce these restrictions .

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation

Software Interoperability Coordinator
3605 Highway 52 N
Rochester, MN 55901-7829
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Programming Interface Information

This publication is intended to help application programmers manage the communication functions of the IBM OS/400-licensed program. This publication documents General-Use Programming Interface and Associated Guidance Information.

General-Use programming interfaces allow the customer to write programs that obtain the services of the OS/400 licensed program.

Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, or other countries, or both:

Advanced Peer-to-Peer Networking
AnyNet
Application System/400
APPN
AS/400
AS/400e
AT
Client Access
CallPath/400
IBM
LPDA
NetView
Operating System/400
OS/400
System/36
System/38
System/370
System/390
VTAM
400

C-bus is a trademark of Corollary, Inc. in the United States and/or other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and/or other countries.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States and/or other countries.

PC Direct is a trademark of Ziff Communications Company in the United States and/or other countries and is used by IBM Corporation under license.

ActionMedia, LANDesk, MMX, Pentium, and ProShare are trademarks of Intel Corporation in the United States and/or other countries.

UNIX is a registered trademark in the United States and/or other countries licensed exclusively through X/Open Company Limited.

Other company, product, and service names may be the trademarks or service marks of others.

Bibliography

AS/400 Books

The following IBM AS/400 books referred to in this book can be used for more information on the subject matter.

- *Alerts Support*, SC41-5413-00, provides information for configuring and using AS/400 alert support. The book discusses how to allow end-user applications to create alerts and notify the alert manager of alerts that need to be handled. Other topics include, how to control the creating and sending of alert messages for problem management, and how to perform central site problem analysis for the AS/400 systems in the network.
- *APPC Programming*, SC41-5443-00, provides the application programmer with information about the support that is provided by the AS/400 system. Included in this book are application program considerations, configuration requirements and commands, problem management for advanced program-to-program communications (APPC), and general networking considerations.
- *Asynchronous Communications Programming*, SC41-5444-00, provides information on developing asynchronous communications applications programs that use the intersystem communications function (ICF).
- *Backup and Recovery*, SC41-5304-03, contains a subset of the information that is found in the *Backup and Recovery*, SC41-5304-03. The book contains information about planning a backup and recovery strategy. Other topics include, the different types of media available to save and restore procedures, and disk recovery procedures. It also describes how to install the system again from backup.
- *BSC Equivalence Link Programming*, SC41-5445-00, provides the information that is needed to write programs that use binary synchronous communications equivalence link (BSCEL) to communicate with a remote system. It also contains information about other systems and devices that communicate with BSCEL on the AS/400 system. This book describes how to set up BSCEL and how to run application programs that use BSCEL.
- *CL Programming*, SC41-5721-02, provides the application programmer or programmer with a wide-ranging discussion of AS/400 programming topics. Included is a general discussion of objects and libraries, control language (CL) programming, and controlling flow and communications between programs.
- *CL Reference (Abridged)*, SC41-5722-03, provides the application programmer with a description of the AS/400 control language (CL) and its commands. Each command description includes a syntax diagram, parameters, default values, keywords, and an example.
- *Communications Configuration*, SC41-5401-00, contains general configuration information. Included are detailed descriptions of network server, network interface, line, controller, device, mode, and class-of-service, and NetBIOS descriptions.
- *DSNX Support*, SC41-5409-00, provides information for configuring an AS/400 system to use communications and systems management functions. For the AS/400 system, this support includes change management functions in an IBM NetView Distribution Manager (NetView DM) network and problem management functions in a network.
- *Finance Communications Programming*, SC41-5449-00, describes how finance support communicates with a controller, and how to set up finance support. It provides information for writing application programs to communicate with application programs on the finance controller.
- *ICF Programming*, SC41-5442-00, provides programming information for writing application programs that use the intersystem communications function (ICF). It describes how ICF provides program-to-program communications between the AS/400 system and other systems and program-to-device communications between the AS/400 system and hardware devices.
- *Internetwork Packet Exchange (IPX) Support*, SC41-5400-00, contains information on configuring and using the Internetwork Packet Exchange (IPX) support on AS/400 business computing systems.
- *Intrasystem Communications Programming*, SC41-5447-00, provides information about interactive communications between two application programs on the same AS/400 system. This book describes the communications operations that can be coded

into a program that uses intrasystem communications support to communicate with another program. It also provides information on developing intrasystem communications application programs that use the intersystem communications function (ICF).

- *ISDN Support*, SC41-5403-00, provides the programmer who is responsible for configuring and using integrated services digital network (ISDN), as well as specific information about IBM's implementation of ISDN.
- *LAN, Frame-Relay and ATM Support*, SC41-5404-01, contains general local area network (LAN) information, Token-Ring information, and Ethernet information as well as Token-Ring Network Manager support information.
- *OS/2 Warp Server for AS/400 Administration*, SC41-5423-00, provides information about the Warp Server/400 licensed program. Warp Server/400 provides fast file serving that uses the file server input-output processor (IOP). This information includes how to plan your network to use the AS/400 system as a file server. It also includes how to install and configure Warp Server/400, and how to use the Warp Server/400 file system.
- *Physical Planning Reference*, SA41-5109-02, provides the data processing manager, system administrator, and installation planning representative with information for planning to set up the AS/400 system. This book also includes information on cable considerations, physical specifications, electronic customer support, and unpacking considerations.
- *Remote Work Station Support*, SC41-5402-00, provides the system operator, application programmer, system programmer, and service personnel with information about setting up and using remote work station support. Topics include display station pass-through, distributed host command facility, SNA pass-through, network routing facility, Systems Network Architecture (SNA) primary LU2 support, and 3270 remote attachment.
- *Retail Communications Programming*, SC41-5448-00, describes how to set up, start, and end retail communications. It includes information about creating communications definitions for retail communications as well as information on writing retail communications applications and about using retail pass-through and retail communications. This book also includes information on how to write application programs to communicate with programs on point-of-sale controllers.
- *Security - Reference*, SC41-5302-03, tells how system security support can be used to protect the system and data from being used by people without proper authorization. Other topics include, protecting data from intentional or unintentional damage or destruction, keeping security information up-to-date, and setting up security on the system.
- *Simple Network Management Protocol (SNMP) Support*, SC41-5412-00, describes how to configure an AS/400 to use Simple Network Management Protocol (SNMP), and discusses SNMP agents, subagents, managers, and Management Information Bases (MIBs). Other topics include, client inventory management, journal for SNMP logging, and problem analysis for SNMP.
- *SNA Distribution Services*, SC41-5410-01, provides the system operator or system administrator with information about configuring a network for Systems Network Architecture distribution services (SNADS) and the Virtual Machine/Multiple Virtual Storage (VM/MVS) bridge. In addition, object distribution functions and document library and distribution services are discussed.
- *Sockets Programming*, SC41-5422-03, provides information about sockets, a method of communicating between processes. The use of sockets to connect processes is key to the distribution of work among clients and servers. This book provides information about setting up the sockets environment as well as a description of the sockets application program interface (API).
- *System API Reference*, SC41-5801-03, provides information for those customers or systems houses that want to:
  - Write their own communications protocol on the AS/400 system to connect to systems in ways not currently possible with IBM communications support, or
  - Connect programmable work stations (PWSs) through a specialized Virtual Terminal Manager interface.
- *System Operation*, SC41-4203-00, provides the system operator or system administrator with information about controlling jobs, sending and receiving messages, responding to error messages, starting and stopping the system.

Other topics include, using control devices, and managing your AS/400 operations.

- *Basic System Operation, Administration, and Problem Handling*, SC41-5206-03, provides the system operator or system administrator with information about how to use the system unit control panel, and starting and stopping the system. Other topics include, how to use tape units, tapes, diskettes, and optical libraries, working with PTFs, and handling and reporting system problems.
- *TCP/IP Configuration and Reference*, SC41-5420-03, provides information for configuring and using AS/400 TCP/IP support. The applications included are Network Status (NETSTAT), Packet InterNet Groper (PING), TELNET, File Transfer Protocol (FTP), Simple Mail Transfer Protocol (SMTP), Line Printer Requester (LPR), and line printer daemon (LPD). The Transmission Control Protocol (TCP) and UDP Pascal application program interface (API) is also discussed.
- *Work Management*, SC41-5306-03, provides the programmer with information about how to create and change a work management environment.
- *X.25 Network Support*, SC41-5405-01, provides information about configuring and using X.25 networks.
- *AS/400e Handbook*, GA19-5486-17, provides information about all aspects of AS/400 Advanced Series. It includes descriptions of AS/400 Advanced Application Architecture; system concepts; AS/400 Advanced System, AS/400 Advanced Server, and AS/400 Advanced Portable. Other topics include, other hardware components including all peripherals; and licensed programs available.

Non-AS/400 Books

The following IBM non-AS/400 books can also be used for more information.

- *Common Programming Interface Communications Reference* , SC26-4399
- *IBM 3270 Information Display System: 3274 Control Unit Customization Guide* , GA23-0061
- *IBM 3270 Information Display System: X.25 Operation*, GA23-0204
- *System/38 Data Communications Programmer’s Guide* SC21-7825

Index

Readers' Comments — We'd Like to Hear from You

AS/400e
Communications Management
Version 4

Publication No. SC41-5406-02

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	□	□	□	□	□

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	□	□	□	□	□
Complete	□	□	□	□	□
Easy to find	□	□	□	□	□
Easy to understand	□	□	□	□	□
Well organized	□	□	□	□	□
Applicable to your tasks	□	□	□	□	□

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? □ Yes □ No

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

Name	Address
Company or Organization	
Phone No.	

Readers' Comments — We'd Like to Hear from You
SC41-5406-02

Cut or Fold
Along Line

Fold and Tape — **Please do not staple** — Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

IBM CORPORATION
ATTN DEPT 245
3605 HWY 52 N
ROCHESTER MN 55901-7829

Fold and Tape — **Please do not staple** — Fold and Tape

Cut or Fold
Along Line

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

SC41-5406-02