GC27-6998-2
File No. S370-30

Systems

VTAM Concepts and Planning

Virtual Telecommunications Access Method (VTAM)

VTAM Level 2

DOS/VS
OS/VS1
OS/VS2

IBM

Third Edition (August, 1975)

This edition applies to VTAM Level 2 when available in DOS/VS, OS/VS1, and OS/VS2. Until VTAM is available in an operating system, the information in this publication is for planning purposes and is subject to change. VTAM Level 2 includes support for switched SDLC links, local 3790 attachment and the SDLC 3270, 3650, 3660, 3767, 3770 and System/32. For VTAM Level 2 users, this edition replaces the previous edition, GC27-6998-1. Changes will be reported in subsequent revisions or Technical Newsletters.

Copies of this and other IBM publications can be obtained through your IBM representative or the IBM branch office serving your locality.

A form has been provided at the back of this publication for readers' comments. If this form has been removed, address comments to: IBM Corporation, Department 63T, Neighborhood Road, Kingston, New York 12401

Preface

This publication provides a detailed overview of Virtual Telecommunications Access Method (VTAM) concepts and planning considerations. It is directed primarily to data processing managers and system programmers of DOS/VS and OS/VS installations that may install or maintain a telecommunication system that uses VTAM.

The installer of a system that uses VTAM should use this book to get a general understanding of VTAM concepts and the requirements and options that must be considered in planning and installing a VTAM system. Then, to perform specific steps, such as estimating main storage requirements and writing VTAM definition statements, the installer should use the appropriate VTAM System Programmer's Guide, the *NCP Generation* publication, relevant operating system publications, and publications that may be required to install the telecommunications terminals and terminal systems that will communicate with VTAM application programs. The programmer who writes VTAM application programs can use this book (although it is not required) to understand the context in which his program will be executed.

A more general description of VTAM is provided in *Introduction to VTAM,* GC27-6987.

How This Book Is Organized

- Chapters 1 and 2 provide an introduction to VTAM, discuss its relationship to Systems Network Architecture (SNA), and describes VTAM's major concepts and facilities. Complex concepts are introduced in these chapters and described in detail in subsequent chapters.
- Chapters 3, 4, and 5 describe the primary interfaces to VTAM. Chapter 3 describes how to define a VTAM telecommunication system. Chapter 4 describes how to control a VTAM system, and Chapter 5 describes in general how to write VTAM application programs.
- Chapter 6 describes VTAM's reliability, availability, and serviceability features.
- Chapter 7 describes hardware and software requirements for VTAM and discusses planning considerations for functions such as telecommunication security.
- Chapter 8 describes support for local 3270, BSC, and start-stop terminals.

Related Publications (Other Than VTAM)

The reader should be familiar with basic teleprocessing concepts. These may be found in the publication *Introduction to Data Communications Systems*, SR20-4461.

Depending on the terminal product with which VTAM is used, the reader may be required to become familiar with certain details of Systems Network Architecture (SNA). Consult the programming publications associated with each SNA terminal product to determine whether this is a requirement. Apart from installation requirements, some readers may wish to become familiar with SNA for general understanding; these readers are referred to *Systems Network Architecture General Information* GA27-3102 and *Systems Network Architecture Format and Protocol Reference Manual,* GA27-3112.

References are made in this publication to the *NCP Generation* publication. The complete title and form number is: *IBM 3704 and 3705 Communications Controllers Network Control Program/VS Generation and Utilities Guide and Reference Manual for OS/VS and DOS/VS VTAM Users,* GC30-3008.

Other publications (other than VTAM publications) referred to in this publication or related to this publication are listed in the Bibliography at the end of this book.

Related VTAM Publications

VTAM Concepts and Planning is one of a number of related VTAM publications. Figure P-1 shows the reading order of these publications for different user needs.

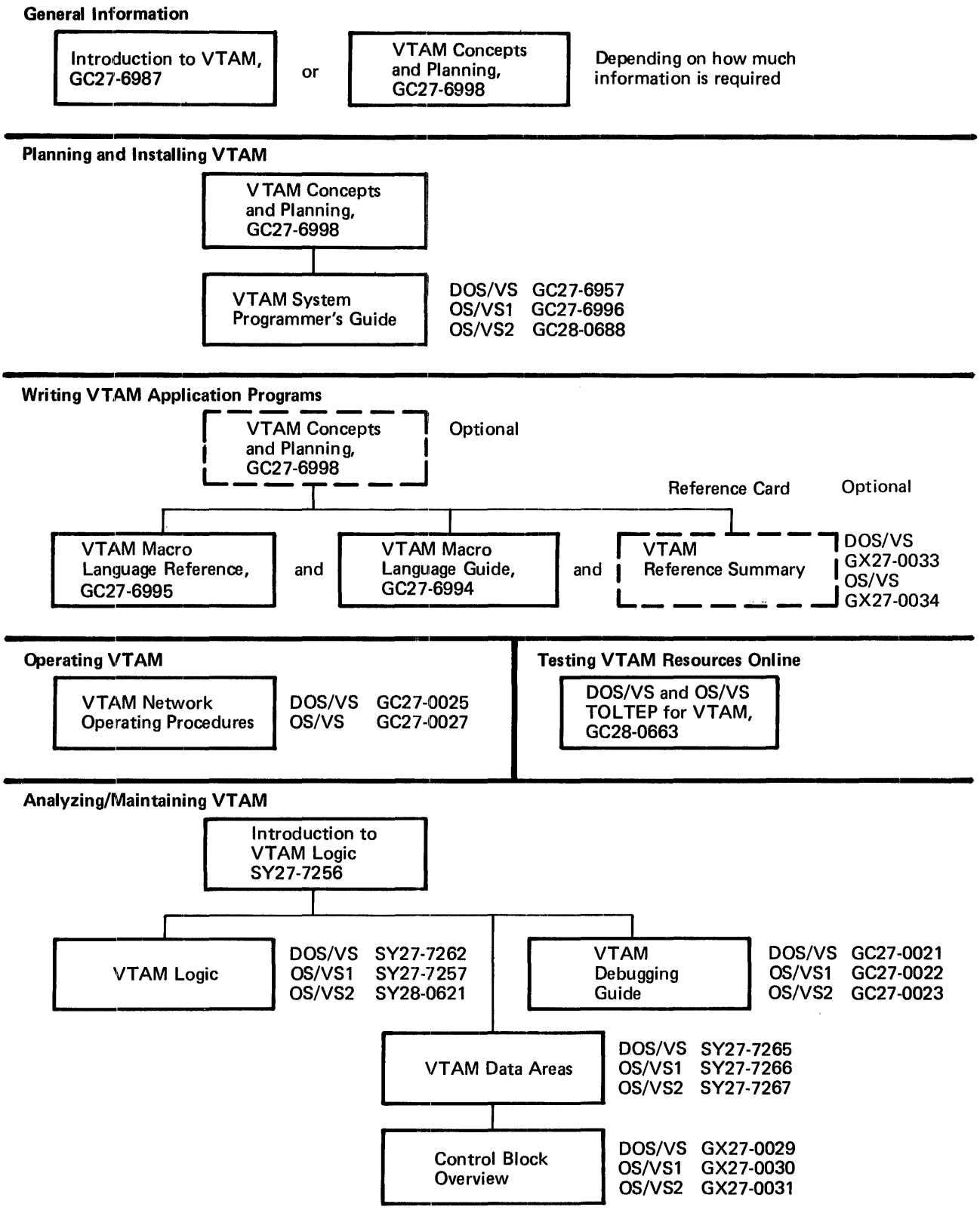


Figure P-1. The VTAM Publications Plan

Contents

Figures

Summary of Changes for Third Edition

Changes to this publication since the previous edition include:

- Relating VTAM to Systems Network Architecture (SNA)
- Describing new functional support by VTAM
- Correcting and improving the previous edition

The changes are summarized below.

Relating VTAM to SNA

These changes have been made to relate VTAM to Systems Network Architecture (SNA):

- VTAM is now described as a component of IBM's Systems Network Architecture. The relationship between SNA and VTAM is explained generally in Chapters 1 and 2.
- Certain terms previously used by VTAM have been changed to conform to uniform SNA definition of terms. The old terms cross-refer to the new terms in the glossary and a definition of the new terms has been added. Figure SC-1 summarizes these terminology changes.

Describing New Functional Support

These changes have been made to describe new functional support by VTAM:

- VTAM's support for certain SNA terminal products on switched (dial) SDLC lines is described. "Terminal" is used in a general sense of "remote point" relative to VTAM; it can include program logic in a complex subnetwork or a single input/output device. An installation defines its particular switched network requirements and options when it defines the network. The installation's VTAM application programs need not be aware of whether an SNA terminal is on nonswitched or switched lines or is locally attached. Defining switched lines and defining sets of SNA terminals (expressed as physical units and logical units) that can be made active or inactive is described in Chapter 3. Control by the network operator is described in Chapter 4. A summary of switched network support for SNA terminals is provided in Chapter 7.
- VTAM's support for a locally attached 3790 Communication System is described. As with switched network support, an installation defines local 3790s when defining the network; the VTAM application program addresses logical units and is not aware whether the 3790 is locally or remotely attached. Defining a local SNA major node (a group of local 3790 physical units and logical units that can be made active or inactive as a unit) is described in Chapter 3. Control by the network operator is described in Chapter 4.

This term in the previous edition	Has been changed to this term in this edition
Synchronous flow	Normal flow
Asynchronous flow	Expedited flow
FME response	Definite response 1
RRN response	Definite response 2
Indicator (for session control and data flow control)	Command or reply

Figure SC-1. Terminology Changes in This Edition

- Support for terminal-entered LOGON or LOGOFF commands for certain SNA terminals is described. These commands can be entered from SNA IBM 3270 Information Display System terminals, IBM 3767 Communication Terminals, and IBM 3770 Data Communication System terminals. This facility corresponds to the SNA unformatted system services (USS) provided by the system services control point function (SSCP) in VTAM. In addition to the USS or character-coded command that is entered by a terminal operator, VTAM supports the same logon and logoff request facility from other SNA terminals in a field-formatted command, corresponding to the SNA formatted system services (FSS). FSS support is provided for SNA terminals that can provide a field-formatted request; USS support is provided for SNA terminals that cannot provide a field-formatted request.
- SNA terminals can now be associated with a logon mode which defines a set of application protocols. This set of protocols usually corresponds to a type of work (such as batch or interactive) that is to be done by the VTAM application and the SNA terminal. Certain SNA terminals can work in only one logon mode; other SNA terminals can work in one of several logon modes. Defining logon modes is described in Chapter 3. The effect of a logon mode on a VTAM application program is described in Chapter 5.
- VTAM Support for the following IBM SNA terminals is listed in Appendix A:
  - IBM 3270 Information Display System (adapted for SNA)
  - IBM 3660 Supermarket System
  - IBM 3767 Communication Terminal
  - IBM 3770 Data Communication System

  In addition, the 3650 Retail Store System and the 3790 Communication System, already supported on nonswitched lines, are now supported on switched lines, and the 3790 Communication System is now supported on local channel attachment.

Correcting or Improving the Previous Edition

These changes have been made to correct or improve the previous edition:

- Chapter 3, "Creating a Telecommunication System with VTAM," has a rearranged description of the steps required to create a VTAM system.
- A new chapter, Chapter 8, "Support for Local 3270, BSC, and Start-Stop Terminals," has been added to describe this support in one place.
- The terminals supported by VTAM, described in Appendixes A and B in the previous edition, are now described in Appendix A.
- A bibliography has been added listing publications that are referred to in this book or that may relate to VTAM. (VTAM publications are listed in Figure P-1 of the Preface.)

CHAPTER 1. INTRODUCTION TO VTAM

This chapter summarizes the IBM Virtual Telecommunications Access Method (VTAM). It briefly tells what VTAM is and what it does.

What is VTAM?

VTAM directs the transmission of data between VTAM application programs and SNA terminals, local 3270 terminals, and certain BSC and start-stop terminals. VTAM enables application programs to communicate with supported terminals without concern for intermediate connections such as controllers and telecommunication lines. Appendix A lists the terminals VTAM supports.

VTAM is compatible with Systems Network Architecture (SNA) and performs the following services:

- Establishes, terminates, and controls access between application programs and terminals
- Permits transfer of data between application programs and terminals
- Permits application programs to share communication lines, communications controllers, and terminals
- Permits the operation of the telecommunication network to be monitored and altered
- Permits the configuration of the telecommunication network to be changed while the network is being used

Various users have access to these services:

- The application programmer can use the connection and data transfer services.
- The terminal operator and teleprocessing subsystems (such as the 3600 Finance Communication System) can use VTAM's services for logging onto application programs.
- The network operator can use VTAM's services for monitoring and controlling the telecommunication network.
- The system programmer can use VTAM's services for configuring the telecommunication network.

VTAM is executed under the DOS/VS, OS/VS1, and OS/VS2 operating systems.

VTAM supports the local attachment of the IBM 3790 Communication System and the IBM 3270 Information Display System. It uses the facilities of IBM 3704 or 3705 Communications Controllers with the network control program (NCP) in network control mode to control remotely attached terminals. These terminals can also be attached to a communications controller that is, itself, attached to a local communications controller. VTAM uses the NCP with or without the partitioned emulation programming (PEP) extension.

VTAM supports the IBM teleprocessing terminals that use synchronous data link control (SDLC). VTAM is designed to use the remote computing capability of the SNA terminals that have that capability and the full-duplex capability of SDLC. VTAM also supports some terminals that use start-stop or binary synchronous communication (BSC) line disciplines.

VTAM allows application programs written for TCAM to be executed in a VTAM system, using VTAM facilities and resources; these application programs use VTAM through

TCAM. VTAM can coexist with BTAM; that is, application programs written for BTAM can be executed in a data processing system containing VTAM, but they must use a network of terminals separate from the VTAM network. VTAM and BTAM can share a 3704 or 3705 Communications Controller by using an NCP with the PEP extension.

VTAM Provides System Coordination of Telecommunication Activity

VTAM assists in providing a system solution to telecommunication problems. Using the facilities of the operating system, of NCP's, and of teleprocessing subsystems, VTAM manages a telecommunication network and enables data to be transmitted between application programs in the central computer and remote locations.

VTAM supports the distribution of function outward from the central computer to devices such as the 3704 and 3705 Communications Controllers and SNA terminals that have computing capability (for example, the IBM 3601 Finance Controller). By distributing function, less of the central computer's capacity needs to be devoted to activities associated with controlling the telecommunication network and more of its capacity can be applied to the processing of data.

VTAM enables application programs in the central computer to communicate with application programs in SNA terminals that have computing capability. It provides a method of communication for the transmission of data independently of how devices are attached to the central computer and independently of the type of terminal being used.

In addition to enabling application programs in the central computer to communicate with SNA terminals, VTAM enables these same application programs in the central computer to communicate with local 3270, and supported start-stop and BSC terminals. VTAM offers direct control communication with supported terminals. If queued control is needed, applications in the central computer can use VTAM through TCAM.

In a VTAM system, the application program in the central computer:

- *Is independent* of line activities (such as line scheduling)
- *Is independent* of attachment concerns (such as whether a terminal is locally or remotely attached)
- *Is capable* of referring to terminals symbolically
- *Communicates* with terminals on a real-time basis

VTAM Provides a Telecommunication Base

VTAM is designed to use System/370 virtual storage, the IBM communications controllers, DOS/VS, OS/VS1, OS/VS2, and both general purpose and industry-oriented SNA terminals (such as the IBM 3790 Communication System and the IBM 3600 Finance Communication System). With the IBM Virtual Storage Access Method (VSAM), VTAM can provide a complementary data base/data communication facility. In addition to its primary role of data transmission, VTAM has features that establish it as a base for building both large and small telecommunication systems. These features are:

- Sharing of network resources, which reduces line costs and makes more efficient use of the network
- Concurrent execution of TCAM and VTAM application programs using the same telecommunication network
- Services required for interactive applications (for example, online inquiries and updates)

- Operation of the IBM 3704 and 3705 Communications Controllers to reduce the number of functions performed in the central computer for remote devices
- Generation options for tailoring the telecommunication system to user needs
- Support for remotely attached central processing units (CPUs) using binary synchronous communication (BSC)
- Support for SNA terminals with their own computing capability, such as the IBM 3600 Finance Communication System
- Support for SDLC networks that use either switched or nonswitched lines, or both
- Support for local attachment of the IBM 3790 Communication System and the IBM 3270 Information Display System
- Support for many different terminals which use different line disciplines (start-stop, binary synchronous communication, and synchronous data link control)
- Reliability, availability, and serviceability aids to assist in reducing errors in the telecommunication system, in reducing the impact of errors that do occur, and in maintaining the telecommunication system

System Requirements for Using VTAM

VTAM can be a component of the DOS/VS, the OS/VS1, and the OS/VS2 operating systems. The DOS/VS supervisor must be generated to include DOS/VS multiprogramming support.

The System/370 instruction set must include the Compare and Swap and the Compare Double and Swap instructions. These instructions are part of a hardware feature available on System/370 CPUs.

VTAM requires a 3704 or 3705 Communications Controller in network control mode to support remotely attached communications controllers and terminals. Appendix A lists terminals that can be used by VTAM.

CHAPTER 2. VTAM'S ROLE IN AN SNA TELECOMMUNICATION SYSTEM

This chapter discusses the role of VTAM in a telecommunication system that is compatible with Systems Network Architecture (SNA) and explains how VTAM relates to the other components in the system. Basic VTAM characteristics and concepts are also introduced.

Functions of VTAM

VTAM manages the activities of a telecommunication system. It allocates resources and manages the flow of data between the central processor and the remote locations in the telecommunication system. To accomplish this, VTAM provides the following functions:

Starting and stopping the network: VTAM enables an installation to define the telecommunication system and some of its characteristics. Once the system is defined, VTAM can be started and the system initialized. VTAM can also be used to shut down the telecommunication system in an orderly fashion.

Changing the configuration dynamically: VTAM enables the network operator to monitor the use of the resources within the telecommunication system and to alter the network as necessary.

Allocation: VTAM controls the allocation of network resources. By owning and controlling all resources, VTAM provides a focal point within the system for controlling the network; yet, at the same time, it enables the installation to use its network efficiently.

I/O processing: VTAM manages the transmission of data between application programs and terminals. It enables application programs and terminals to communicate with each other independently of how the terminal is connected to the central processing unit. VTAM also relies upon the distributed function throughout the network (such as in communications controllers and programmable terminals) to reduce the processing requirements in the central processing unit.

Reliability, availability, serviceability (RAS): VTAM offers a design and facilities that reduce the incidence of problems in the telecommunication system, reduce the impact of errors that do occur, and assist in maintaining the telecommunication system.

These functions provide a flexible, dynamic telecommunication system—a system that is responsive to varying installation needs, that allows extensive sharing of network resources, and that provides central control for important activities such as telecommunication security.

The next two sections describe how the access method fits into a teleprocessing system and a telecommunication network.

Teleprocessing—An Overview

The purpose of a *teleprocessing system* is to make the power of a central computer available to a number of users working at remote locations. To achieve this aim, a teleprocessing system must:

- Transmit data between the central computer and the remote locations
- Process data in the central computer

VTAM is directly involved in the transmission of data but not in the processing of data in the central computer. The application program using VTAM processes data.

The *telecommunication system* is the part of a teleprocessing system devoted to the transmission of data between the central computer and the remote locations. Figure 2-1 depicts the major elements in a VTAM telecommunication system.

The core of the telecommunication system is the central computer, which comprises a central processing unit (referred to as the *host CPU,*) channels, and auxiliary storage. In a VTAM system, the CPU must be a System/370. The major components in the CPU are the operating system (DOS/VS, OS/VS1, or OS/VS2), VTAM, and the application programs that use VTAM.

Besides the central computer, the other major component of the system is the *telecommunication network.* The network is composed of communications controllers, telecommunication lines, control units, and terminals.

A VTAM Telecommunication System

A telecommunication system can be viewed as a network of communications controllers, lines, and terminals coordinated by the telecommunication access method.

An application program and a terminal can communicate with each other only through VTAM. To communicate, these two elements must be connected. *Connection* is the allocation process by which VTAM establishes a path between an application program and a terminal. The *path* is composed of other elements (including lines, communications controllers, and control units) needed to transmit data between the application program and the terminal.

A terminal is connected to an application program for as long as explicitly requested by the application program, but intermediate elements are allocated (but not connected) only for the duration of a single data-transfer operation (such as a single read or write operation). Consequently, intermediate elements can be available simultaneously for paths to other terminals.

In providing the mechanism for transmitting data between application programs and terminals, VTAM removes much of the responsibility of network management from the application programs. The primary task involved in network management is resource allocation.

As network manager, VTAM handles the allocation of resources; that is, VTAM controls who uses what resources. The only aspects of resource allocation of direct concern to application programs are the issuance of requests to have terminals connected to them and acceptance of requests from terminals (logon) to be connected to an application program. Other aspects of allocation, including the control and use of intermediate elements, are handled by VTAM. The number, type, and disposition of intermediate elements are not apparent to the terminals and to the application programs. (For more details on the concept of connection, see "Application Program Concepts and Facilities," in Chapter 5.)

The major elements of a VTAM system are described to VTAM as *major and minor nodes* as shown in Figure 2-2. They include:

- VTAM application programs
- Network control programs (NCPs) for communications controllers
- Terminals

These elements are discussed below.

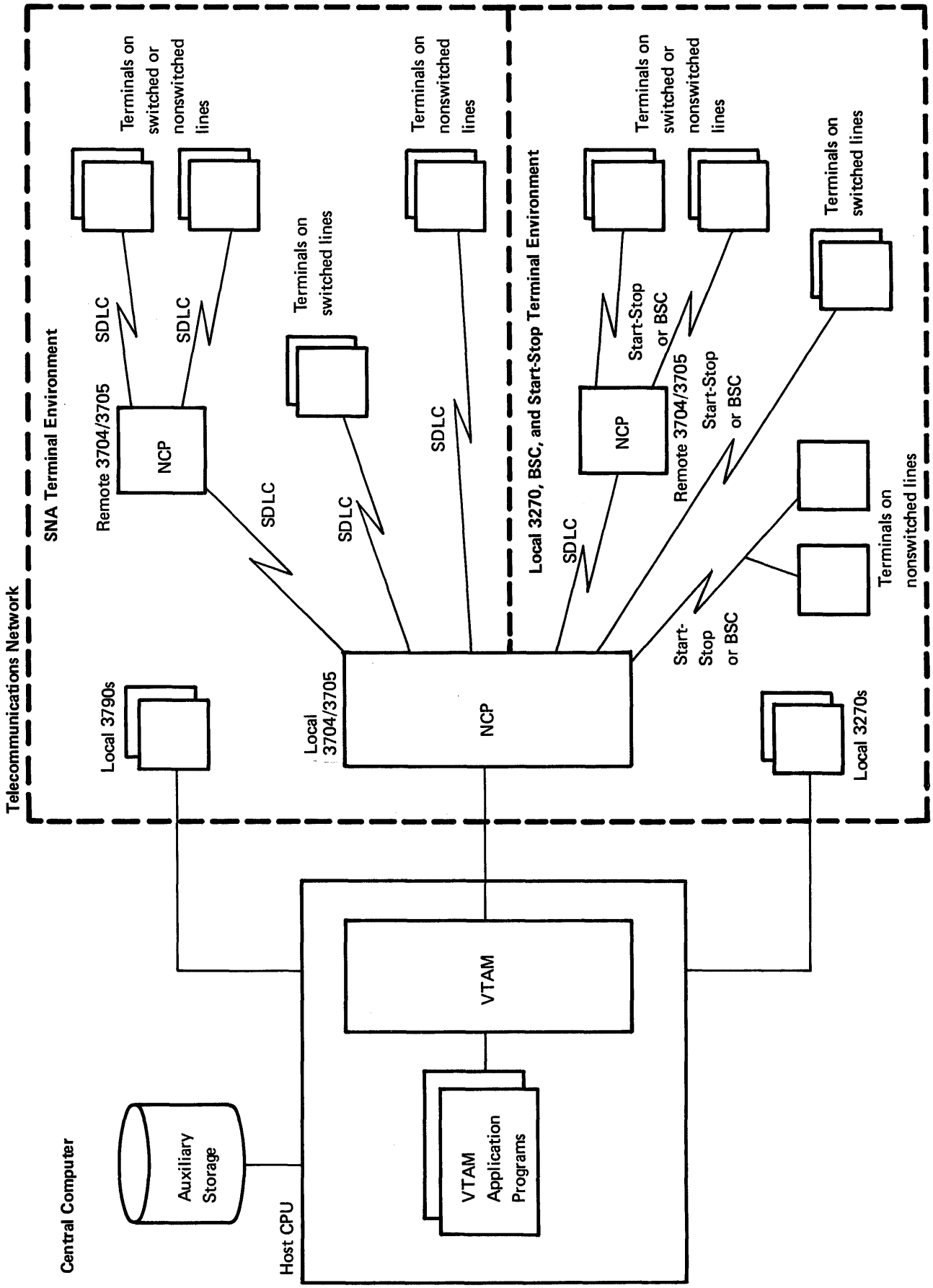


Figure 2-1. The Major Elements in a VTAM Telecommunications System

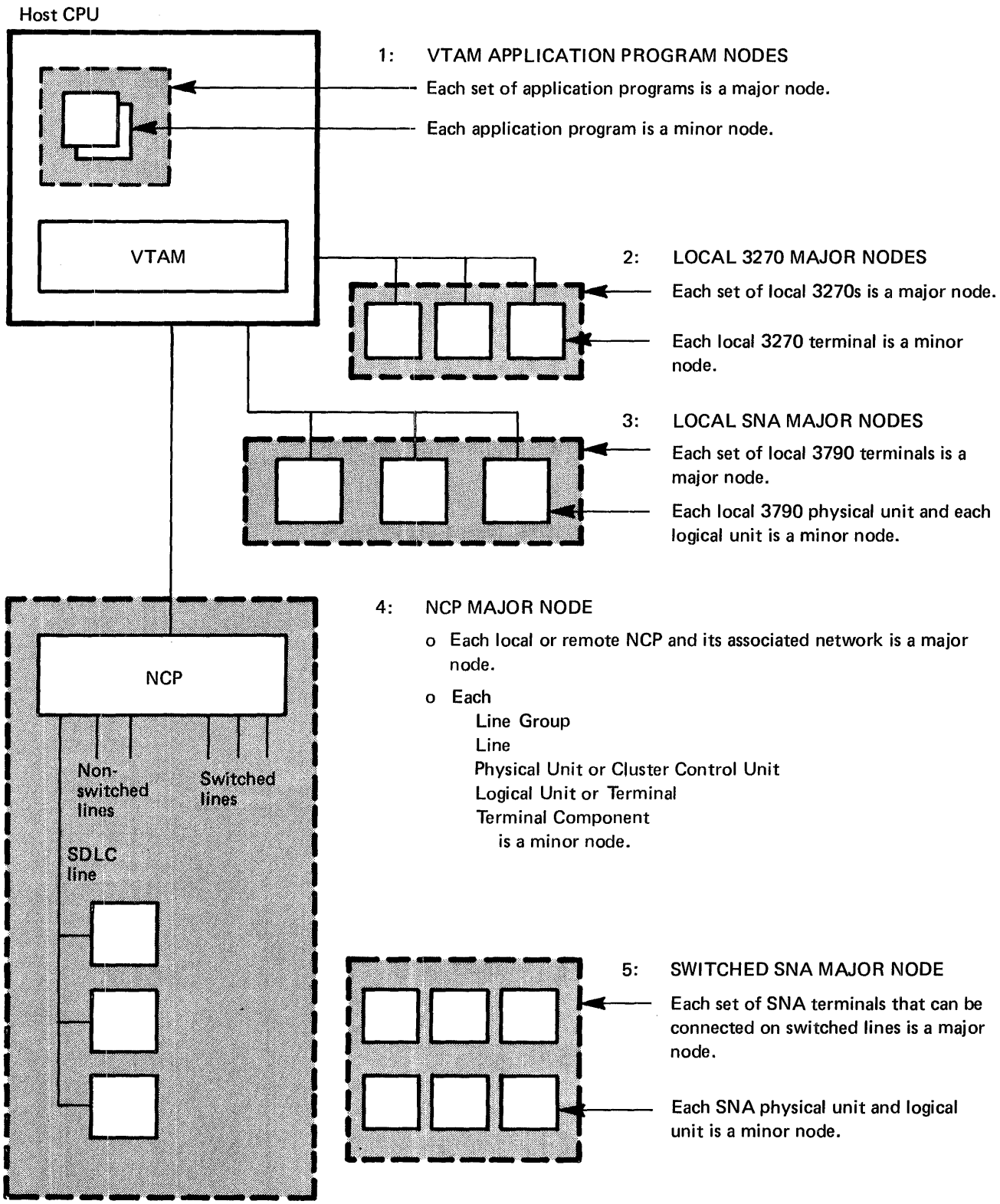


Figure 2-2. Major and Minor Nodes in a VTAM Telecommunications System

Application Programs Using VTAM

Each application program that uses VTAM establishes access to VTAM by a special control block. This control block, the access-method control block (ACB), is built by the application program.

The program must identify itself to VTAM prior to using any of VTAM's facilities; this identification is complete when the application program successfuly opens its ACB (that is, initializes its ACB by issuing an OPEN macro instruction) and the application program is connected to VTAM. As long as the ACB is open, the application program can request VTAM services, such as:

- Allocating terminals to the application program
- Transmitting data between a terminal and the application program

A VTAM application program is any program that uses VTAM and its macro instructions to communicate with a terminal. The VTAM application program could be a program that both communicates with a terminal and processes its data. On the other hand, the VTAM application program could be an installation-written service program that handles the I/O requests of other application programs and does not process data. The use of the term *application program* throughout this book is not meant to imply only one type of program.

Note: *Communication between application programs in the host CPU is not supported by VTAM, although such communication may be possible through other components of the operating system.*

Communications Controllers in a VTAM Telecommunication Network

VTAM uses the IBM 3704 and 3705 Communications Controllers to communicate with the remote network. These controllers can be either *locally attached* (that is, connected to the host CPU by a data channel) or *remotely attached* (that is, connected to a locally attached communications controller by a telecommunication line). Communications controllers link VTAM with the remote portions of the network and control the flow of information between terminals and VTAM.

The communications controllers are programmable devices; the program that controls these devices for VTAM is called the network control program/VS (referred to as *NCP* in this publication). The NCP has generation options that allow a communications controller to be operated in different modes. When operated in *network control mode,* the communication controller allows its network resources to be shared. Communications controllers can also be operated in *emulation mode;* when operated in this mode, they emulate the IBM 2701 Data Adapter Unit and the IBM 2702 and 2703 transmission control units. In addition, an NCP can be generated with the partitioned emulation programming (PEP) extension, which allows a communication controller to handle separate telecommunication lines in either network control or emulation mode at the same time. VTAM uses only the network control mode of the NCP, with or without PEP.

The NCP allows some functions previously performed entirely in the host CPU to be performed in the communications controller. Among the functions now provided by the communications controller are:

- Controlling lines
- Controlling dynamic buffering
- Deleting and inserting line control characters
- Detecting machine checks
- Detecting permanent line errors

- Gathering line statistics
- Activating and deactivating lines
- Closing down the network
- Handling recoverable errors
- Providing error statistics to VTAM

In addition, for start-stop and BSC terminals:

- Translating character codes
- Stamping dates and times

By performing these functions outside the host CPU and by allowing much of the network traffic to be controlled in the network, the NCP conserves central computing resources.

Although these activities are actually performed in the communications controllers, they are controlled by VTAM. For example, when the NCP is to deactivate a line, the command to deactivate comes from VTAM. VTAM, in turn, is controlled by the installation's definition of the system, by the network operator, and by the VTAM application programs.

Terminals in a VTAM Telecommunication Network

VTAM supports remotely attached terminals (that is, terminals connected to a communications controller by a telecommunication line) that use the following line disciplines:

- Synchronous data link control (SDLC)
- Start-stop
- Binary synchronous communications (BSC)

The type of line discipline used depends upon the terminal. Appendix A lists remotely attached devices supported by VTAM for each line discipline.

In addition to supporting remotely attached terminals, VTAM provides telecommunication support for the local attachment of 3270 terminals and 3790 terminal systems through a data channel to the host CPU.

Figure 2-3 shows the types of terminals that VTAM supports.

SNA Terminals

Each SNA terminal is supported remotely by VTAM using SDLC switched or nonswitched lines or locally (through a channel interface). To VTAM, an SNA terminal is either a physical unit (PU) or a logical unit (LU) associated with a physical unit. The physical units and logical units perform functions defined by SNA. In general, a physical unit corresponds to a controller or control unit that is attached to a line or channel; a logical unit corresponds to an addressable unit of logic within that controller or control unit. The relationship of the logical unit with the other components in an SNA terminal product depends upon the particular product. See the programming publication for the terminal product for a description of that relationship.

To a VTAM application program, an SNA terminal is a logical unit; the program is not aware of the physical unit with which a logical unit is associated. Before a VTAM application program begins to communicate with a logical unit, VTAM must communicate with the associated physical unit.

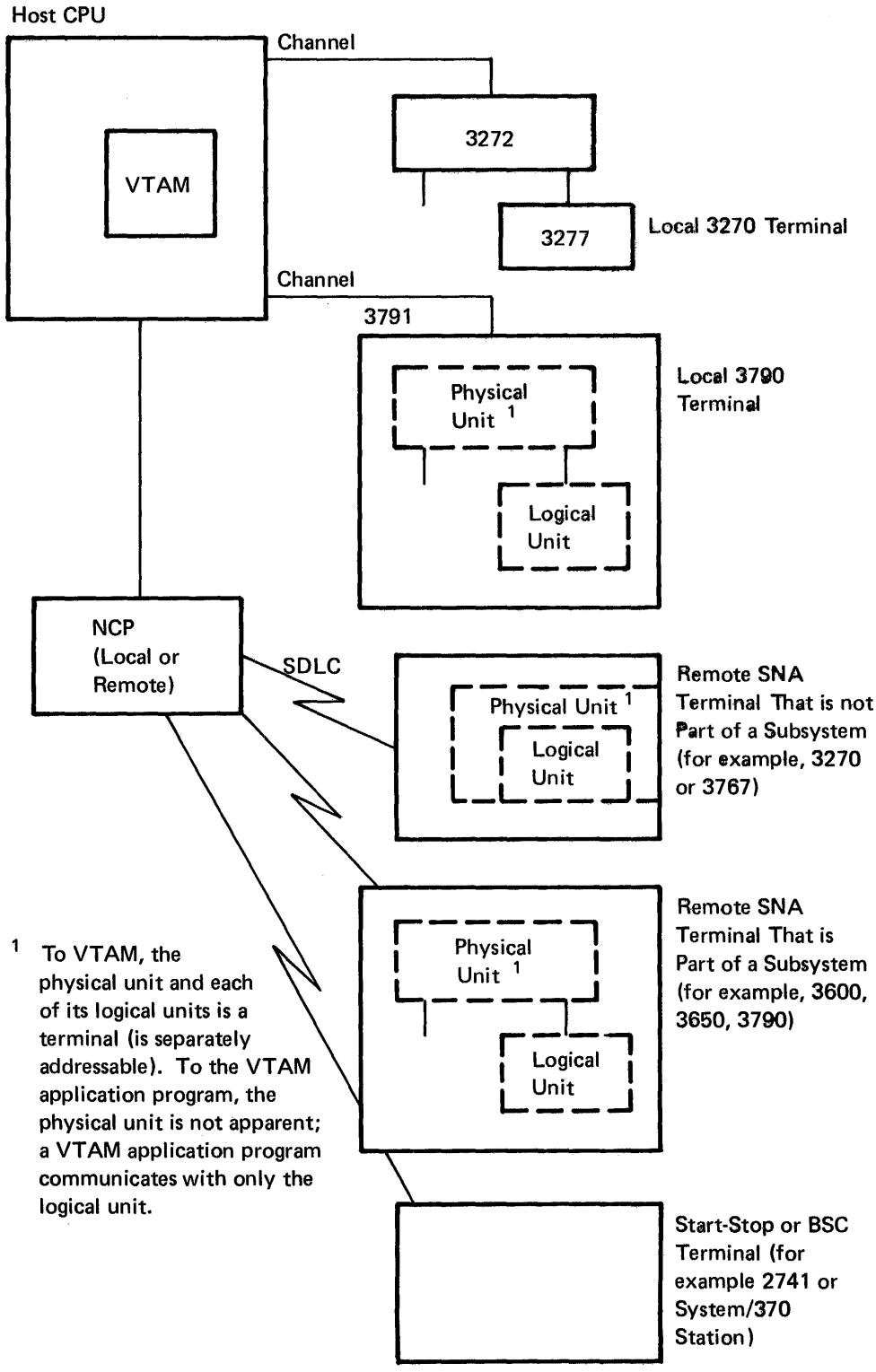


Figure 2-3. Types of VTAM Terminals

Local 3270, BSC, and Start-Stop Terminals

VTAM supports remotely attached BSC and start-stop terminals (such as the IBM 3780 Data Communication Terminal and the IBM 2741 Communications Terminal) and locally attached 3270 Information Display System terminals (displays or printers attached to a 3272 cluster control unit that is attached to a host CPU by a data channel). With these devices, there is a one-to-one relationship between the VTAM terminals and the physical devices; a 2741 is a terminal to VTAM and also a physical end of a network.

For additional information pertaining to local 3270, BSC, and start-stop terminals, see Chapter 8 and Appendix A.

Distrubuted Function

VTAM is responsible for the transfer of data between the elements in only the VTAM portion of a telecommunication system. VTAM application programs and terminals mark the limits of the VTAM system. Control of the data flow beyond these limits is the responsibility of elements at those limits.

In a VTAM system, control functions may be distributed among elements, rather than concentrated in the host CPU. Some control functions have been distributed to elements such as NCPs in communications controllers, physical units, and logical units. VTAM's management of the system:

- Uses the NCP to perform activities such as line-scheduling and polling so that much of the network control and host CPU processing can be performed simultaneously.
- Relies on processing capabilities of logical units and remote stations so that the composition and function of the network beyond these elements are transparent to VTAM.

In effect, the system under VTAM's direct control is only a subset of the total telecommunication system; the remainder of the system is controlled by nodes at the edge of the VTAM system. But, while VTAM directly controls only a subset, the access method can be used to coordinate the activities of the entire system. The remainder of this publication describes VTAM's operation within that subset. The terms *telecommunication system* and *telecommunication network* are used in the remainder of this publication to refer to only the areas controlled by VTAM, unless stated otherwise.

Data Flow through a VTAM Telecommunication System

VTAM manages the flow of data:

- Between itself and the application program
- Between the CPU (using the I/O facilities of the operating system) and the communications controller
- Between the communications controller (using the facilities of the NCP) and the terminal

In a telecommunication system that contains SNA terminals associated with a subsystem or remote stations, there are additional communication connections. Figure 2-4 shows the data flow for a VTAM system in which the VTAM terminal is a logical unit.

Note that VTAM manages only part of the actual data transfer between the application program and the VTAM terminal (the logical unit in this case). The logical unit is responsible for the transfer of data between the IBM 3601 Finance Communication Controller and the physical input/output devices (3604 and 3610, in this example), which mark the end of the physical network.

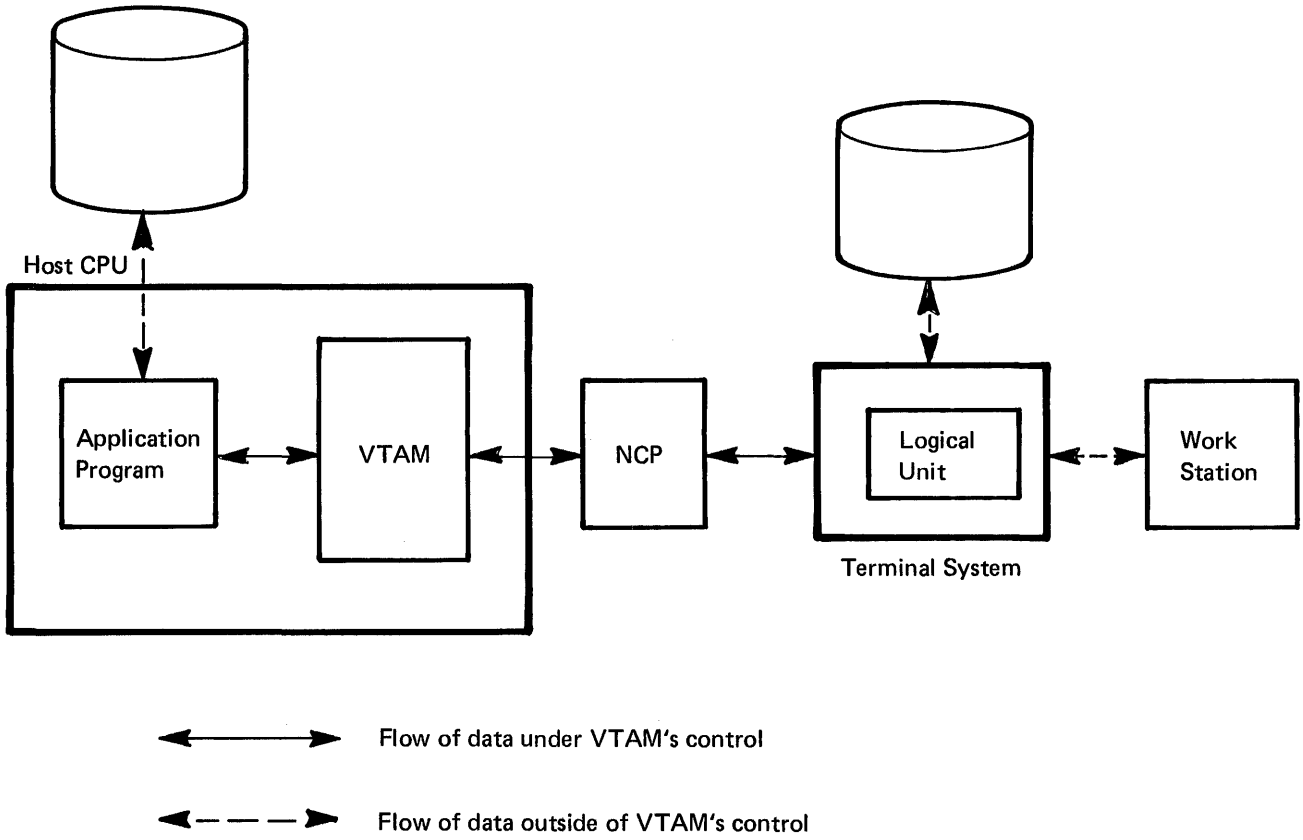


Figure 2-4. Data Flow Through a VTAM Telecommunication System

The application is responsible for any data transfer activity in which the application program uses auxiliary storage devices. (The application program can use an access method such as IBM's Virtual Storage Access Method (VSAM) for access to auxiliary storage of the host system.)

Sharing Resources—An Introduction

Because VTAM allocates network resources, it permits parts of the network to be shared among the application programs executed in the host CPU. Shared resources include:

- Communications controllers, physical units, and lines, which may be used by more than one application program. Actually, an application program is unaware of communications controllers, physical units, and lines; it communicates only with terminals. By sharing these items, several application programs may communicate with different terminals on the same multipoint line. Also, the terminals on a single multipoint line may communicate with any of the application programs using VTAM.
- Terminals, which may be used by more than one application program. Any one terminal may communicate with any one of the application programs that is using VTAM. However, once a terminal and an application program are connected, the terminal can communicate with only that application program until released by the program.

Figure 2-5 shows how resources can be shared in a VTAM telecommunication network. Through VTAM, TCAM application programs can share resources in the same manner as VTAM application programs. For more details on resource sharing through VTAM, see "Sharing Telecommunication Resources," in Chapter 7.

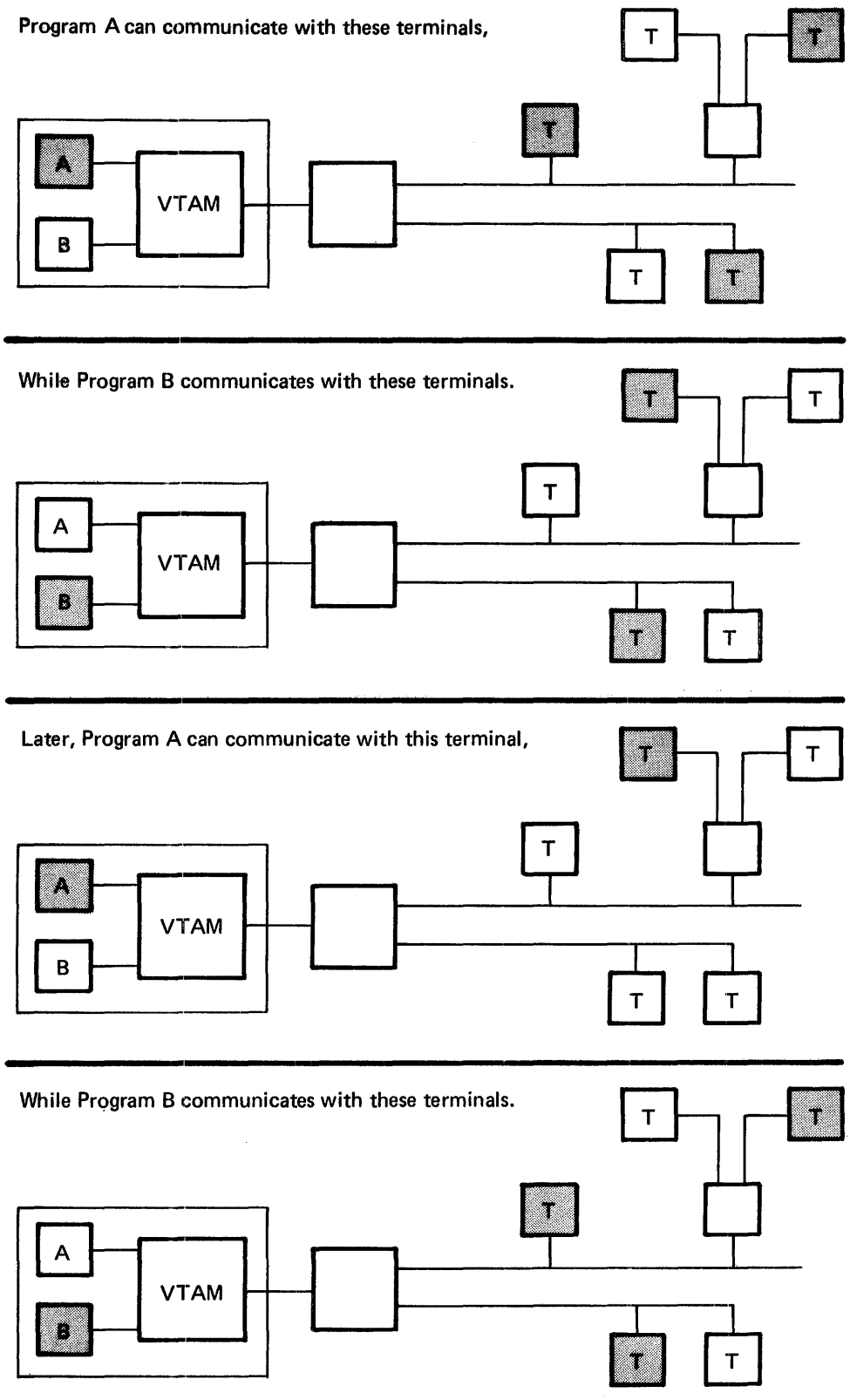


Figure 2-5. Sharing Resources

Overview of VTAM

This section describes the relationship among the major elements of a VTAM system and also introduces concepts that must be understood for proper planning and use of VTAM. These concepts are expanded later in this manual.

Four Views of a VTAM System

To understand VTAM, a user should know what the system looks like from four viewpoints. These viewpoints are shown in Figure 2-6, which depicts the system: (a) as seen in terms of its physical components, (b) as seen by the operating system, (c) as seen by VTAM, and (d) as seen by application programs.

Section A of Figure 2-6 shows a possible physical configuration of the system. A VTAM system includes a central processing unit (referred to as the host CPU) with a system console (labeled S1). The system console is used to enter VTAM operator commands (called network operator commands) to control the telecommunication system. Also attached to the host CPU are auxiliary storage devices that contain data sets used by VTAM.

The network shown in Section A includes a locally attached 3270 terminal (labeled L1), a locally attached 3790 terminal (labeled L2), and a locally attached communications controller. Attached to the local communications controller are three terminals (labeled T1, T2, and T3) and a remote communications controller. T1 is attached on a point-to-point line. T2 and T3 are logical units (in a 3600 system). Attached to the remote communications controller are three terminals (labeled T4, T5, and T6). T4 is attached on a point-to-point line; T5 and T6 are attached on the same multipoint line.

A telecommunication system has a definite physical configuration, but its definition and uses are different for the operating system, VTAM, and application programs. Sections B, C, and D of Figure 2-6 depict these differences.

Section B of Figure 2-6 depicts the telecommunication system as viewed by the operating system.

Support is generated in the operating system for only the system console, the auxiliary storage devices, and the locally attached devices of the telecommunication network. Remote device support need not be generated at system generation if these attachments are to be used only through VTAM. Such support is generated within VTAM, as explained later.

The number of auxiliary storage devices used by VTAM depends upon data requirements that, in turn, are influenced by factors such as the size and complexity of the telecommunication system. In general, data used by or generated by VTAM falls into one of three categories as shown in Section B of Figure 2-6:

VTAM libraries which contain VTAM load modules, descriptions of the telecommunication system, and operational specifications of the installation

NCP libraries, which contain NCP load modules and dump records

RAS (reliability, availability, and serviceability) libraries, which contain records to assist in error recording and maintenance of the VTAM system

VTAM and NCP libraries include VTAM, NCP, and operating system data sets; most of the library requirements for RAS involve operating system data sets. The composition and organization of these libraries depend upon the operating system under which VTAM is being executed. (See Chapter 7 for details on operating system requirements and on data set requirements for VTAM.)

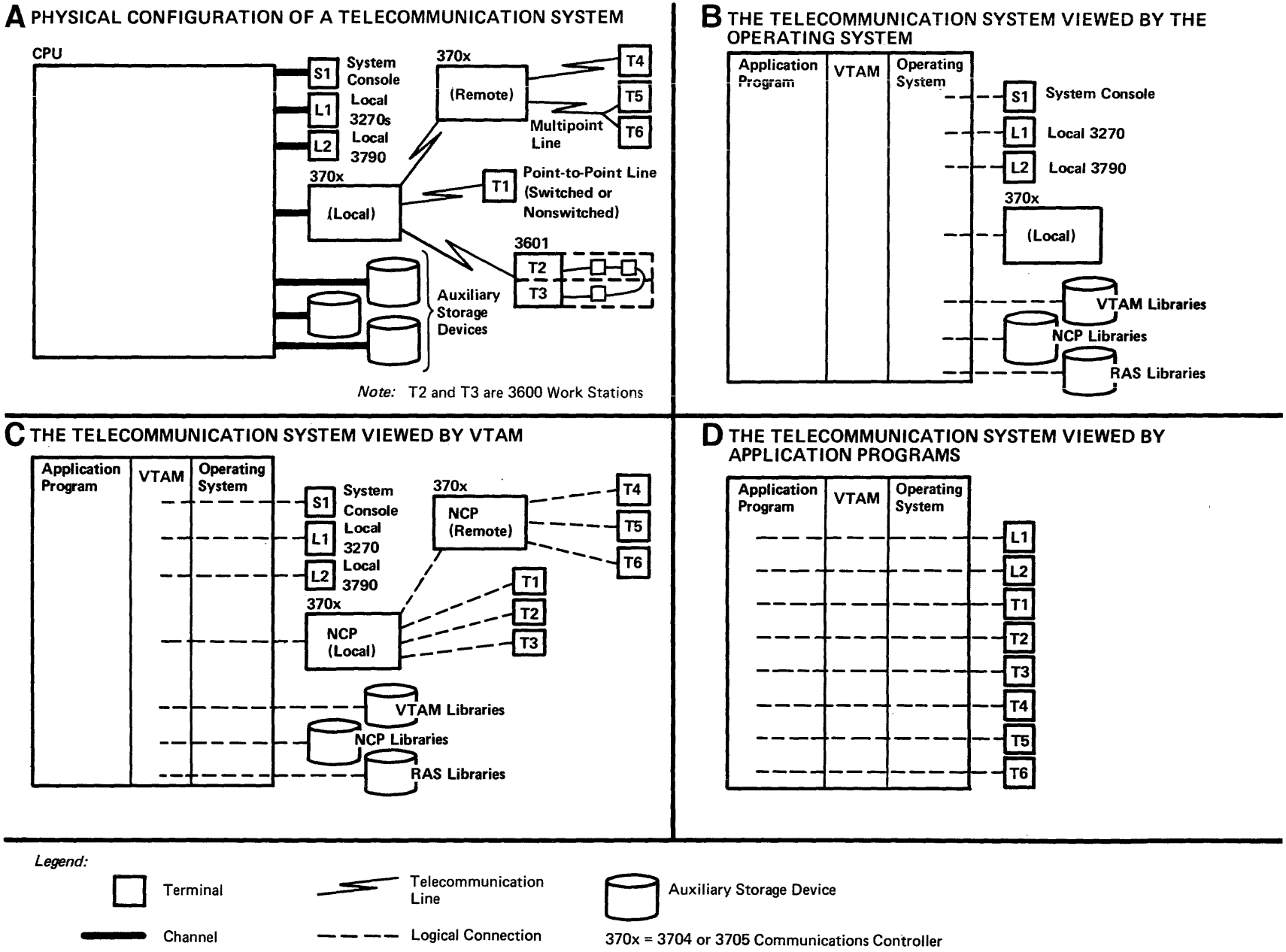


Figure 2-6. Four Views of a VTAM Telecommunication System

Section C of Figure 2-6 depicts the telecommunication system as viewed by VTAM.

The host CPU must contain the operating system (DOS/VS, OS/VS1, or OS/VS2), VTAM, and one or more application programs. To VTAM, an active application program is an open (that is, an initialized) access method control block (ACB). As shown in Section B, all locally attached devices are initially "owned" by the operating system, but (as indicated in section C) when VTAM is started and begins activating parts of the telecommunication network, VTAM acquires the use of these devices. Some of the devices are allocated explicitly to VTAM, some are allocated implicitly, and others are used by VTAM but remain allocated to the operating system.

Devices that are explicitly allocated to VTAM include the local 3270, the local 3790, and the local communications controller. These devices are explicitly allocated because support for these devices is generated in the operating system, and this support is extended to VTAM. When one of these devices is allocated to VTAM, VTAM becomes its sole "owner"; the only access to the device is through VTAM. Locally attached devices not allocated to VTAM remain available for allocation by the operating system to other, non-VTAM users.

The remote attachments (terminals T1 through T6, the remote communications controller, and the 3601 controller in Figure 2-6, Section C) are implicitly allocated to VTAM. They are implicitly allocated because the only access to these devices is through the local communications controller, and VTAM controls access to this controller. *Note:* In the case of an NCP with PEP, VTAM only controls the access to that part of the remote network serviced by the network control mode of the NCP.

Also allocated to VTAM are the data sets on the auxiliary storage devices. These data sets contain the VTAM libraries, the NCP libraries, and some of the RAS libraries. RAS libraries not allocated directly to VTAM are used by VTAM through the RAS facilities of the operating system.

The system console is used by VTAM but not actually allocated to VTAM. Communication is established between VTAM and the network operator through this console. The network operator enters VTAM commands through this console, and VTAM transmits messages to the network operator at this console.

As noted previously, a primary purpose of VTAM is to provide the communication link between application programs and terminals. VTAM uses the operator services of the operating system to communicate with the network operator at the system console; it uses the operating system's data management services and, in some cases, its RAS facilities, to gain access to the libraries on the auxiliary storage devices; but VTAM directly handles communication with terminals in the telecommunication network and with application programs in the host CPU are handled directly by VTAM.

Section D of Figure 2-6 shows the telecommunication system as viewed by the application program. This view results from VTAM's ownership of all elements in the network and from the way VTAM allocates the use of these elements. VTAM connects application programs to only terminals; the other, intermediate, elements are allocated only for the time needed to satisfy a specific transmission request.

As shown in Section D of the figure, application programs connect with terminals and need not be concerned with intermediate connections such as channels, communications controllers (and NCPs), and telecommunication lines. Application programs are also not directly concerned with the system console used by the network operator or with the VTAM, NCP, or RAS libraries.

VTAM in Operation

To operate VTAM requires defining the telecommunication system, controlling the activities of VTAM from the network operator's console, and executing application programs that use VTAM for data transmission.

Defining the system to VTAM involves identifying and describing all of the elements in the system. These element descriptions are collected and filed as members in OS/VS, or books in DOS/VS, in a VTAM library. Activating an element in VTAM includes using the description to establish a definition of, and control information for, the element. VTAM uses this information to control the allocation of the element.

When VTAM is started, its initiation function establishes the telecommunication system according to the specifications of the installation. Once VTAM has been started, active application programs can connect with active terminals in the telecommunication network. As long as a terminal is connected to an application program, it can communicate with that application program.

To form a connection with a terminal, an application program must first identify itself to VTAM. Once identified, the application program can request connection to a specific terminal (or list of terminals). An application program can either acquire or accept a connection. When it acquires a connection, the application program initiates the connection; when it accepts a connection, the connection is initiated by a logon. When a logon is requested, the terminal is "queued" to the application program; that is, the connection process is initiated but not completed. The application program must accept the terminal to complete the connection. Connection is made to the terminal, not the line. When the connection request is completed, the application program is able to transmit data (by input/output requests) to the terminal.

In transmitting data to a terminal, the data is moved from the application program's output data areas to VTAM buffers. VTAM then transmits the data to the terminal (through the NCP for remotely attached devices).

Input from the terminals travels the same (but reverse) route. The transmission moves from the terminal to VTAM (through the NCP for remotely attached devices). VTAM then moves the information to the application program input areas.

When an application program no longers needs a terminal, it can disconnect from the terminal. VTAM can then reallocate the terminal to another application program.

When the telecommunication system is to be closed down, VTAM's termination function enables the installation to terminate VTAM processing in an orderly manner and to cease telecommunication activity.

From the time that VTAM is started to the time it is terminated, the VTAM network operator facilities enable the installation to control and monitor the telecommunication system. Most modifications to the network can be made dynamically, without terminating VTAM.

The steps involved in installing an operating telecommunication system with VTAM are discussed in "Installing a VTAM System," below.

Installing a VTAM System

To install a VTAM system, it must be created, procedures should be defined for using it, the active system must be controlled, and application programs must be designed and coded for the host CPU. These steps are discussed below.

Creating the System

A VTAM telecommunication system is created by generating VTAM into the operating system, defining the network to the communications controllers and to VTAM, and defining special VTAM facilities.

Generating VTAM is part of operating system generation. Defining the network to VTAM is a separate process of identifying and describing the network and then filing these definitions in a VTAM library. Defining special VTAM facilities includes coding exit routines that perform functions such as checking the validity of connection requests between application programs and terminals, collecting information, and structuring VTAM's logon facility to the installation's specifications.

See Chapter 3 for detailed information on using VTAM to create a telecommunication system.

Controlling the Network

VTAM enables a network operator to dynamically control the telecommunication network. The network operator can start and stop VTAM, monitor the activity of the telecommunication system, activate and deactivate network elements such as data links, physical units, and logical units, and start and stop specified VTAM facilities. To perform these functions, the network operator has a set of VTAM commands. The operator uses these commands to monitor and dynamically configure the network as it was defined.

See Chapter 4 for detailed information on the responsibilities and actions of the network operator and for a description of the VTAM network operator facilities. Chapter 4 also contains detailed information on activation of network elements.

Designing VTAM Application Programs

VTAM enables application programs to request connection with specific terminals and to request the transfer of data between the applications and their connected terminals. VTAM also provides facilities for application programs to process requests synchronously or asynchronously. Other facilities include exit routines that are scheduled upon the completion of specified events and an extensive error notification scheme.

Chapter 5 introduces the VTAM facilities available to the application program, and it provides suggestions on designing VTAM application programs. It also provides more detail on VTAM concepts pertinent to application programs—such as connection, communication, and synchronous and asynchronous processing.

Establishing Procedures for Using the System

Once a VTAM telecommunication system has been started, it is available to application programs, terminal operators, and the network operator. To ensure that the telecommunication system is used effectively and efficiently, the installation needs to establish procedures for users of the system and controls that monitor these procedures.

Procedures should be established for the network operator for starting, stopping, and manipulating the VTAM system. The network operator needs to know how and when to activate and deactivate nodes and specific VTAM functions. The network operator also must know what to do when error conditions are encountered and what action to take to avoid unnecessary downtime. (These actions might include responding to error messages, collecting status information, or correcting the problem.)

The application programmer needs to know the conventions to be followed when connecting with VTAM and with terminals. Procedures should also be established for the interaction between the application program and the rest of the system. Such procedures might encompass passing terminal connections between application programs and reacting to system closedown.

The terminal operator may need to know how to log on to and log off from application programs.

Controls should be established to ensure that only authorized users can gain access to VTAM resources. VTAM facilities can be used to control connection to VTAM and between application programs and terminals. Facilities are also available to restrict the use of certain VTAM functions to only authorized users and to protect confidential data.

Chapter 6 discusses the RAS capabilities of VTAM. Chapter 7 describes various VTAM planning considerations. The exact procedures defined by an installation depends upon the needs and requirements of the installation itself coupled with the functions and facilities of VTAM as addressed in this publication.

CHAPTER 3. CREATING A TELECOMMUNICATION SYSTEM WITH VTAM

This chapter describes in general how a VTAM telecommunication system is created to meet installation requirements. The *VTAM System Programmer's Guide* for each operating system describes these procedures in detail.

Creating a VTAM telecommunication system consists of:

- Defining VTAM and local devices to the operating system
- Generating a network control program (NCP)
- Defining the network to VTAM
- Defining connection and disconnection procedures for terminals in the network
- Coding and including installation exit routines
- Defining start options (the status of the system when it is initialized and the size, thresholds, and other characteristics of VTAM storage pools to be used for incoming and outgoing data and for control blocks)

Creating a VTAM telecommunication system may also include the defining of network operating procedures, described in Chapter 4, and the writing of VTAM application programs, described in Chapter 5.

Figure 3-1 summarizes the steps in creating a VTAM telecommunication system that are discussed in this chapter. Some of the steps in Figure 3-1 are related to other steps. For example, step C, defining the network to VTAM, consists not only of describing the structure of the network to VTAM but also of relating each described logical unit to an associated logon mode table. Thus, step C also involves step E, defining connection and disconnection procedures. The steps in Figure 3-1, based on the separate sets of statements that may be required in setting up the system, are the basis for the organization of this chapter and are a convenient way to look at the processes involved in planning and setting up a VTAM system.

Defining VTAM and Local Devices to the Operating System

During system generation, VTAM modules are generated and included in the operating system. To generate the VTAM modules and the required support in the operating system, the following are specified in the input stream for the first stage of system generation:

- VTAM is specified as a parameter of the TP operand of the SUPVR macro instruction for DOS/VS, or VTAM is specified as a parameter of the ACSMETH operand of the DATAMGT macro instruction for OS/VS.
- Control unit and device statements are specified only for locally attached devices to be used by VTAM; that is, only for locally attached 3270s, locally attached 3790s, and locally attached communications controllers.

  Note: *Remotely attached devices are not specified during system generation. These devices are specified and described during network definition. Remember that network definition can be done online, without disrupting other jobs in the operating system. (See "Defining the Network to VTAM," below, for a description of network definition.)*

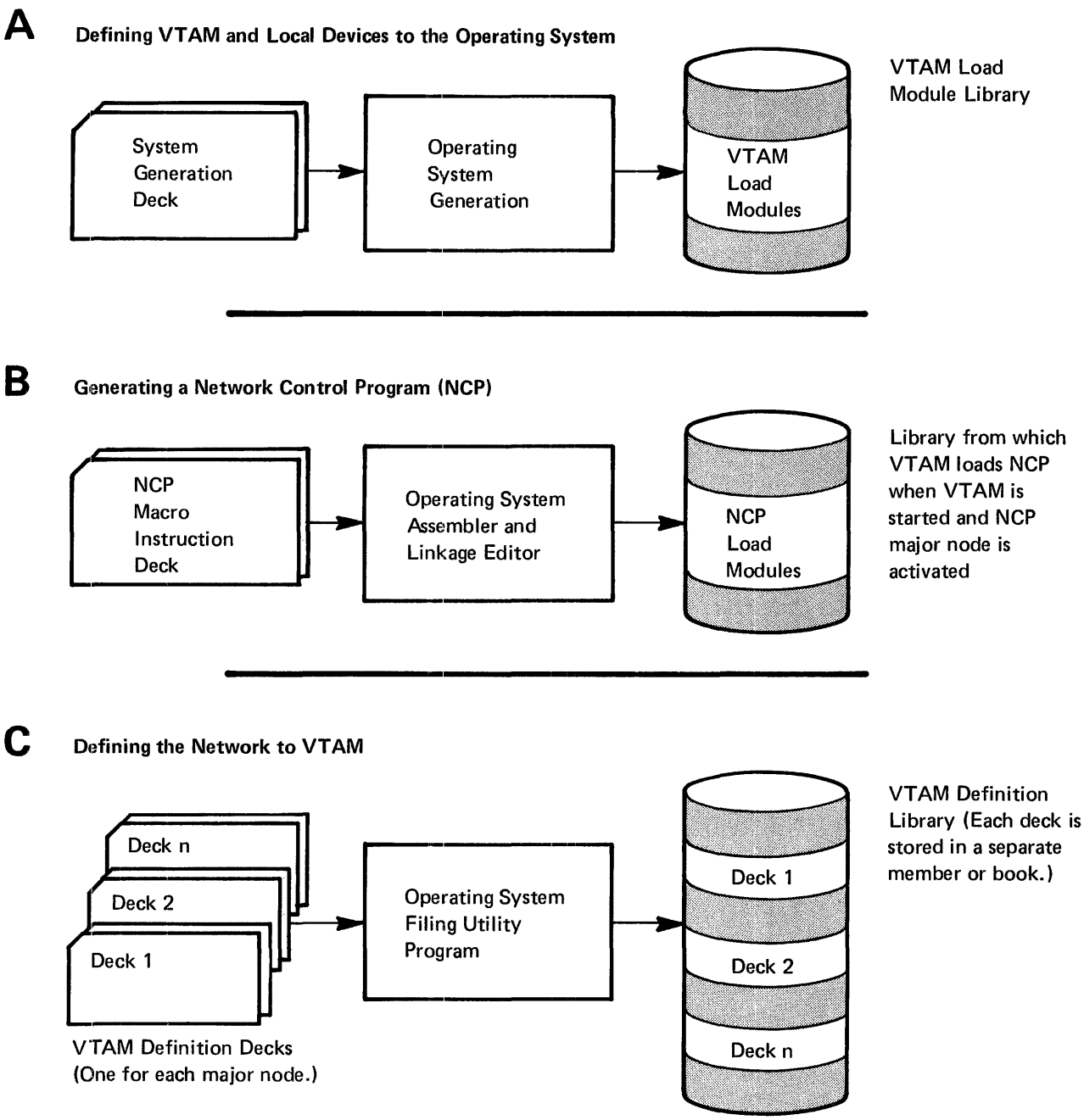
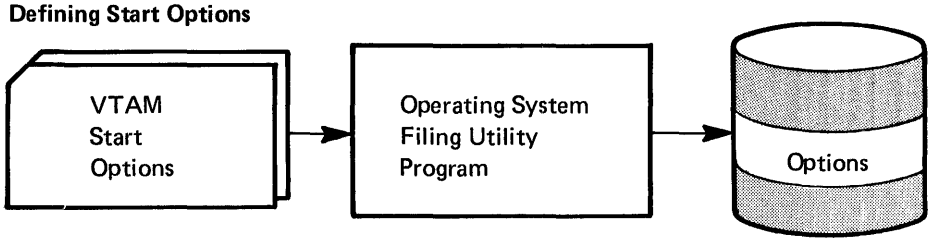


Figure 3-1 (Part 1 of 2). The Steps in Creating a VTAM Telecommunications System

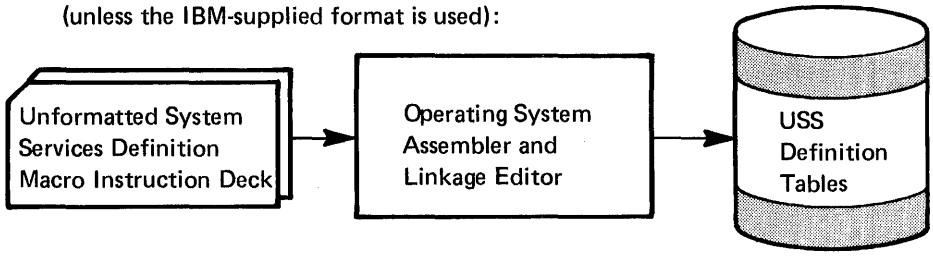
D **Defining Start Options**



VTAM Definition Library (Multiple set of options can be filed, each in a separate member or book.)

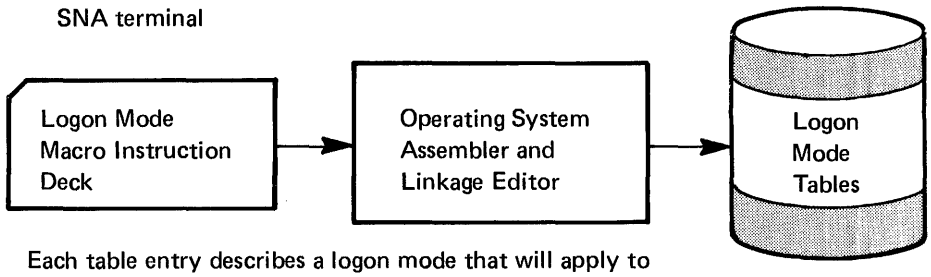
E **Defining Connection and Disconnection Procedures**

- For SNA terminals using character-coded (USS) logons and logoffs (unless the IBM-supplied format is used):



VTAM Load Module Library (More than one table can be filed.)

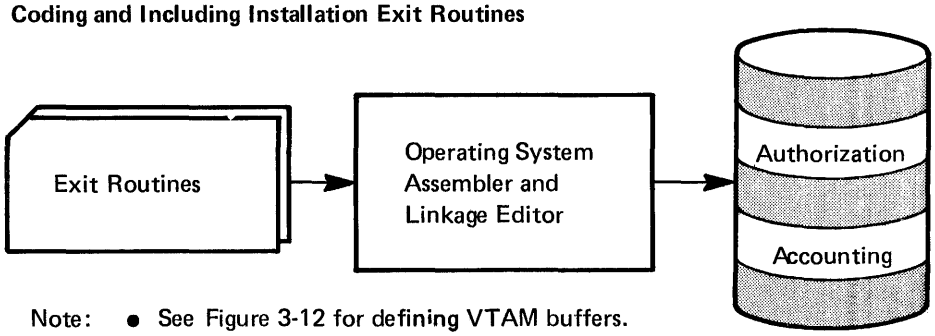
- To define the logon mode (set of session parameters) for each SNA terminal



VTAM Load Module Library (More than one table can be filed.)

Each table entry describes a logon mode that will apply to a logical unit defined as using that table.

F **Coding and Including Installation Exit Routines**



VTAM Load Module Library (These two routines replace IBM-supplied modules.)

Note:
- See Figure 3-12 for defining VTAM buffers.
- See Figure 7-4 or 7-5 for a summary of data sets or files that are created

Figure 3-1 (Part 2 of 2). The Steps in Creating a VTAM Telecommunications System

Additional operating system support for VTAM can be included at system generation. See "Operating System Requirements," in Chapter 7, for details on operating system support.

Generating a Network Control Program (NCP)

If the VTAM telecommunication system includes remote logical units or terminals, they must be connected on telecommunication lines to a 3704 or 3705 Communications Controller that contains a Network Control Program. In creating this system, an NCP must be coded, using NCP macro instructions. The macro instructions are assembled and the output is then filed so that it can be loaded into the communications controller when the VTAM system is started. (The NCP is loaded automatically as the result of starting VTAM.) The macro instructions required to code an NCP are described in *NCP Generation*; this publication can be used with the *VTAM System Programmer's Guide* to write a set of statements both to generate an NCP and to define that NCP's configuration and functions to VTAM.

VTAM considerations when writing the set of NCP statements are described further in this chapter in "Defining NCPs for Communications Controllers."

Defining the Network to VTAM

Defining the network to VTAM consists of describing the network configuration and its characteristics to VTAM. These descriptions are coded in VTAM definition statements; the coded definitions are then filed in the VTAM definition library.

As part of the network definition, the five types of major nodes in the telecommunication system are defined to VTAM:

- NCPs for locally or remotely attached communications controllers, including their attached logical units, physical units, and terminals
- Sets of logical units and physical units on switched lines
- Sets of logical units and physical units attached locally
- Sets of locally attached 3270 terminals
- Sets of application programs that use VTAM

The definition of each node is filed separately (as a member in OS/VS or as a book in DOS/VS) in the VTAM definition library. When VTAM activates a major node, it uses the filed definition of that node as a description of the node's configuration. (In OS/VS, if the major node has previously been activated, a description of the node configuration is taken from a copy of the previously built table describing the major node.) A major node can be defined anytime following system generation but prior to the first use of that major node by VTAM. Before defining the major nodes to VTAM, it is necessary to understand VTAM's major node and minor node structure.

The Major and Minor Node Structure

Major and minor nodes are the controllable elements of the VTAM network. Network operator commands can be issued to activate or deactivate each major or minor node defined to VTAM. (The terms "major node" and "minor node" are unrelated to such terms as "host node" and "communications controller node" used in SNA publications.)

The installation uses VTAM definition statements to identify all major and minor nodes. These statements not only identify an element as a major or a minor node but also place the node within a hierarchical structure of controllable elements. All major or minor nodes are addressed symbolically; the installation assigns the symbolic names at VTAM definition.

A major node is a set of controllable elements (minor nodes) in the VTAM network. Each major node structure has the general form:

Major Node
    Minor Node 1
    Minor Node 2
    Minor Node 3
    .
    .
    .
    Minor Node n

Thus, each major node structure can be controlled as a whole or portions of it can be controlled through the minor nodes within each major node.

The name of a major node is the name of a member in the VTAM definition library, SYS1.VTAMLST (OS/VS), or of a book in the source statement library (DOS/VS). This member or book contains the statements that define that major node. The statements can be any one of the following:

- The definition statements defining one or a set of application programs
- The definition statements defining one or a set of locally attached 3270s
- The definition statements defining one or a set of local 3790 physical units and logical units
- The definition statements defining the configuration known to a locally or remotely attached communications controller's NCP
- The definition statements defining one or a set of SNA physical units and logical units on switched lines

One or more minor nodes can be defined for each major node. The type of minor nodes used in a major node depends on the type of major node being defined, as described in the following discussion. The name of a minor node is the name assigned to the VTAM statement defining it.

Node Structure for Application Programs

One or more major nodes can be defined for application programs. Each major node can contain one or more minor nodes. Each minor node is represented by an APPL definition statement.

The application program nodes that can be defined and addressed are therefore major nodes and individual programs represented to VTAM as minor nodes. Each major node can contain more than one minor node, but each minor node represents only one application program.

Node Structure for Local 3270s

The node structure for locally attached 3270s is similar to that of application programs. More than one major node can be defined for local 3270s. Each major node begins with an LBUILD definition statement. Each major node can contain one or more minor nodes. Each minor node is represented by a local definition statement.

Node Structure for Physical Units and Logical Units Attached Locally

One or more major nodes cna be defined to VTAM as sets of attached physical units and logical units. Each major node contains two types of minor nodes. A minor node that is a physical unit is represented by a PU definition statement; a minor node that is subordinate to a physical unit is represented by an LU statement.

Node Structure for Physical Units and Logical Units on Switched Lines

One or more major nodes can be defined to VTAM, each as a set of physical units and logical units on switched lines. Each switched major node contains two types of minor nodes: physical units, each represented by a PU definition statement, and, for each physical unit, one or more logical units, each represented by an LU definition statement.

Node Structure for Local NCP

One or more major nodes can be defined for each locally attached communications controller used by VTAM. Only one of these major nodes can be active at one time, however, for a given communications controller. A major node for a local communications controller and its subordinate minor nodes are defined by the definition statements filed for an NCP.

The types of minor nodes for an NCP major node are:

- Groups of lines
- Lines
- Other types of minor nodes, depending on whether SNA terminals and terminal systems are being defined or whether start-stop and BSC terminals are being defined

Each minor node in an NCP is identified by a statement in the set of NCP definition statements. Each group of lines is identified by a GROUP statement. Each line in a group is identified by a LINE statement.

The other types of minor nodes for SNA terminals and terminal systems are:

- Physical units
- Logical unit pools (for SNA switched-line networks)
- Logical units

A physical unit minor node identifies: a terminal system controller (such as a 3601 or 3791), a switched-line access to the communications controller, or a remote communications controller. If it identifies a switched-line access or a remote communications controller, it does not have any subordinate logical unit minor nodes. A logical unit pool minor node is used with switched-line networks and identifies the maximum number of logical units that can be active at one time over the NCP's switched lines. Logical unit minor nodes identify logical units attached on nonswitched lines. (Logical units that can become connected on switched lines are identified to VTAM through the SNA switched-line major node definition.)

Depending on the type of minor node, the NCP definition for each minor node statement is:

- A PU statement for a physical unit minor node
- An LUPOOL statement for a logical unit pool minor node
- An LU statement for a logical unit minor node

Note 1: *A PU statement should be used to identify a terminal system controller instead of the CLUSTER statement that was used in VTAM prior to Level 2. However, where a CLUSTER statement has already been used, VTAM recognizes the statement as a PU statement; the statement need not be recoded.*

Note 2: *A PU statement should be used to identify a remote communications controller instead of the INNODE statement which was used in VTAM prior to Level 2. However, where an INNODE statement has already been used, VTAM will recognize the statement as a PU statement. The statement need not be recoded.*

The other types of minor nodes for start-stop and BSC terminals (other than groups and lines) are:

- Ports (for switched-lines only)
- Clusters
- Terminals
- Components

A port minor node identifies a start-stop or BSC switched-line access to the communications controller. A cluster minor node identifies a control unit for certain terminals, such as 3270 terminals. A terminal minor node identifies a terminal, such as a 2740 or a 3277, or a remote station, such as a System/3. A component minor node identifies a terminal component, such as a printer or a card reader.

VTAM distinguishes between a remote communications controller attached as an extension of the network and as a part of a remote station. See Appendix C for a discussion of this distinction.

The NCP definition statement for each minor node is:

- A TERMINAL statement for a port minor node
- A CLUSTER statement for a cluster minor node
- A TERMINAL or VTERM statement for a terminal minor node
- A COMP statement for a component minor node

Because the minor node structure depicts the physical configuration of a network, an NCP major node can contain more than one group, line, physical unit, or other minor node definition. (Only one LUPOOL definition is possible, however.) SNA minor nodes and start-stop and BSC minor nodes can be defined within the same NCP major node.

Node Structure for Remote NCP

A remote communications controller NCP node structure is the same as a local communications controller NCP node structure. The only exception is that a remote communications controller cannot attach another remote communications controller. See "Node Structure for Local NCP," above, for a description of the NCP node structure.

Using the VTAM Node Structure

To VTAM, major and minor nodes are the network elements that can be addressed, used, and manipulated by application programs through VTAM's application program macro instructions and by the network operator through VTAM's network operator commands.

Before VTAM can address a minor node (such as a single terminal or an application program), its major node must be activated. A major node can be activated either during start processing or, thereafter, by VTAM's VARY command.

In most cases, the activation and deactivation of major nodes are independent of each other. However, a local NCP major node must be activated before its remote NCP major nodes are activated, and major nodes containing a definition of the same minor node or containing minor nodes with the same name cannot be active at the same time.

An application program major node and a local 3270 major node each has a two-level structure (the major node with a single level of minor nodes). The NCPs, on the other hand, can have a structure of several levels.

The hierarchical node structure is made of higher-level nodes and subordinate nodes. For example, an application program minor node is subordinate to the higher-level application

program major node. Likewise, a component node is subordinate to the higher-level terminal node; the terminal node, in turn, is subordinate to nodes above it (such as the line or the group nodes). All minor nodes are subordinate to their higher-level major node.

This hierarchical, or layered, structure is important when attempting to activate nodes because a node cannot be activated until all node layers above it have been activated. See Chapter 4 for details on activating nodes.

With the layered node structure, an installation can activate large segments of the network, then activate and deactivate subsections within each segment as needs change.

A major node for locally attached 3270s contains one or more local 3270 minor nodes (that is, one or more locally attached 3270 definitions); likewise, a major node for application programs contains one or more application program minor nodes (that is, one or more individual application program definitions).

Collecting minor nodes under a major node facilitates system control. With major nodes an installation can activate and deactivate a set of minor nodes at one time. For example, if several application programs are normally executed at the same time, they could all be defined (by an APPL statement for each ACB to be opened) in the same major node. To identify these application programs to VTAM, only the activation request for the major node would be required. In contrast, if each application program is defined as a minor node under a different major node, an activation request is required for each major node.

The same minor node can be defined in more than one major node. (However, only one major node containing a definition of the same minor node can be active at one time.) Also, minor nodes of the same type (for example, local 3270s) need not all be defined in the same major node. It is possible to create several major nodes for local 3270s, each major node containing a subset of all the local 3270s in the system.

The ability to create different combinations of minor nodes can be used not only to control the telecommunication system but also to assist in testing and expanding it.

For example, to help in controlling the network, a local 3270 could be defined in a major node that is activated when activity on the system is low. This major node would define only a subset of all the locally attached 3270s. The terminal could also be defined in another major node (defining all of the local 3270s) that is activated when telecommunication activity is high.

In addition, to help with testing, this same terminal could also be in a third major node that defines some terminals that are to be added to expand the system. This major node would be activated when testing the expanded system, or (after testing is completed) to support that expansion.

For more information on naming nodes, see "Naming Major and Minor Nodes," below. See Chapter 4 for information on VTAM commands and on activating and deactivating nodes.

Naming Major and Minor Nodes

All major and minor nodes must be named. The name of each major node is the name of the member (in OS/VS) or book (in DOS/VS) containing the definition statements for that major node. The name of each minor node is supplied on its definition statement.

Node names provide the means of addressing and using a VTAM system. These names are used:

- During VTAM definition to define the telecommunication system

- In application programs to connect with terminals
- By the network operator to control the VTAM system
- In VTAM messages to identify specific portions of the system
- By, possibly, logical units and operators at start-stop and BSC terminals to log on to an application program

A VTAM system may have a large number of nodes, particularly minor nodes, and may therefore require the generation of many node names. Considering the quantity of names and their extensive use, some care must be taken in choosing, assigning, and disseminating names.

The following rules apply to assigning names:

- Duplicate major node names are not permitted.
- Duplicate minor node names are not permitted in the same major node.
- Two or more major nodes containing duplicate minor-node names cannot be active simultaneously.
- The names NETSOL, TOLTEP, and TRACE are assigned to IBM-supplied facilities and therefore cannot be assigned by the installation to nodes.

Controlling the dissemination of node names provides some control over the security of the telecommunication system. See "Telecommunication Security," in Chapter 7, for more information on using node names to protect the system.

Defining NCP Major Nodes

At least one group of definition statements must be provided to VTAM for each communications controller in the VTAM network. To aid in defining the remote network for a communications controller, the same deck of macro instructions used to generate an NCP can be used as the major node definition by VTAM. Using this deck both for NCP generation and for VTAM network definition requires additions to the NCP generation macro instructions. These additions include statements and parameters that are used only by VTAM. See Chapter 7 for VTAM requirements for an NCP.

An NCP configuration is defined by filing the NCP generation deck separately as a member in OS/VS or a book in DOS/VS in the VTAM definition library. The member name is thereafter used when addressing the NCP through VTAM. Each NCP defined to VTAM is called a major node. See "The Major and Minor Node Structure," earlier in this chapter, for a detailed definition of major node.

If an NCP is modified (regenerated), it must be redefined to VTAM. It is redefined by refiling the altered or new NCP generation deck. In OS/VS, it is also necessary to delete the NCP's corresponding member of SYS1.VTAMOBJ. Thus, remote network configurations can be modified without requiring a new or partial system generation of the operating system.

Figure 3-2 depicts the steps for generating NCP support in a VTAM telecommunication system. The steps are as follows:

1. *Planning the NCP.* Keep VTAM requirements, restrictions, and considerations in mind.
2. *Coding the NCP generation statements.* Include the parameters and definition statements required by VTAM as well as those used to generate the NCP.
3. *Generating the NCP.* Use the statements coded in step 2. Include those parameters and definition statements that are used only by VTAM; NCP generation initially verifies these parameters and statements, though they are not used to generate the NCP.

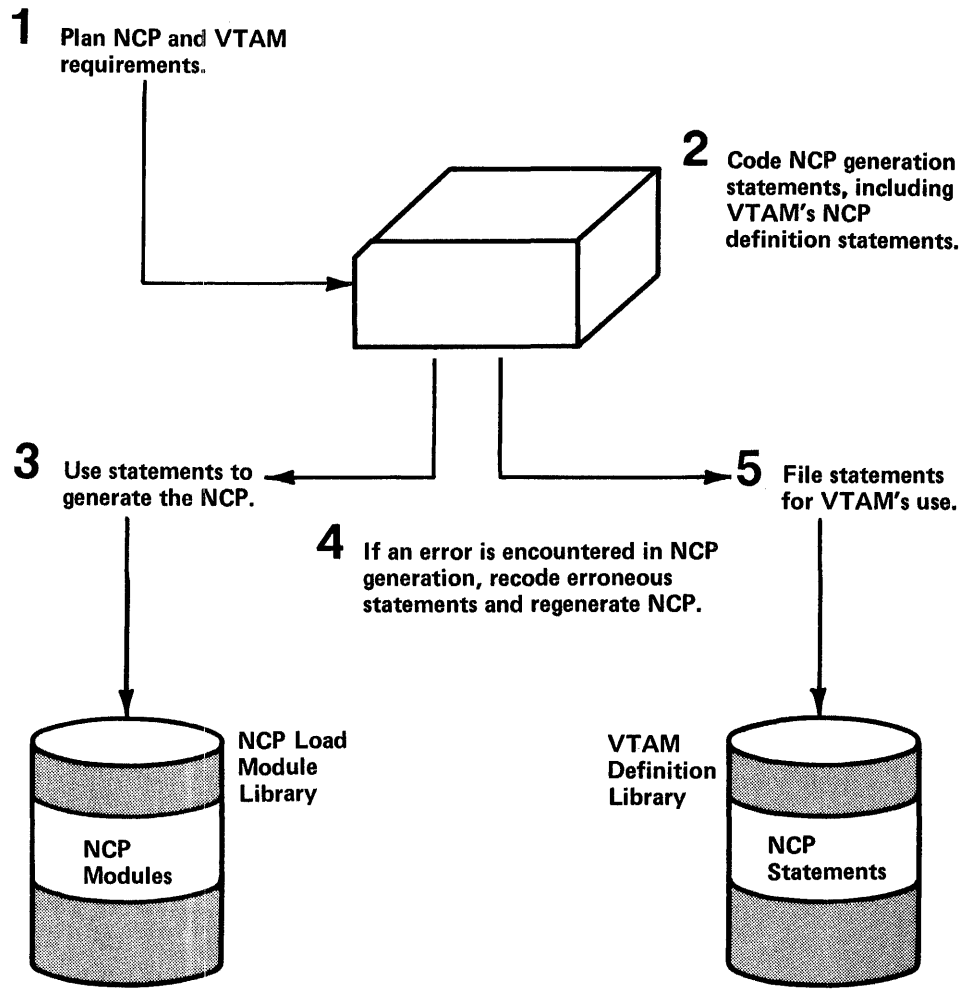


Figure 3-2. Generating NCP Support in a VTAM Telecommunication System

4. *Verifying that the generation is successful.* If the NCP is not generated successfully, correct the generation deck and repeat step 3.
5. *Filing the generation deck.* File the deck (as a member in OS/VS or a book in DOS/VS) in the VTAM definition library. This deck was coded in step 2 and used in step 3. When VTAM activates this NCP, VTAM extracts the information it needs from the filed definition and from the generated NCP itself.

Using the same deck to generate an NCP and to define it to VTAM ensures that the generated NCP agrees with its VTAM definition. The deck is filed on the definition library using a utility of the operating system.

Including the VTAM-only information in the generation deck permits an initial verification of these specifications during NCP generation. Generating the NCP prior to filing the deck ensures that the NCP generated is the one that is defined. If the deck is filed first and an error is encountered in the generation process, the updated deck would have to be refiled.

Note: *The statements used to generate the NCP can be referred to as macro instructions because they are assembled and generate communications controller instructions. As used by VTAM, these same statements are not assembled and do not generate instructions. Therefore, in this publication, they are referred to as statements, not macro instructions.*

Defining Local 3270 Major Nodes

Each locally attached 3270 terminal must be defined to VTAM, either individually or as part of a logical set of locally attached 3270s. Definitions of locally attached 3270s are provided in VTAM's LOCAL definition statements. A LOCAL statement defines one terminal (a printer or a display unit), and each locally attached terminal in a VTAM network must be defined by at least one LOCAL statement.

A logical set of locally attached terminals can be defined by filing the LOCAL statements, one for each terminal in the set, along with an LBUILD statement (as a member in OS/VS, or a book in DOS/VS) in the VTAM definition library. The LBUILD statement identifies a local 3270 major node. A terminal can be included in more than one local 3270 major node as shown in Figure 3-3.

The following information is provided by the LOCAL statements:

- The symbolic name of the terminal.
- The channel and unit address of the terminal.
- The features that are available on the terminal.
- The name of the logon description (interpret table) to be used when analyzing logon requests for the terminal. A terminal's interpret table is also inspected whenever an INTRPRET macro instruction is issued by an application program for that terminal. (See Chapter 8, for a description of defining logons in interpret tables. See "The VTAM Language," in Chapter 5, for a description of VTAM's INTRPRET macro instruction.)
- The name of an application program to which VTAM is to automatically transmit a logon, on behalf of the terminal, whenever the terminal is available for connection. See "Defining VTAM-Initiated Connection," later in this chapter, for a description of automatic logon.
- Whether the terminal is to be considered active or inactive when the logical set of which it is a part is activated.
- The buffer limit for the terminal. See "Defining VTAM Buffering," later in this chapter, for a description of how buffer limits are established using this specification.

Defining Local SNA Major Nodes

A local SNA major node is a set of physical units and their associated logical units that are connected directly with VTAM by a channel interface. In defining a local SNA major node, the following definition statements are used:

- A VBUILD definition statement, which describes the start of a specific local SNA major node
- One or more PU statements, each of which identifies a particular physical unit and describes its characteristics (for example, a default channel unit address with which it may be associated)
- For each PU statement, one or more LU statements that identify a logical unit and define specific characteristics of the logical unit

Defining Switched SNA Major Nodes

A switched SNA major node is a set of physical units and their associated logical units that can be connected on switched lines. Each switched major node is defined and filed (as a member in OS/VS or a book in DOS/VS) in the VTAM definition library. The major node is defined using:

- A VBUILD definition statement, which defines the start of a switched SNA major node definition and describes its characteristics to VTAM, such as the maximum number of unique telephone numbers that can be used by the PATH statements for this major node

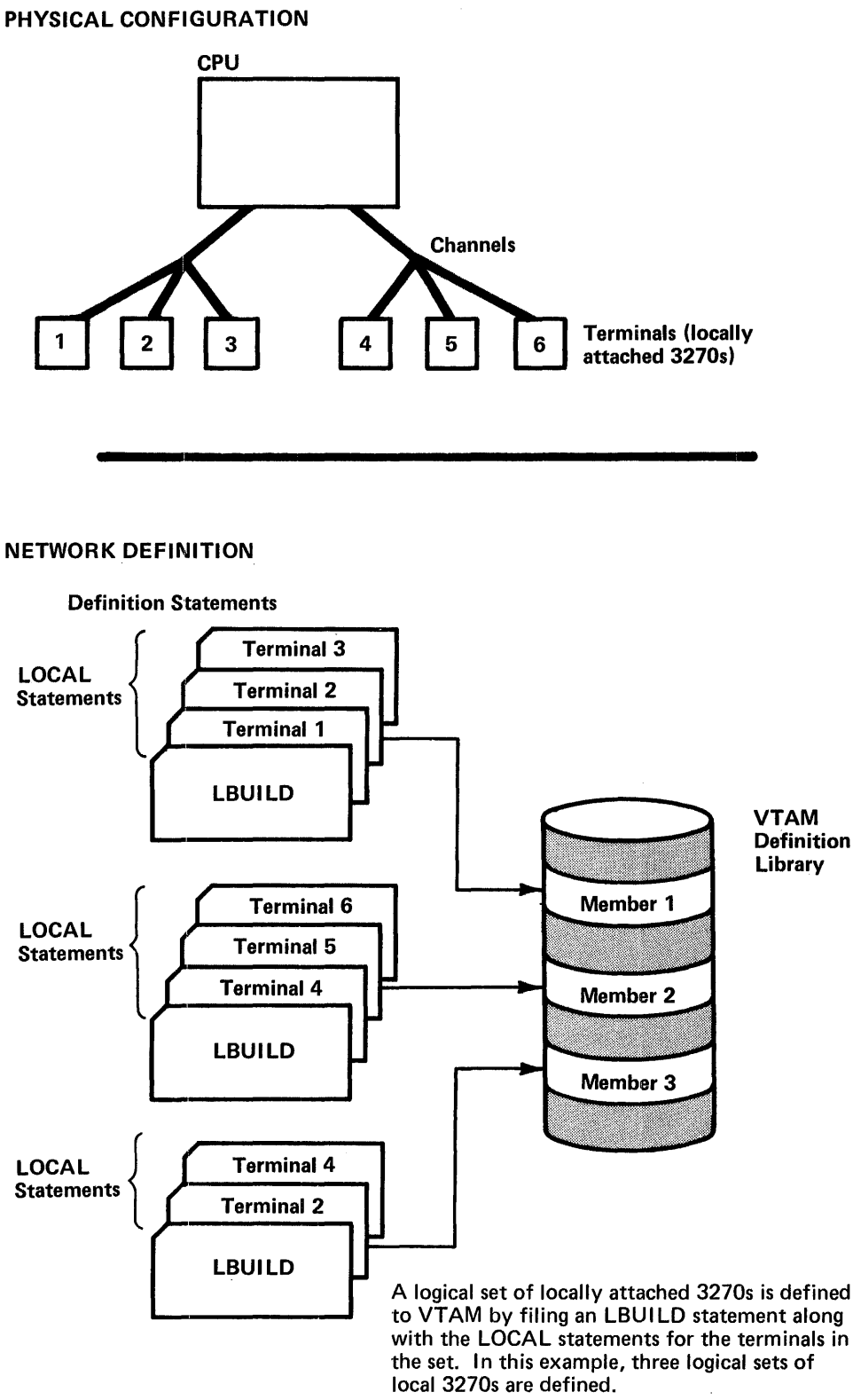


Figure 3-3. Grouping Locally Attached 3270s into Logical Sets

- One or more PU statements, each of which identifies a particular physical unit (by specifying its identification number for VTAM verification) and its characteristics
- One or more PATH statements for each PU, each of which describes how a dial-out operation will be carried out (the telephone number to call and the number of redial attempts to be made)
- One or more LU statements for each PU, each of which describes the specific characteristics of each logical unit associated with the physical unit

Defining Application Program Major Nodes

Application programs must also be defined to VTAM. Each program is defined with an APPL definition statement. The name specified in the APPLID field of the ACB specifies the address of the name stored in the application program that must match the name of an APPL statement. To VTAM, an application program is an open access method control block (ACB).

APPL statements can be filed separately (as members for OS/VS, or books for DOS/VS) in the VTAM definition library, or they can be grouped in various combination and filed as sets.

The APPL definition statement can provide the following information:

- The symbolic name to be located by the APPLID field in the ACB.
- The password to be specified by the application program when the ACB is opened.
- The specifications of those VTAM facilities that an application program using this APPL definition is allowed to use.
- The buffer factor for the application program. See "Defining VTAM Buffering," later in this chapter, for a description of how buffer limits are established using the buffer factor of the application program.

An application program can be authorized (by its APPL statement) to perform each of the following through VTAM:

- Pass connections to another application program. (That is, issue a CLSDST macro instruction with the PASS option.)
- Initiate connection requests for terminals. (That is, issue the OPNDST macro instruction with the ACQUIRE option or issue the SIMLOGON macro instruction.)
- Request input data from start-stop or BSC terminals in blocks instead of in messages or transmissions.

An APPL statement must be defined for each unique ACB, although the same APPL statement can be named by only one open ACB at any one time. Like locally attached 3270s, an application program can be defined as part of more than one logical set of application programs; that is, the same APPL statement can be filed in more than one member of the VTAM definition library. (See Chapter 5 for descriptions of the ACB and the OPNDST, CLSDST, and the SIMLOGON macro instructions.)

A logical set of application programs filed in the same member of the VTAM definition library is called a major node. See "Major and Minor Node Structure," earlier in this chapter, for a definition of major node.

Defining Connection Procedures

Having defined the configuration of a VTAM telecommunication system and its initial status when it is started, an installation must decide when and how the terminals (logical units and local 3270, BSC, and start-stop terminals) in the system are to be connected to VTAM application programs.

Connection Concepts

When a terminal is activated (either when VTAM is started or when the network operator enters a command requesting activation) it is connected to VTAM. VTAM performs any physical preparation required for connection (such as transmitting an SNA Activate Physical Unit command) with no network operator or VTAM application program action. For terminals on switched lines, the active status is logical rather than physical; physical connection to VTAM is not actually established until a dial-in or dial-out operation takes place. Once a terminal is active, a request for connection to a VTAM application program can be made by:

- The terminal (logic associated with a logical unit or a terminal operator)
- VTAM (an automatic connection request)
- A VTAM application program
- The network operator

Except for a VTAM application program's request to acquire a terminal, a request for connection to a VTAM application program takes place in two stages:

1. VTAM receives the request and passes it to the VTAM application program.
2. The VTAM application program, having determined that there is a pending request for connection, either connects or disconnects the terminal to or from itself by making the appropriate request of VTAM.

The VTAM application program issues an OPNDST (Open Destination) macro instruction that specifies OPTCD=ACQUIRE to request acquisition of a terminal. The VTAM application program issues an OPNDST macro instruction that specifies OPTCD=ACCEPT to ask VTAM to accept the connection request. A request to disconnect a terminal is performed by issuing a CLSDST macro instruction. VTAM passes on a request for connection to a VTAM application program either by scheduling the program's LOGON exit routine, if one has been designated, or by completing an OPNDST macro instruction that specifies OPTCD=ACCEPT.

Chapter 5, "Writing a VTAM Application Program," discusses how a VTAM application program connects to terminals.

Defining Terminal-Initiated Connection

An installation can define that a terminal can request connection to a VTAM application program. Defining this kind of connection request requires defining logon messages and formats to VTAM and to the VTAM application program, and to the terminal. This section describes defining these procedures for logical units; see Chapter 8 for a description of defining these procedures for local 3270, BSC, and start-stop terminals.

Types of Logical Unit Logon Requests

VTAM identifies a logon request from a logical unit by either an SNA Initiate Self command or by a specially defined character string. The logon request contains the name of the application program with which connection is requested, an indication of the type of application program activity required (for example, batch or interactive), and data that can be passed to the VTAM application program as the logon message. Figure 3-4 illustrates the kind of information that is included in a terminal-initiated connection request from a logical unit. (The exact format of a logon request or other kind of SNA

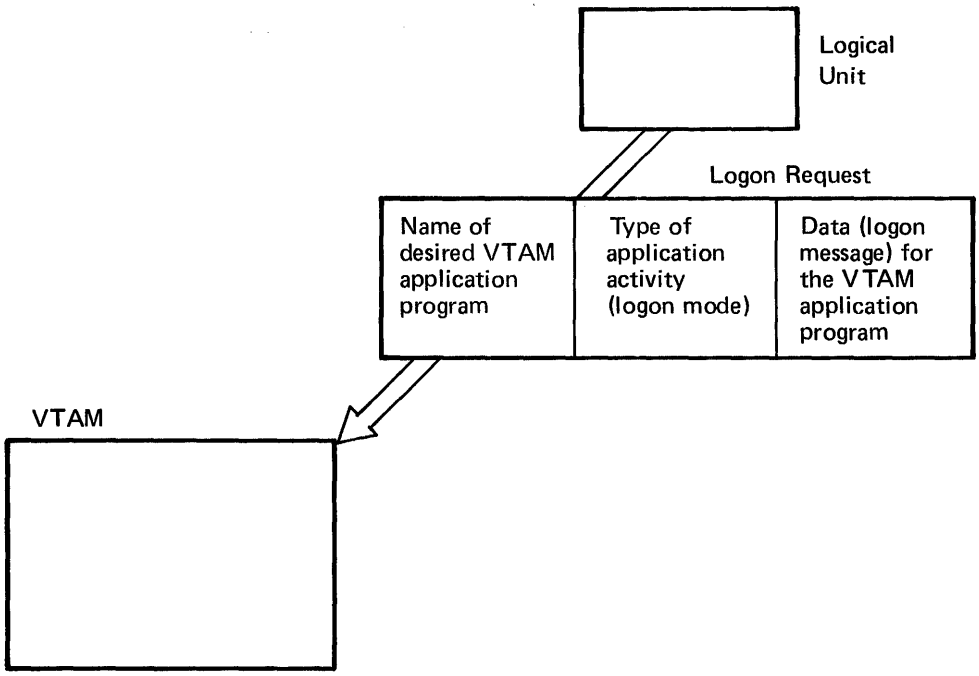


Figure 3-4. The Kind of Information Contained in a Logon Request From a Logical Unit

messages is defined by SNA and is described in *Systems Network Architecture Format and Protocol Reference Manual*, GA27-3112. With some SNA terminals, an installation may not need to know the exact format because VTAM and the terminal or the IBM-supplied terminal system program will format the message appropriately.)

The connection request message is sent in either field-formatted or character-coded form. In general, logical units associated with SNA terminals with programmable controllers can send field-formatted logon requests. Some terminals can also send character-coded logon requests. In general logical units associated with non-programmable SNA terminals send character-coded logon requests. See the programming publications for the particular SNA terminal product being installed to determine whether to plan for field-formatted or character-coded requests or both. Figure 3-5 provides an example of the types of requests for several IBM SNA terminal products.

An installation that uses only field-formatted logon requests may not need to understand character-coded logon requests. An installation that uses only character-coded requests need understand field-formatted requests only to the extent that a character-coded request is converted by VTAM into a field-formatted request.

A field-formatted request uses the SNA formatted system services (FSS) of VTAM. A character-coded request uses the SNA unformatted system services (USS) of VTAM. SNA *system services* are performed by the SNA *system services control point (SSCP)*, which is part of VTAM and provides such services as processing requests for connection and disconnection. VTAM converts a character-coded request into a field-formatted request before it is processed.

Defining Field-Formatted Connection Requests

A field-formatted connection request is already formatted when VTAM receives it. VTAM determines that it is a connection request for a particular VTAM application program. VTAM passes the request to the VTAM application program by scheduling the program's LOGON exit routine. If there is no LOGON exit routine, VTAM completes a

Type of SNA Terminal	Form of Logon/Logoff Request That Can Be Used	In the LU Statement, Specify SSCPFM=
3270	Character-coded (required)	USS3270
3600	Field-formatted	FSS
3767	Character-coded (required)	USSSCS
3790	Field-formatted	FSS

Figure 3-5. The Form of Logon/Logoff Requests Required or More Commonly Used By Several SNA Terminals

pending OPNDST macro instruction. No special action is required when the network is defined to VTAM to prepare for field-formatted requests. The logical unit specifies the name of the VTAM application program that is requested (the name of an APPL statement) in the field-formatted connection request.

In general, field-formatted connection requests are made by programmable SNA terminals. Depending on the product, the program may be supplied by IBM or the user. The program formats the request into a field format, based either on a logon request from an operator at an associated device or on some predefined program logic. In either case, the programmer of the SNA terminal program must know the name of the VTAM application program with which connection is requested. To specify that connection requests from a logical unit will be field formatted, each LU statement should specify SSCPFM=FSS or omit the operand when defining the network to VTAM.

Figure 3-6 shows a field-formatted connection request.

Defining Character-Coded Connection Requests

A character-coded connection request is designed to provide flexibility for SNA terminals that do not contain a program. The terminal operator directly keys in a request for connection or logon to a specific VTAM application program, indicating an application mode (such as interactive or batch) and specifying data (a logon message) to be passed to the VTAM application program. VTAM receives the character-coded connection request, converts it to a field-formatted connection request, and processes it as a field-formatted request. An installation has a great deal of flexibility in defining the format and content of what the terminal operator can enter as a connection request, either using an IBM-supplied definition table or supplying its own. To specify that connection requests will be character-coded, each LU statement should specify SSCPFM=USSSCS or, for an SNA 3270 terminal, SSCPFM=USS3270, when defining the network to VTAM.

Figure 3-7 shows a character-coded connection request.

Using the IBM-Supplied Character-Coded (USS) Definition Table

For SNA terminals whose associated logical units send character-coded connection requests, IBM defines a connection or logon request format and related definition table that VTAM uses to convert it to field-formatted requests. To use this IBM-supplied definition table, the installation specifies SSCPFM=USSSCS or SSCPFM=USS3270, as appropriate, in the LU statement during VTAM definition.

The IBM-supplied USS definition table (ISTINCDT) requires that the terminal operator enter a connection request or logon command in this form:

LOGON APPLID(application program name) LOGMODE(logon mode) DATA(logon message)

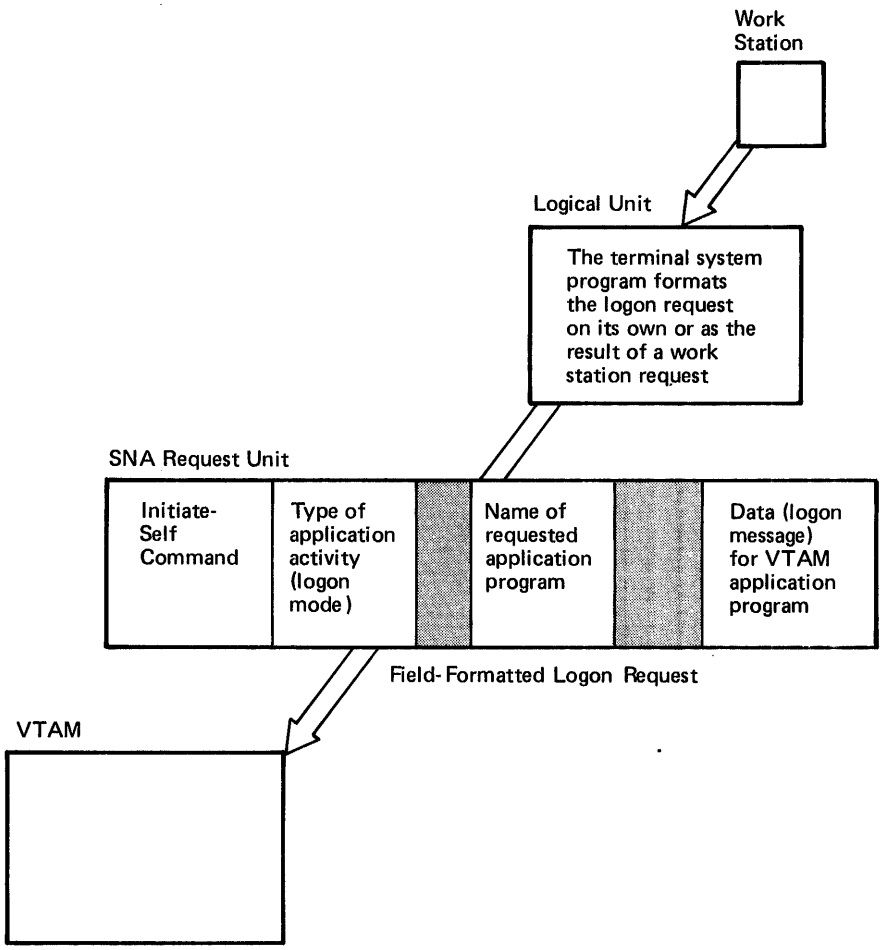


Figure 3-6. A Field-Formatted Logon Request

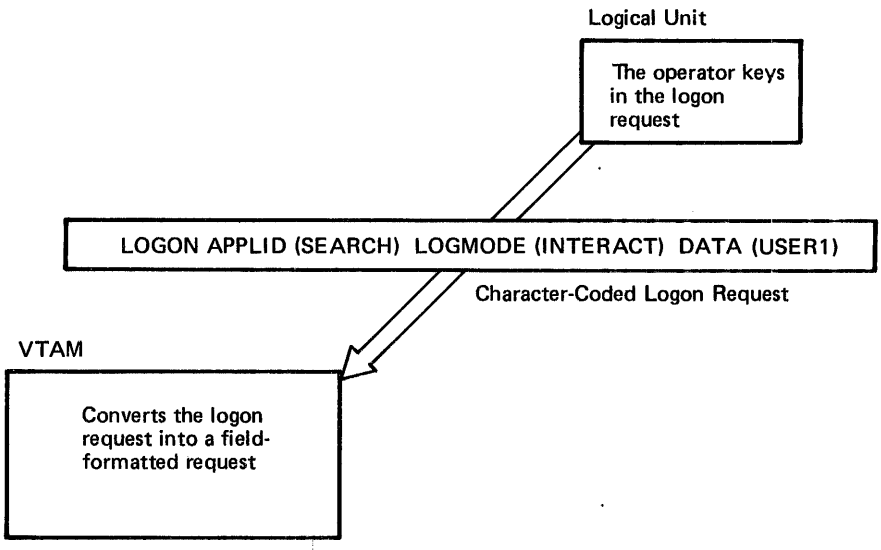


Figure 3-7. A Character-Coded Logon Request

The logon mode is the type of application work planned, such as batch or interactive; defining a logon mode table is described later in this chapter. If the IBM-supplied logon command is entered in lowercase, VTAM translates it to uppercase.

If a character-coded logon request is incorrectly entered, VTAM sends an appropriate message to the terminal operator.

Defining a Character-Coded (USS) Definition Table

An installation can define its own logon format and associated definition table instead of using the logon format and definition table supplied by VTAM. This allows an installation to select:

- Its own character translation table instead of the table supplied by IBM, which translates from lowercase to uppercase. The installation table could translate numeric characters to alphabetic characters, for example.
- A BAL format rather than a PL1 format (for example, the terminal operator could enter LOGON APPLID=application name,LOGMODE=logon mode,DATA=logon message).
- Replacement for the IBM-supplied verbs or parameters.
- Defaults for the application name, logon mode, or logon message.
- Alternate text for USS error messages.

As shown in Figure 3-8, to define its own logon format and definition table, the installation specifies the name of the definition table in the USSTAB operand of the LU statement when the network is defined to VTAM. The USS definition table is created by following this procedure:

1. Code a USSTAB macro instruction to indicate the beginning of the USS definition table. Use the TABLE operand to specify a character translation table if one is to be substituted for the default character translation table (which translates from lowercase to uppercase).
2. For each verb or command that a terminal operator can enter, code a USSCMD macro instruction that specifies the verb or command, the replacement USS verb (LOGON or LOGOFF), and the format of the user-entered command (PL1 or BAL).
3. For each USSCMD macro instruction, code a USSPARM macro instruction for each positional or keyword parameter on the user-entered command. USSPARM specifies the parameter, the USS keyword to identify the parameter value, and a default value to be used if the parameter is not entered.
4. For each USS message to be replaced, code a USSMSG macro instruction that specifies the number of the message and the replacement text.
5. Code a USSEND macro instruction to indicate the end of the definition table.
6. Assemble, link-edit, and load the resulting module into the VTAM load module library.

Figure 3-9 shows an expample of how an installation-defined USS definition table can translate a terminal operator's logon request.

Defining and Using an Interpret Table

For either field-formatted or character-coded system service requests, an installation can define and use an interpret table. An interpret table provides an additional translation of the application program name to which connection is requested. The name of the interpret table is coded on the LOGTAB operand of the LU statement when the network is defined to VTAM. The interpret table is designed primarily for use in connection procedures for local 3270, BSC, and start-stop terminals. It is described in Chapter 8, "Support for Local 3270, BSC, and Start-Stop Terminals."

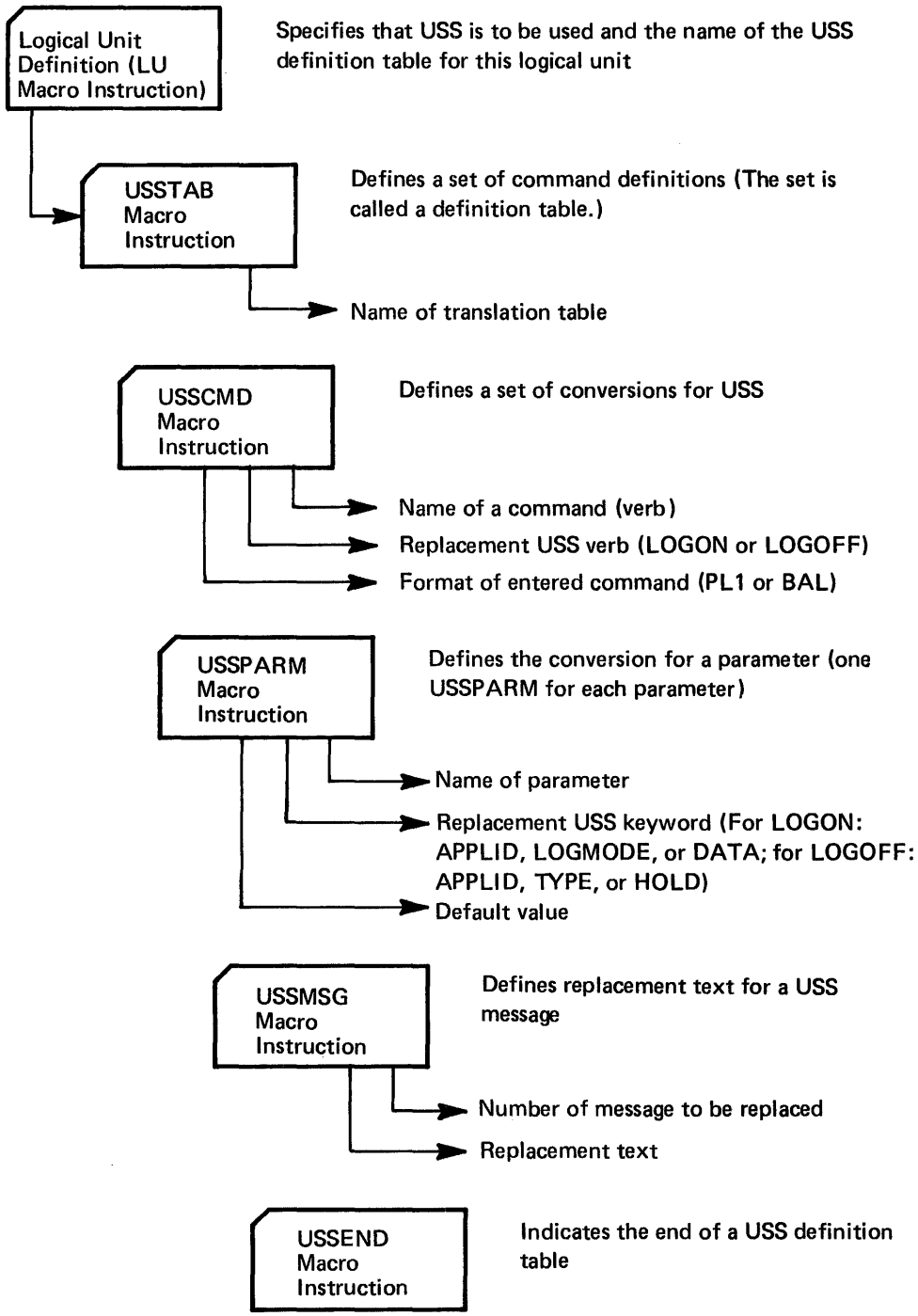


Figure 3-8. Defining a USS Definition Table

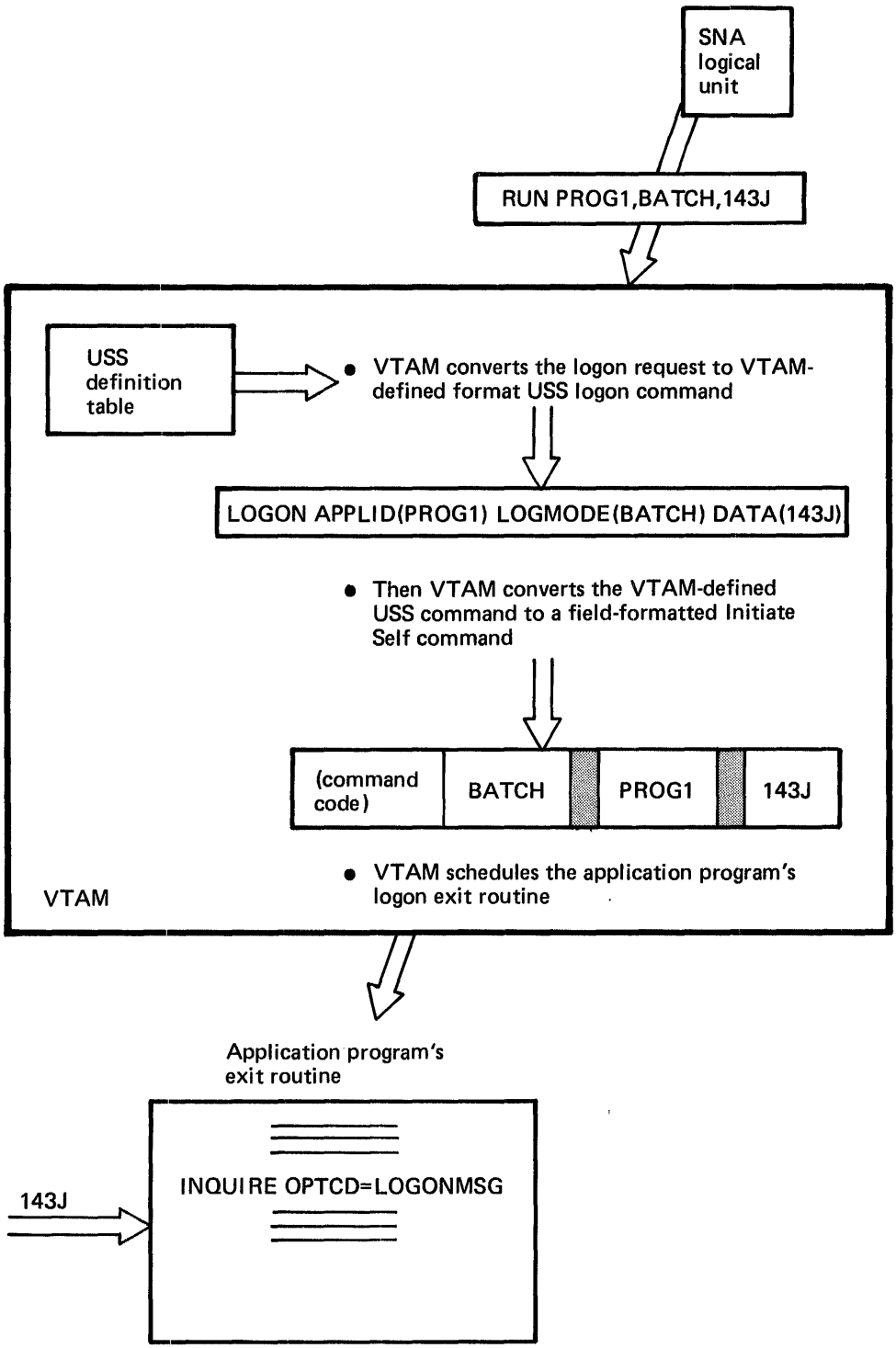


Figure 3-9. Example of VTAM Handling a Character-Coded (USS) Logon Request

Defining VTAM-Initiated Connection (Automatic Logon)

An installation can specify that a logon request is to be automatically generated on behalf of a terminal (a logical unit or a local 3270, BSC, or start-stop terminal) for a selected application program whenever that terminal is active and not already connected to another program. This specification is made by coding the name of the program in the terminal's definition statement. The application program name is specified in the statement's LOGAPPL operand; it is the same as the name specified for the application program in its APPL definition statement. (In the case of terminal-initiated requests for local 3270, BSC, and start-stop terminals, the terminals can be automatically logged on to the network solicitor, which handles these requests. NETSOL is the name of the IBM-supplied network solicitor and can be specified in the terminal's LOGAPPL operand. See Chapter 8 for details on the network solicitor.)

For logical units that may call in on switched lines, VTAM holds an automatic logon designation for an activated logical unit pending until a dial-in operation for the logical unit occurs. VTAM then schedules the designated LOGON exit routine or completes an OPNDST that specifies OPTCD=ACCEPT.

When defining a VTAM-initiated connection, it should be noted that the VTAM application program does not receive a logon message in connection with the request.

Defining VTAM Application Program-Initiated Connection

A VTAM application program can initiate a request for connection of a terminal to itself or to another application program. To request that a terminal be connected to itself, a VTAM application program can:

- Directly acquire the terminal by issuing an OPNDST macro instruction that specifies OPTCD=ACQUIRE
- Cause its LOGON exit routine to receive a request for connection from another part of the program by issuing a SIMLOGON macro instruction

A VTAM application program can request that a terminal currently connected to it be passed to another active VTAM application program. This is done by issuing a CLSDST macro instruction that specifies OPTCD=PASS with the name of the program to which control is to be passed specified in a designated area of the program issuing the CLSDST. Optionally, a logon message can also be passed. As a result of this CLSDST, VTAM schedules the receiving program's LOGON exit routine or, if there is no LOGON routine, completes an outstanding OPNDST for that logical unit or terminal. An installation must authorize a program to use this procedure by specifying AUTH=PASS on the program's APPL statement during VTAM definition.

Considerations for using these forms of application program-initiated connection are discussed below.

Acquiring Connection (OPNDST with OPTCD=ACQUIRE)

This form of connection might be useful when:

- The VTAM application program determines whether or not a shareable resource, such as a master display terminal or printer, is in use by another program, and acquires it if it is not.
- An application program requires an output-only terminal that cannot initiate a connection request itself.
- As an alternative to a VTAM-initiated (automatic logon) request, a terminal is to be routinely assigned to the program for the duration of its being online.
- It is not necessary for all connected terminals to be processed by the program's LOGON exit routine (if it is, a SIMLOGON should be used).

An installation must authorize acquisition of logical units and terminals by specifying AUTH=ACQ on the APPL statement when the network is defined to VTAM.

Simulating a Logon Requets (SIMLOGON)

This form of connection might be useful:

- When a program wants to acquire a terminal and wants to have all connections occur at the same place in the program (in the LOGON exit routine). This might be because the LOGON exit routine keeps track of how many or which terminals are connected to the program.
- When an installation wants all terminals to be connected in the same manner to an application program, using a correct logon message. (Logons automatically initiated by VTAM cannot specify a logon message.)
- If a terminal is not available when requested, and SIMLOGON is issued and queuing of the request is specified, the request is queued if the logical unit or terminal is unavailable (active but already connected to another program). When the terminal becomes available, the program's LOGON exit routine is scheduled.

An installation must authorize a simulated logon request for a logical unit or a terminal from the VTAM application program by specifying AUTH=ACQ on the APPL statement when the network is defined to VTAM.

Passing Connection to Another Program (CLSDST with OPTCD=PASS)

This form of connection might be useful when a set of VTAM application programs are to be used in sequence. For example, an installation maintenance program may produce data that is used by a related program and then printed on a terminal.

An installation must authorize passing connection to another program by specifying AUTH=PASS on the APPL statement when the network is defined to VTAM.

Defining Network Operator-Initiated Connection

The installation can plan for a network operator to initiate connection for a terminal (a logical unit or start-stop, BSC, or local 3270 terminal) to an application program. This could be done routinely or as the result of special occurrences, such as an authorized user's calling and requesting connection of terminals to a particular VTAM application program. The network operator can initiate a connection request with the VARY command, specifying the names of both the terminal and the application program and for SNA terminals, the logon mode.

Besides generating a logon request for a terminal, the VARY command modifies any automatic logon designation for the terminal. Once the command is issued, the terminal specified in that command is automatically logged on to the application program whenever it is available unless that program has specifically released it. The newly designated automatic-logon specification remains in effect until (1) the network operator issues another logon command for that terminal and specifies another application program, or (2) the major node containing that terminal is deactivated. When the major node is reactivated, the automatic logon conditions specified when the VTAM network was defined are in effect.

If the network operator initiates the connection request, a logon message cannot be specified.

Defining Logon Modes

Whenever a connection request is initiated for a logical unit (whether by VTAM, a VTAM application program, the network operator, or the logical unit itself), a *logon mode* is specified or defaulted for the logical unit. The logon mode is a set of communication rules that the VTAM application program and the logical unit agree to follow. (VTAM does not necessarily enforce these rules; the VTAM application program, however, may

not work properly if it fails to conform to the rules agreed upon with the logical unit.) These rules generally correspond to modes of application program operation, such as interactive or batch operation. For logical units of some IBM SNA terminal products, the logon mode is preestablished as, for example, always interactive, or always batch. For others, the logon mode can be varied, and the connection request initiator must select the appropriate logon mode.

The logon mode for a logical unit can be specified when the network is defined or when the connection request is initiated.

If no logon mode is specified when the connection request is initiated, the logon mode specified for the logical unit during network definition is used. If no logon mode was specified during network definition, an IBM-supplied logon mode is used that defines a set of session parameters for interactive operation.

The logon mode requested by the connection initiator other than the VTAM application program or defined during network definition can be modified by the logon mode that the VTAM application program specifies when it issues an OPNDST macro instruction.

The VTAM application program can issue an INQUIRE macro instruction to determine the logon mode requested for a logical unit with which it is establishing connection (unless it is itself the connection initiator).

Defining Logon Modes When Creating the VTAM System

An installation can create its own logon mode tables in addition to or instead of the IBM-supplied logon mode table. Each table contains one or more entries; each entry describes a logon mode. To create a logon mode table, an installation uses VTAM's MODETAB, MODEENT, and MODEEND macro instructions as shown in Figure 3-10. The MODETAB and MODEEND macro instructions specify a group of logon modes and each MODEENT specifies one logon mode. Each logon mode table is assembled and filed separately (as a member in OS/VS or book in DOS/VS) in the VTAM load module library. The name assigned to the load module is specified in the MODETAB operand in

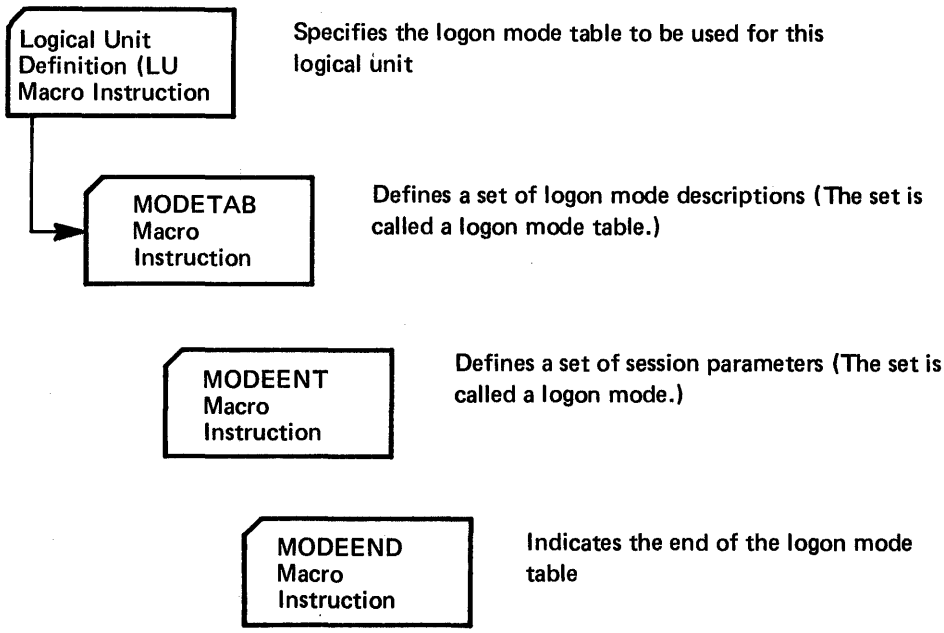


Figure 3-10. Defining a Logon Mode Table

the LU statement for each logical unit with which this table is to be associated. Each entry in that table has both a name, such as INTERACT, and a related set of session parameters associated with that name.

Note that a logon mode can also be specified when the logon request is initiated by VTAM, a VTAM application program, or the network operator.

Defining Disconnection Procedures

An installation must also plan the procedure by which a terminal is disconnected. A request to disconnect a terminal can come from:

- The terminal (in a logoff request)
- The VTAM application program
- The network operator

If a terminal is still active after it has been disconnected, it may be available for connection to another VTAM application program.

Disconnection Requested by the Terminal

For start-stop, BSC, and local 3270 terminals, the VTAM application program may search each data message from the terminal for a predesignated character or character-string indicating a request for logoff or disconnection. VTAM does not provide any special facility for logoff requests from start-stop, BSC, and local 3270 terminals. When the application program receives a logoff request from a local 3270, BSC, or start-stop terminal, it may issue a CLSDST macro instruction, causing VTAM to logically disconnect the terminal from the program.

For SNA terminals, VTAM provides a logoff facility similar to the logon facility described previously in this chapter in "Defining Connection Procedures." A logical unit can send either a field-formatted or character-coded logoff request. The field-formatted request is an SNA Terminate Self command; the character-coded request is a logoff request which VTAM converts into a formatted Terminate Self command. Consult the programming publications for each particular terminal product to determine the kind of logoff request required (field-formatted or character-coded or either).

If a character-coded logoff request is used, the installation can either use the IBM-supplied logoff format, or it can define its own logoff request format by defining a USS definition table, as previously described in "Defining a Character-Coded (USS) Definition Table." The IBM-supplied logoff format is:

LOGOFF APPLID(application name) TYPE(COND or UNCOND) HOLD(YES or NO)

The logoff parameters, whether they arrive in a Terminate Self command or in a logoff request format defined by IBM or the installation, specify:

- The VTAM application program from which the logical unit wants to be disconnected
- Whether disconnection is mandatory (UNCOND) or at the discretion of the application program (COND)
- Whether the physical unit with which this logical unit is associated is also to be disconnected if this is the last remaining logical unit that is connected

The VTAM application program from which disconnection is requested gets control as a result of the logoff request. Its action depends on the conditional or unconditional parameter.

Conditional or Unconditional Disconnection

If the logoff request specifies conditional termination, VTAM schedules the application program's LOSTERM exit routine. The program can then ignore the request or disconnect the logical unit (by issuing a CLSDST macro instruction). This kind of request allows the application program to exchange some final messages with the logical unit. If the logoff request specifies unconditional termination or omits the parameter, VTAM disconnects the logical unit from the application program, and then schedules the program's LOSTERM exit routine with an indication that the logical unit has been disconnected.

Holding the Physical Unit Connection

Although not apparent to the VTAM application program, VTAM establishes a connection between itself (the system services control point) and each physical unit before it establishes a logical connection between a VTAM application program and a logical unit subordinate to a physical unit. For physical units on leased lines, this connection is established between VTAM and a physical unit when the physical unit is activated. For physical units on switched lines or locally attached, the connection between VTAM and the physical unit is not established until a VTAM application program or a logical unit associated with a physical unit requests a connection between a VTAM application program and a logical unit. Once the physical unit on a switched line is connected to VTAM, VTAM maintains the connection as long as one of its logical units is connected unless the installation specifies otherwise.

An installation can specify:

- When defining the physical unit to VTAM whether or not the physical unit is to be disconnected as soon as the last of its logical units has become disconnected (DISCNT=YES) or whether this is to be decided by the physical unit location (DISCNT=NO). (DISCNT=YES is the default.)
- If DISCNT=NO is specified during VTAM definition, each logoff request can specify whether or not the physical unit connection is to be held if this is the last logical unit that is connected. This is done by specifying the Last or Not Last indicator in a field-formatted Terminate Self command or HOLD(YES or NO) in a character-coded logoff request. The default for a field-formatted command is to hold the connection; the default for a character-coded request is not to hold the connection if this the last remaining logical unit.
- If the logoff request for a logical unit specifies that the physical unit connection be held, the physical unit can be disconnected by a Ready-to-go-on-Hook (Request Disconnect) request from the physical unit (which might result from timing logic in a program associated with the terminal, or from the action of an operator at the terminal or terminal controller represented by the physical unit).

An installation might want to hold the connection to a physical unit when logical unit connections to application programs are expected to be frequent but of short duration to allow periods of time during which no logical unit is in session with the host. For physical units on switched lines, holding the physical unit connection allows control over the frequency of dial-in or dial-out operations for the physical unit.

Disconnection of the physical unit from VTAM is different for physical units on non-switched lines and physical units on switched lines or locally attached. When a non-switched line physical unit is disconnected from VTAM, it is also deactivated. When a switched line or local physical unit is disconnected, it remains active.

Using the Request Shutdown Protocol

A logical unit can also use an SNA Request Shutdown command to request disconnection. This command advises the VTAM application program that the logical unit is ready to complete its work. The application program can send any final messages to the logical unit and then issue a Shutdown command. When the application program in turn

receives a Shutdown Complete reply, it can disconnect the logical unit using the CLSDST macro instruction.

Disconnection Requested by the VTAM Application Program

A VTAM application program can decide that communication with a connected terminal has ended and that the terminal is to be disconnected. This might be the normal procedure for disconnection on termination of batch output to a terminal. The VTAM application program simply issues a CLSDST macro instruction, causing VTAM to logically disconnect the program from the terminal. OPTCD=RELEASE can be specified to allow the terminal to be available for connection to any other program. OPTCD=PASS can be specified to notify a specified application program that the terminal is available for connection. The installation must authorize a VTAM application program (when it defines the network to VTAM) to use OPTCD=PASS.

A VTAM application program can also issue a CLSDST macro instruction after receiving an SNA Shutdown Complete command from a logical unit as discussed previously in "Using the Request Shutdown Protocol."

Disconnection Requested by the Network Operator

The network operator can deactivate a terminal (or a higher level of major or minor node) *normally* or *immediately*. Normal deactivation does not result in a disconnection or logoff request on behalf of a terminal; it simply indicates to VTAM that the next time the terminal becomes disconnected from a VTAM application program, it is to become inactive. Immediate deactivation, however, acts as a network operator logoff request on behalf of a terminal. The VTAM application program's LOSTERM exit routine is scheduled and the program should then issue a CLSDST macro instruction to disconnect the terminal, allowing the terminal to become inactive. This procedure might occur most frequently when the network operator has become aware of a problem with the terminal or network of which the VTAM application program was not aware.

Effect of Disconnection on Terminals Defined with an Automatic Logon

Whatever procedure an application program follows for logoffs, when a terminal with an automatic logon specification is disconnected, VTAM usually attempts to queue the terminal to the application program named in that specification. Some conditions can arise in which a terminal is not immediately queued in accordance with the automatic logon specifications. For a description of these conditions, see the discussion on disconnection in Chapter 5, "Writing a VTAM Application Program."

Coding and Including Installation Exit Routines

VTAM provides two types of exit routines: application program exit routines and installation exit routines. The application program exit routines are coded and identified in each application program that uses VTAM. These exit routines are executed as part of the application program and are under the control of the application program. The installation exit routines for VTAM are coded and included in VTAM as part of VTAM definition. These routines are executed as part of VTAM and are not under the control of application programs. Information on the application program exit routines is provided in "Application Program Facilities," in Chapter 5. The remainder of this section provides information on coding and using the installation exit routines, and on including them in VTAM.

VTAM provides three types of installation exits:

- An authorization exit, to validate connection, disconnection, and logon requests.
- An accounting exit, to collect accounting information.
- Logon-interpret exits, to determine the appropriate application program to receive a logon. These are intended for use with local 3270, BSC, and start-stop terminals and are described in Chapter 8.

VTAM provides for one authorization and one accounting exit routine. If the installation does not provide its own exit routines, IBM-supplied modules are used. One logon-interpret routine can be coded for each LOGCHAR macro instruction, although the same routine can also be used for more than one LOGCHAR instruction. Logon-interpret routines are not provided with VTAM; if these routines are needed, they must be supplied by the installation. Each type of exit routine is described below.

Authorization Exit Routine: VTAM provides an exit that permits the installation to authorize connections between application programs and terminals. (Authorization in the exit routine is in addition to that performed by VTAM.) This exit is scheduled whenever a terminal is to be queued for logon to, connected to, or disconnected from an application program. (See "Application Program Concepts and Facilities," in Chapter 5, for the distinction between connection and queuing for logon.) Thus, the routine at this exit is executed whenever a connection is to be made or broken as a result of an OPNDST or CLSDST macro instruction in the application program, or whenever a terminal is to be queued as the result of a logon.

Upon entry to this exit routine, the following information is available:

- The type of request that has been made; that is, whether the request is for a connection, a disconnection, or a logon. Also, if it is a connection request, VTAM specifies whether it is an accept or an acquire. If it is a logon request, VTAM specifies the type of logon.
- The names of the terminal and the application program to be connected, disconnected, or queued. Also, if the operation is a logon resulting from a CLSDST macro instruction with the PASS option, VTAM also provides the name of the application program issuing the pass request.

Using this input, an installation-coded authorization routine can determine whether each connection, disconnection, or logon request should be processed by VTAM. For example, each request might be compared against a predefined, installation-specified table of valid or invalid requests. The results of this examination are then returned to VTAM. If the request is determined to be valid, it is completed by VTAM. If it is invalid, it is rejected by VTAM. Output from this routine must include whether the request is valid or invalid.

In DOS/VS, the authorization routine is included in VTAM by cataloging it into the core image library. In OS/VS1, it is included by link-editing it as a single load module into the VTAM load module library. In OS/VS2, it is included by link-editing it as a single load module into the SYS1.LPALIB data set. This routine replaces an IBM-supplied module. If the installation does not replace the IBM-supplied authorization exit routine, all connection, disconnection, and logon requests handled by this module are treated as valid.

In planning an authorization routine, a number of factors must be considered:

- The exit routine is executed in the supervisor state under VTAM's protection key. Therefore, errors within the routine may cause damage to VTAM's control blocks and modules. Also, security violations can occur if such a routine is designed or coded by unauthorized persons, since they then have access to much of the VTAM partition or private address space.
- The exit routine is executed under the task for which the request is being authorized (for example, the application program that issued the connection request). If the exit routine is abnormally terminated, this task may be terminated also.
- The exit routine is executed inline with VTAM processing. Therefore, performance may be degraded if the routine requires lengthy processing time. While this routine is being executed, no new connection, disconnection, or logon requests are processed by

VTAM, and requests involving VTAM's VARY command are not processed. Therefore, system waits (such as for disk I/O) should be avoided.

- The exit routine is notified of pass requests. (The pass option is an option of the OPNDST macro instruction; this option is used by VTAM's network solicitor and can be used by application programs.) The network solicitor uses the pass option to pass valid terminal-initiated logons to active application programs. If the installation is to use the network solicitor to monitor terminals for logons, the installation-coded authorization routine should be designed to process the validation of pass requests involving the network solicitor.
- The exit routine is notified if VTAM's network solicitor or the Terminal Online Test Executive Program (TOLTEP) attempt to connect a terminal. An installation-coded authorization routine should be designed to process these requests. (See "Serviceability Aids," in Chapter 6, for a description of TOLTEP.)

Accounting Exit Routine: VTAM provides an exit that permits an installation to maintain accounting information about connections. This exit is scheduled whenever a terminal and an application program are connected or disconnected. Thus, the routine is executed each time the application program OPNDST or CLSDST macro instruction is issued.

Upon entry to this routine, the following information is available:

- The name of the application program
- The name of the terminal
- The type of request; that is, whether the operation is a connection or a disconnection

Using this input, an installation-coded accounting routine could note and record the time a connection is initiated. Upon re-entry (when that connection is being broken), the routine can note the time of the disconnection. The difference between these two times is the connection time for the terminal and that application program. Note that connection time is an approximate value affected by such factors as the running time of the application program, the paging rate, and the operating system setting the application non-dispatchable.

In DOS/VS the accounting routine is included in VTAM by cataloging it into the core image library. In OS/VS1, link-editing it as a single load module into the VTAM load module library. In OS/VS2, it is included by link-editing it into the SYS1.LPALIB data set. This routine replaces an IBM-supplied module. If the installation does not replace the IBM-supplied accounting exit routine, no accounting statistics are gathered.

In planning an accounting routine, a number of factors must be considered:

- The exit routine is executed in the supervisor state under VTAM's protection key. Therefore, errors within the routine may cause damage to VTAM's control blocks and modules. Also, security violations could occur if such a routine were designed or coded by unauthorized persons, since they would have access to much of the VTAM partition or private address space.
- The exit routine is executed under the task about which the accounting information is being collected (for example, the application program that issued the connection request). If the exit routine is abnormally terminated, this task may be terminated also.
- The exit routine is executed inline with VTAM processing. Therefore, performance may be degraded if the routine requires lengthy processing time. While this routine is being executed, no new connection, disconnection, or logon requests are processed by VTAM, and requests involving VTAM's VARY command are not processed. Therefore, system waits (such as for disk I/O) should be avoided.

- The exit routine is notified of connection and disconnection requests involving terminals and IBM-supplied facilities. These facilities are TOLTEP, the network solicitor, the port solicitor (this facility supports start-stop and BSC switched networks), and the interface to the Telecommunications Access Method (TCAM). The installation-coded accounting routine should be designed to process requests involving these facilities.

Logon-Interpret Routines: Details on the logon-interpret routines are provided under "Specifying Interpret Tables," in Chapter 8, "Support for Local 3270, Start-Stop, and BSC Terminals." They are mentioned here with installation exits to indicate that they can be thought of as installation-level routines as opposed to application-program-level routines. The logon-interpret routines would probably be planned and coded as part of VTAM definition. Also, since these routines can be used to prohibit logon requests, they can be used in conjunction with the authorization exit routine to control connections.

Defining VTAM Start Options

When VTAM is started, options can be used to define the initial VTAM network and to select optional VTAM facilities. These start options can be specified by the operator as part of the START command (in OS/VS only) or as a response to the prompting of VTAM. The options can also be predefined and filed on the VTAM definition library. (See "Starting VTAM," in Chapter 4, for information on specifying options via the system operator's console.)

The VTAM start options tailor VTAM to the installation's needs each time the telecommunication system is started. Predefining start options relieves the network operator of this activity. In addition, more than one version of the start options can be predefined, each version specifying a different VTAM configuration. With different sets of predefined options, the installation can initialize a particular VTAM system merely by selecting the appropriate set of options.

The predefined start options are stored under the member name ATCSTR*xx* (where *xx* and a data set must be filed under this name even if it does not contain any options). Other ATCSTR*xx* data sets can be created, but the specific data set required is specified by the installation when VTAM is started.

The following start information can be supplied in the ATCSTR*xx* data set:

- Which major and minor nodes are to be traced by the VTAM trace facility
- Which major nodes are to be activated during start processing (VTAM initialization)
- The size of VTAM storage pools and in OS/VS, which pageable storage pools should be fixed
- The maximum number of major nodes active at any one time
- Whether certain network operator messages are to be suppressed
- Whether prompting messages should be sent to the network operator to obtain additional start information
- Whether the VTAM logon monitor facility for start-stop and BSC terminals (referred to as the network solicitor) is to be started when VTAM is started

If the network solicitor is activated, it begins monitoring the active terminals assigned to it for logons. See "The Network Solicitor," in Chapter 8, for a description of the network solicitor.

The trace facility is activated for specified terminals, and the trace continues for as long as the terminals are active or until the network operator stops the trace. See "Starting and Stopping VTAM Facilities," in Chapter 4, for information on stopping the VTAM traces. See "Serviceability Aids," in Chapter 6, for a description of VTAM's trace facility.

The names of major nodes to be activated when VTAM is started are stored as a member of the VTAM definition library. These names must be stored under the member name ATCCON*xx* (where *xx* is a two-character identification created by the installation). ATCCON00 is a default member name; other ATCCON*xx* member names must be specified by the installation during start processing. All ATCCON*xx* members must be created and filed by the installation. If an ATCCON*xx* member is not created and filed, the default member name ATCCON00 will be filed and a warning message will be provided when VTAM is started. See "Major and Minor Node Structure," earlier in this chapter, for a definition of a major node.

To activate major nodes during VTAM start processing, an ATCCON*xx* member is specified as a start option. This member then contains a list of the major nodes to be activated. This option enables the installation to specify the initial telecommunication configuration.

VTAM uses storage pools to allocate space for control blocks, buffers, and channel programs. Pools are established in both fixed and pageable storage. See "Defining VTAM Buffering," later in this chapter, for more information on specifying storage pools and on VTAM's use of these pools.

If the network operator is to be prompted, VTAM transmits messages to the system operator's console requesting that start options be entered from the console. The network operator is prompted only (1) if prompting is requested in the ATCSTR00 data set and (for OS/VS) no start parameters are entered with the START command or (2) if an error is encountered by VTAM during VTAM initialization. See "Starting VTAM," in Chapter 4, for information on the role of the network operator in starting VTAM.

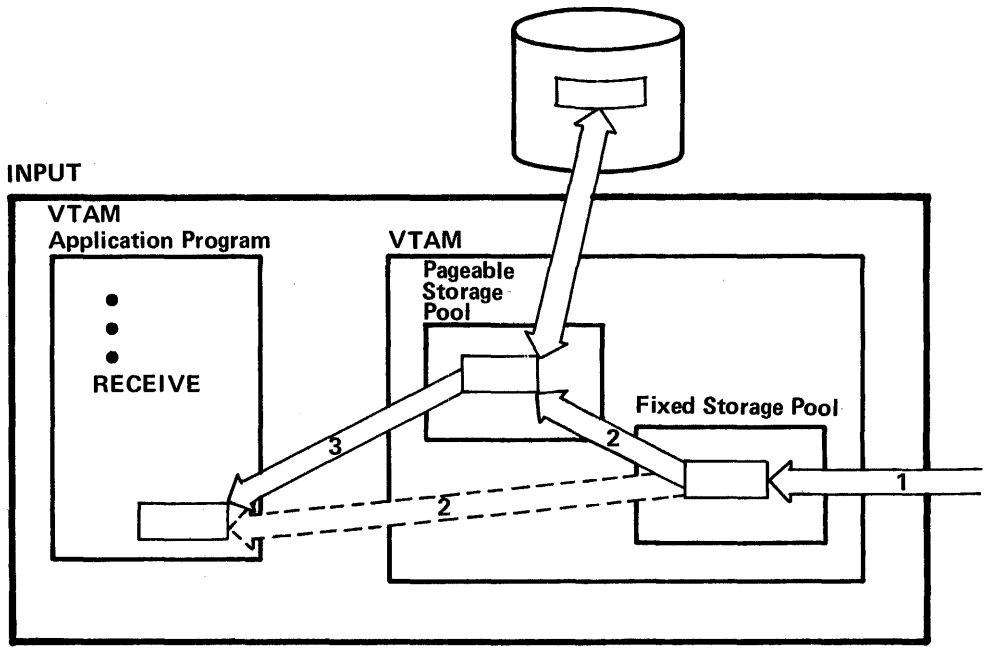
An installation can prepare and file multiple start-parameter members and multiple node-name members. Various combinations of these parameters can thus be used when VTAM is started to initialize VTAM systems with different characteristics.

Defining VTAM Buffering

VTAM is not a queued access method, such as TCAM; VTAM does not save a message until the message is requested by a terminal or an application program. However, VTAM can hold data temporarily until a VTAM application program requests it or until it can be sent to a terminal. Figure 3-11 shows how VTAM buffers handle input and output data.

VTAM has both fixed and pageable storage pools for buffering data. In addition, storage pools are required for the VTAM control blocks that keep track of incoming data that has not yet been requested by a VTAM application program and to keep track of individual VTAM application program input and output requests. Each storage pool serves all active application programs. Because installation needs may vary, VTAM allows an installation to specify the size of each of these storage pools. In OS/VS, the installation can fix one or more of these pools in main storage. The names of each of the storage pools are different in each operating system and may be found in the appropriate VTAM System Programmer's Guide.

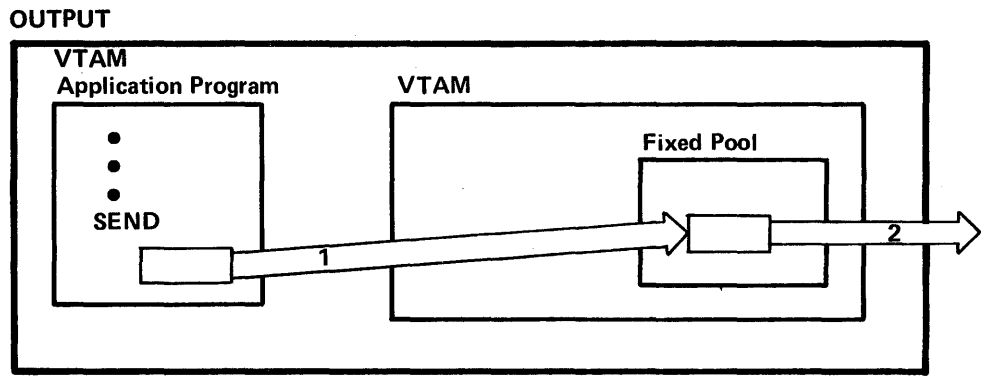
When an installation starts VTAM, it specifies the number of buffers (elements) in each storage pool, the size of each buffer, and for the data storage pool, a threshold number of buffers. The threshold number is the point beyond which, in OS/VS, requests for buffers for input or output are queued until buffers in use become free. Where no value is



1 Data arrives from a logical unit independent of any VTAM application program request for data. It is placed in a buffer in the fixed storage pool of data buffers.

2 If there is an outstanding request for data from the logical unit, the data is moved from the buffer in the fixed storage pool to the specified VTAM application program data area (shown with dotted lines).
If there is no request yet for the data, it is moved from the fixed to the pageable data buffer pool. Later, it can be paged out of main storage.

3 When an input request is issued, the data is paged into main storage, if necessary and moved to the specified VTAM application program data area.



1 The VTAM application program requests data transmission. When VTAM has sufficient buffers in the fixed storage pool, it moves the data to the fixed pool.

2 A short time later, the data is sent to the NCP or local terminal, freeing the buffers.

Figure 3-11. How VTAM Buffers Input and Output Data

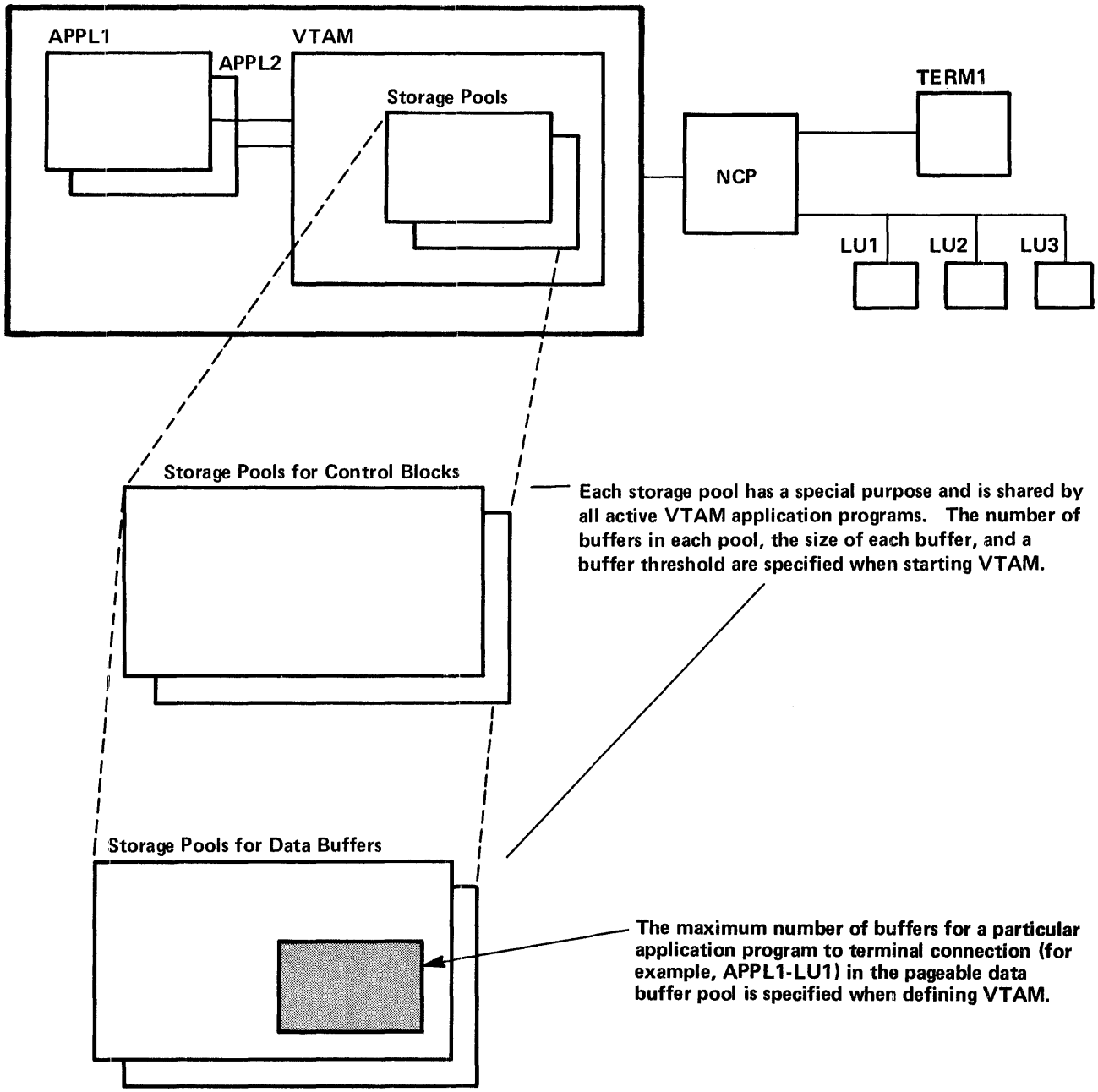


Figure 3-12. How an Installation Controls Control Block and Data Buffer Storage Pools in VTAM

specified, VTAM supplies a default value. In DOS/VS, VTAM supplies the threshold value; any number specified by the installation is ignored. If the threshold is consistently exceeded, program performance and terminal response time may be affected. An installation may have to "tune" the buffer number, size, and threshold of its storage pools to achieve an efficient combination of storage utilization and performance.

In addition to specifying the size and threshold of storage pools when starting VTAM, the installation can control the amount of buffer storage for data that can be used for each individual connection between an application program and a terminal. Controlling buffer storage for connection between application programs and terminals prevents an individual connection from monopolizing data buffers and slowing down communication between other connections. The number of data buffers for each connection is specified during VTAM definition. VTAM determines the data buffer threshold for each connection by multiplying a number specified for the application program (in the APPL statement) by a number specified for the terminal (in a LOCAL, LU, TERMINAL, COMP, or VTERM statement). For incoming data from terminals, if the threshold is exceeded, VTAM clears any data that may have arrived from the terminal, issues a Clear command if the terminal is a logical unit, and schedules the VTAM application program's LOSTERM exit routine. The program is resonsible for resynchronizing the connection. This situation will not occur, however, if the installation ensures that the VTAM application program requests input at a rate that is not below the rate at which data may be coming in to VTAM.

Figure 3-12 shows how an installation controls the buffering of data.

Although VTAM buffers both data coming in from a terminal to a VTAM application program and data going from a VTAM application program to a terminal, input data may have to be buffered for a longer time than output data. For input data, VTAM must buffer the data until it is requested by a VTAM application program. For output data, VTAM decides when it wants to begin handling the data; after receiving an output request, VTAM need not move the data from the VTAM application program until it has ensured that it has buffers for it. For this reason, more planning is required for the buffering of input data. In general, a VTAM application program should be written so that at least one request for input is always outstanding, or at least so that very little time elapses during which no request for input is outstanding. This may reduce the possibility of exceeding the buffer size for data and control block storage pools.

In OS/VS, an installation has the option to fix VTAM storage pools. An installation that requires high performance may want to fix those pools that may not always be resident because of infrequency of use.

Recommendations and guidelines for determining the sizes and thresholds for storage pools of buffers are provided in the *VTAM System Programmer's Guide* and storage estimate publication for the appropriate operating system.

CHAPTER 4. CONTROLLING A VTAM SYSTEM

This chapter describes in general how an installation controls VTAM using predefined specifications and network operator commands. The appropriate *VTAM System Programmer's Guide* describes predefined specifications in detail and the appropriate *VTAM Network Operating Procedures* describes network operator commands in detail.

Levels of Control

An installation can control VTAM by the specifications it makes during VTAM definition, and by using network operator commands.

During VTAM definition and NCP generation, an installation defines and tailors a VTAM system. Defining and tailoring the system are done before the actual use of that system. Extensive planning is required because the definition of the system and the generated NCPs are not usually subject to frequent changes.

The network operator uses VTAM commands to control a VTAM telecommunication system between the time VTAM is started and the time it is halted.

An installation uses VTAM definition facilities to define the operating limitations of the telecommunication system. Network operator facilities enable the network operator to modify, within these limitations, the system's activity in response to varying requirements. For example, using VTAM definition facilities, an installation can define telecommunication configurations and specify valid connections; the network operator, working within these definitions, can activate and deactivate nodes and connections to control the use of the system.

VTAM Commands

VTAM's *network operator commands* enable the network operator to monitor and control the telecommunication system. VTAM commands are a subset of the operating system commands; as such, they must be entered from a system console. Incorrect commands are rejected by VTAM's command facility, and a message specifying the error is written to the operator.

Using commands, the network operator can:

- Start VTAM
- Stop VTAM
- Monitor the status of the telecommunications system
- Activate and deactivate nodes
- Initiate requests for connections between terminals and application programs
- Start and stop selected VTAM facilities
- Change line-scheduling specifications for start-stop and BSC lines

Starting VTAM

VTAM is started by initializing VTAM and the telecommunication network prior to using the VTAM system. Starting VTAM can include the activation of some or all of the nodes. It can also include the activation of selected VTAM facilities.

At the time VTAM is started, the installation tailors the telecommunication system either by entering VTAM start options through the network operator's console or by naming a

data set that contains predefined start options. The data set is one of the ATCSTR*xx* members (members for OS/VS, books for DOS/VS) of the VTAM definition library. See "Defining VTAM Start Options," in Chapter 3, for information on predefining VTAM start options in ATCSTR*xx* data sets.

The VTAM start options that can be entered through the network operator's console are almost the same as those that can be predefined and filed as a ATCSTR*xx* member. The options that can be entered through the console include:

- A host identification number that a physical unit can use for host identification
- Whether VTAM's network solicitor is to be activated
- Whether VTAM's trace facility is to be activated, and if activated, for which nodes
- Which major nodes are to be activated
- Sizes of VTAM storage pools and in OS/VS, which pageable storage pools should be fixed in main storage
- The maximum number of major nodes active at one time
- Whether VTAM storage management trace facility is to be activated
- Whether certain messages are to be suppressed
- Whether other start options should be taken from an ATCSTR*xx* member

Except for the last item (a pointer to a list of predefined start options), start options that can be specified by the network operator can also be predefined in an ATCSTR*xx* member.

Starting VTAM differs slightly between DOS/VS and OS/VS. Each are discussed below.

Starting VTAM in DOS/VS

VTAM is started in DOS/VS by first starting the VTAM partition in which VTAM is to be executed and then invoking the VTAM procedure. No VTAM start options can be entered from the console in either of these two steps.

When the network operator starts VTAM, any start options in the ATCSTR00 book are processed first. If prompting was specified in the ATCSTR00 book, VTAM prompts the network operator for start options. If any errors are encountered in the start processing, VTAM prompts the network operator for the last options to be processed before VTAM completes initialization.

Starting VTAM in OS/VS

The START command starts VTAM in OS/VS. This command can be entered alone or with any of the VTAM start options listed above.

When the network operator starts VTAM, any start options in the ATCSTR00 member are processed first. If the operator included any start options with the START command, these options are processed next. If no start options were specified in the START command and if prompting was specified in the ATCSTR00 member, VTAM prompts the network operator for start options. If any errors are encountered in the start processing, VTAM prompts the network operator for final options. These options are the last to be processed before VTAM completes initialization.

Halting VTAM

The VTAM HALT command can be used for an orderly closedown or a quick closedown. Orderly closedown is designed to halt VTAM under normal, planned conditions. Quick closedown is designed for emergency situations in which halting telecommunications takes precedence over loss of terminal connections and loss of data.

Orderly Closedown

If the network operator specifies an orderly closedown, VTAM does not prohibit connections between terminals and application programs, although new connections of application programs to VTAM are prohibited. VTAM notifies application programs of the pending closedown by scheduling each program's TPEND exit routine. (The TPEND exit routine is an application program exit routine that halts the application's teleprocessing activities when VTAM is terminating. See "Designing TPEND for the HALT Command," later in this chapter, for information on the TPEND exit routine.)

Except for prohibiting the opening of ACBs, VTAM allows normal operation (for example, reading and writing data) until all application programs have closed their access method control blocks (ACBs) and disconnected themselves from VTAM. Then VTAM deactivates all nodes and closes down the telecommunication system.

For VTAM's orderly closedown, the installation should ensure that each application program has a TPEND exit routine and that each TPEND exit routine is designed to halt teleprocessing activity in an orderly manner and then disconnect from VTAM. If an application program connected to VTAM does not have a TPEND exit routine, the application is not notified of the pending closedown. This delays the halt either until the application closes its ACB or the network operator cancels the application.

Quick Closedown

If the network operator specifies a quick closedown, VTAM prohibits any further communication between terminals and application programs. New connections of application programs to VTAM are also prevented. Write requests already being transmitted are allowed to complete, but pending input or output requests are canceled. No additional input or output requests are accepted, although data already read into VTAM buffers can be read by application programs. VTAM also notifies application programs of the pending closedown by scheduling each program's TPEND exit routine. Having scheduled each exit routine, VTAM waits until all application programs have closed their ACBs before it closes down the telecommunication system.

So that the quick closedown functions properly, the installation should ensure that each application program using VTAM has a TPEND exit routine. Since a quick closedown is probably indicative of an emergency situation, the exit routine should do no more than initiate a closedown procedure.

Designing TPEND for the HALT Command

Each application program's TPEND exit routine is scheduled whenever VTAM is about to terminate. If VTAM is terminating as the result of a HALT command, input to the exit routine indicates whether the closedown is orderly or quick.

For an orderly closedown, an application program can be designed to complete current processing, to notify connected terminals that the telecommunication system is closing down, and to disconnect itself from VTAM. For a quick closedown, the application program should be designed to do no more than close its ACB. Note that the ACB cannot be closed in the TPEND routine.

If all application programs do not close their ACBs within 45 seconds, VTAM notifies the network operator. The notification is a message indicating the names of the open ACBs. The network operator can then either permit the application programs to continue processing and wait until the ACBs are closed or can use the facilities of the operating system to cancel the job containing the application programs.

Because a CLOSE macro instruction cannot be issued in an exit routine, this routine should set an indicator that alerts the main portion of the application program to the pending closedown. In the case of a quick closedown, for example, the closedown procedure could post an event control block (ECB) and return to VTAM. This ECB

should have been previously created by the main portion of the application program, and the main portion of the program should have been waiting for it to be posted. When the ECB is posted, the application program regains control and can then issue a CLOSE macro instruction for its VTAM ACB. The CLOSE request automatically disconnects any terminals connected to the application program. (See Chapter 5 for information on the TPEND exit routine.)

Monitoring VTAM Status

The network operator monitors the staus of a VTAM system by requesting and studying status information for nodes in the system. VTAM's DISPLAY command enables the network operator to request status information and to verify changes resulting from previous operator requests. This command enables the operator to request information about the following types of nodes:

- Application programs (minor nodes)
- Terminals (logical units and local 3270, BSC, and start-stop terminals)
- Telecommunication lines
- BSC and start-stop switched-line ports. (Status displays for ports are the same as for telecomunication lines.)
- Physical units
- Network control programs (NCPs)

To display the status of a node, the network operator specifies the symbolic name of the node in the DISPLAY command. A minor node specified in a VTAM DISPLAY command must be part of an active major node. An NCP major node specified in a DISPLAY command must itself be active.

The information displayed by VTAM for each type of node includes:

- For an application program: Whether the application program is currently connected to VTAM, the names of terminals connected to the application program, and the names of terminals queued for logon to this application program. Because an open ACB is an application program to VTAM, a program can be executed in the system but not recognized by VTAM as an application if it does not have an open ACB for VTAM. Displays can still be requested of application program minor nodes for which there is currently no open ACB; such a display indicates that the application program is inactive (not connected to VTAM).
- For terminals: Whether the terminal is active or inactive, the name of the application program (if any) to which the terminal is allocated (that is, connected or queued for logon), the name of the application program (if any) for which an automatic logon is specified for the terminal, whether or not a logical unit is in a state of being connected or disconnected, the name of the switched major node if a logical unit is part of a switched major node, the name of the local major node if a logical unit is part of a local major node and the channel unit address of the associated physical unit, a list of traces in effect for the terminal, the current specification of the device transmission limit (for polled start-stop and BSC terminals only), the device type, a count of the input/output activity and temporary errors for the terminal, and the names of the group and the line to which the terminal is assigned (for remotely attached terminals) or the channel and unit address (for a locally attached 3270).
- For telecommunication lines: Whether the line is active or inactive, the name of the group to which the line is assigned, the names of all nodes assigned to the line and an indication of those that are active, the current specifications for polling delay, negative polling limit, and NCP session limit (for polled start-stop and BSC lines only), and whether the line is switched or nonswitched, whether a line trace is active for the line. In addition, if the line is SDLC, VTAM displays the names of the SDLC cluster

controllers that are assigned to the line and indicates those that are active, or displays the names of the remote communications controllers that are assigned to the line and indicates those that are active.

- For a physical unit: Whether the unit is active or inactive, whether a trace is active for the unit, the names of the VTAM terminals assigned to the control unit and an indication of those that are active, the names of the line and of the group to which the unit is assigned, and path information for physical units on switched lines.
- For NCPs: Whether the NCP is active or inactive, the channel and unit address and the type of the communications controller containing the NCP (if the NCP is for a locally attached communications controller), a list of traces in effect for the NCP the load-module name of the NCP, a count of the input/output activity and of temporary errors for the communications controller in which the NCP currently resides (if the NCP is for a locally attached communications controller), and an indication of whether the NCP is in a local or remote communications controller.

Activating and Deactivating Nodes

Before a node can be used in a VTAM system, the node must be active. The installation can control the activation and deactivation of many of the nodes in VTAM, including both major and minor nodes. All major nodes can be explicitly activated and deactivated by the network operator, including:

- NCPs for local and remote communications controllers
- Sets of physical units and logical units on switched lines
- Sets of locally attached physical units and logical units
- Sets of local 3270s
- Sets of application programs

When VTAM is started, major nodes can be activated by using VTAM start options; all active major nodes are deactivated when VTAM is halted. Major nodes can also be activated and deactivated with VTAM's VARY command while VTAM is being executed.

To activate or deactivate a major node after VTAM is started, the network operator enters a VARY command that contains the name of the node and indicates whether the request is for activation or deactivation. For an activation request, the node name is the name of the member of the VTAM definition library that contains the definiton statements for the node. For a deactivation request, the name is the name that was given in an activation request.

See "The Major and Minor Node Structure" in Chapter 3, for a definition of major nodes. See "Starting VTAM" and "Stopping VTAM," earlier in this chapter, for information on activating and deactivating nodes by other than the use of the VARY command.

Once a major node has been activated, some minor nodes within it can be activated and deactivated by the network operator while VTAM is running. The minor nodes that can be dynamically activated or deactivated at the request of the network operator are:

- SDLC, start-stop, and BSC lines
- Ports
- Physical units
- Logical units and local 3270, BSC, and start-stop terminals
- BSC and start-stop terminal components

Except for telecommunication lines, the installation can also specifically request the activation of any of these minor nodes when VTAM is started. A minor node's definition statement specifies that the node is to be activated automatically when its major node is activated; VTAM start options activate that major node.

SDLC, start-stop, and BSC lines are automatically activated by VTAM when an NCP is initially loaded. All lines can be activated and deactivated by network operator commands.

As noted earlier in this chapter, halting VTAM (by using VTAM's HALT command) automatically deactivates all active nodes. The orderly mode of VTAM's HALT command deactivates the nodes only when all application programs have closed their ACBs. The quick mode of the command begins to deactivate all nodes after the last application program has been notified of the pending halt, although VTAM termination is not completed until all ACBs have been closed.

To activate or deactivate a minor node while VTAM is being executed, the network operator must enter a VARY command containing the name of the node and indicating whether the request is for activation or deactivation. *Note:* A major node containing the minor node definition must be active before that minor node can be activated or deactivated.

The VTAM VARY command can be used for normal deactivation and immediate deactivation. When normal deactivation is specified, the node is not actually deactivated by VTAM until the associated application program and terminal connections have been terminated by either the application program or by the terminal operator. Queued requests for connections involving the nodes in the deactivation are dequeued; associated application programs are notified that the terminal is now inactive. No new requests for connection with these nodes are accepted. An application program connected to a terminal to be deactivated is not notified of pending normal deactivations.

Two common uses for normal deactivation are:

- Allowing many terminals to use a network by periodically and temporarily making some terminals inactive
- Closing down a portion of the total network of terminals, perhaps as a routine end-of-the-day operation

For a large network of terminals, normal deactivation can be used to deactivate some terminals as they log off from a VTAM application program, allowing other terminals to be activated or allowing the remaining active terminals to be responded to more quickly. Later, as network traffic decreases or as the deactivated terminals are to be given another turn, they can be reactivated and possibly other terminals can be normally deactivated. This use of normal deactivation should be defined by the system programmer and understood by the network operator and possibly by terminal operators who may wish to know why they are temporarily being denied access to the network.

Normal deactivation can be used as a routine means of ensuring that some terminals, as soon as they have logged off from the VTAM application program to which they are currently connected, are deactivated and thus unable to use the network for the remainder of the day. Meanwhile, the resources of the network can be used for other terminals that are to be activated or to remain active (perhaps in a different time zone). This use of normal deactivation also requires coordination between the network operator, VTAM application programs, and possibly terminal operators. Since a VTAM application program is not directly informed when the network operator specifies normal deactivation of a terminal to which it is connected, other means may be required to notify the program to send a message to a terminal that is logging off that it will soon be

deactivated and unable to request logon to another VTAM application program. This means could be program-to-network operator communication (a WTOR macro instruction) or a time-of-day routine.

When immediate deactivation is specified, all queued requests for connections to nodes included in an immediate deactivation request are dequeued; no new requests for connection with these nodes are accepted. All input or output operations for these nodes are immediately halted, with possible loss of data. (Data that is already in VTAM buffers prior to the deactivation can still be obtained by the application program that was connected to the terminal; data in transit to VTAM from the terminal may be lost.) If the deactivation involves a terminal connected to an application program, the application program is notified of the deactivation by the scheduling of the application's LOSTERM exit routine; the application program must disconnect the terminal for the deactivation to complete. (See "Application Program Concepts and Facilities," in Chapter 5, for details on the LOSTERM exit routine.)

The immediate mode provides very tight control over the network; normal mode provides less stringent control, but allows for a more orderly deactivation. For deactivation to complete, both modes require that application programs disconnect terminals included in the deactivation. Normal mode does not really eliminate a terminal from the network until it is disconnected; immediate mode eliminates it almost at once, regardless of whether it is connected.

Starting and Stopping an Application Program

Starting a VTAM application program requires that the network operator activate the major node containing the APPL definition statement of the application program and start the job containing the application program. The major node can be activated explicitly by using the VARY command or implicitly, when VTAM is started, by using start options. The application program job is started like any other job in the system. The order of the two operations is not critical; the major node must be active when the application program attempts to open its ACB for VTAM.

The major node containing the application program definition must remain active as long as the application program maintains its open ACB. If an attempt is made to deactivate a major node with the VARY command and all associated ACBs have not been closed, the command is rejected.

To VTAM, an application program is one that is defined within an active major node and that has an open ACB for VTAM. Thus, for VTAM, stopping an application program requires only that the ACB be closed. But the major node itself cannot be deactivated until all ACBs pointing to minor nodes in it have been closed.

An ACB is closed when the application program issues a VTAM CLOSE macro instruction. It is also closed by the operating system when the application program is terminated. An application program major node is deactivated by the VARY command and by the HALT command.

Activating and Deactivating Local 3270s

Activating a minor node for a local 3270 allocates that terminal to VTAM if it is available. If that terminal is not available (that is, if it is allocated to another, non-VTAM user), the activation request is rejected.

Activating a major node for a set of locally attached 3270s allocates those terminals to VTAM that are available, and were defined in the associated LOCAL definition statements as initially active. VTAM notifies the network operator of any terminals that are not available. The network operator can then use the VARY command to activate the minor nodes for these terminals as they become available.

Deactivating a minor node for a locally attached 3270 returns the terminal to the operating sytem. Deactivating the major node for a set of locally attached 3270s returns all active terminals in that set to the operating system. (Inactive terminals in that set are not currently allocated to VTAM.)

If an automatic logon is specified for a locally attached 3270, a logon request is queued to the application program (if it is active and accepting logons) whenever the terminal is activated. If the application program has an active logon exit routine, the routine is scheduled.

When a locally attached 3270 is activated, it is available for connection to application programs using VTAM. When the terminal is deactivated, it is unavailable for connection through VTAM, but it is available to non-VTAM users.

Activating and Deactivating Local SNA Major Nodes

A local SNA major node can be controlled by activating and deactivating the entire set of its minor nodes or individual minor nodes (its physical units and logical units). The VARY command can control the entire major node by specifying the name on the local major node definition statement (VBUILD). An individual minor node is controlled by specifying the name on a PU or LU statement.

Activating and Deactivating an NCP

Activating an NCP for a local communications controller allocates the controller to VTAM. Until the NCP is deactivated, the communications controller remains allocated to VTAM and is not available to other users of the operating system except through VTAM facilities. Allocating a local communications controller to VTAM also implicitly allocates the associated remote attachments to VTAM. (Only locally attached nodes are recognized by the operating system and need to be explicitly allocated to VTAM; the remotely attached nodes are "allocated" to VTAM because they are part of the network controlled by the local communications controllers.)

Activating an NCP can also initiate the loading of the NCP into the controller. The NCP is not loaded if the NCP specified in the activate command is already loaded and is unmodified by the network operator in its initial state. If the specified NCP is not currently in the communications controller, it is loaded.

If a currently loaded NCP is to be used, it must be in its initial state; that is, its status is that as specified for the NCP in the VTAM definition library. Thus, active physical units, logical units, cluster control units, and local 3270, BSC, and start-stop terminals and components are only those that are specified as active in the definition statements for the NCP. (If an NCP is modified by the network operator, it is not considered to be in its initial state.) VTAM does not automatically reestablish connections between application programs and terminals, though automatic logon requests are initiated for active terminals that have automatic logon specifications. Connections must be reestablished by using the OPNDST macro instruction. Also, terminals attached to switched lines have been disconnected and must be redialed.

If the activate request results in the loading of an NCP, the NCP status is as defined in the definition deck filed in the VTAM definition library. Lines are automatically activated, and active physical units, logical units, cluster control units, and local 3270, BSC, and start-stop terminals and components are those defined as active in the VTAM definition statements for the NCP. Automatic logon requests are initiated for active terminals with automatic logon specifications.

Deactivating an NCP for a local communications controller returns the communications controller to the operating system. It does not delete the loaded NCP. The installation is responsible for loading another control program if the current one is not acceptable for the next user.

Terminals attached to a remote communications controller are not available for use by VTAM until the remote controller has been activated. To activate the remote communications controller, both NCPs must be activated: first the NCP for the local communications controller and then the NCP for the remote communications controller. Before deactivating a local communications controller in a VTAM system, the network operator does not have to deactivate remote controllers. Deactivating an NCP for a local communications controller automatically deactivates any remote communications controllers.

Note: *Activating an NCP for a remote communications controller requires that the line connecting the remote controller to the local controller is active.*

If the NCP contains PEP, activation and deactivation requests may have an impact on emulation processing. Activating an NCP with PEP causes the entire NCP to be loaded (including both the emulation functions and the network control functions), although deactivating the NCP through the VARY command makes the NCP and the communications controller inactive only for VTAM. Deactivation does not halt emulation processing in the controller.

VTAM's activation and deactivation requests for telecommunication lines may also have some impact on emulation mode. VTAM controls the assignment of lines that can be reassigned between network control and emulation modes. Lines that can be reassigned are automatically assigned to emulation mode unless they are activated by VTAM. When they are activated, lines that can be reassigned but are currently assigned to emulation mode are reassigned to network control mode (except those lines being used by emulation). When they are deactivated, the lines are returned to emulation mode. See "Network Control Program Requirements," in Chapter 7, for more information on controlling an NCP with PEP.

Activating and Deactivating Remote Attachments

Activating and deactivating a minor node for a remote attachment (such as a terminal, physical unit, or line) does not affect the allocation of that attachment to VTAM. Remote attachments remain implicitly allocated to VTAM as long as the NCP for the locally attached communications controller is not deactivated by VTAM.

For VTAM to treat a terminal as active (that is, for VTAM to permit a terminal to be connected to or queued to an application program) the associated line, physical unit, and terminal must all be active. When the NCP is initially loaded, physical units, logical units, and local 3270, BSC, and start-stop terminals and components are activated as specified in the definition statements for those minor nodes. SDLC, start-stop, and BSC lines are automatically activated when an NCP is initially loaded.

Active application programs can be connected to only active terminals (logical units or local 3270, BSC, or start-stop terminals). Activating and deactivating application programs and locally attached terminals are discussed above. Activation and deactivation of remotely attached terminals are discussed below.

Although an application program can be connected to an active terminal, the connection can only be completed if the terminal is accessible through an active line, an active NCP and for an SNA terminal, an active physical unit. (If the terminal is accessible, as defined in its TERMINAL or VTERM definition statement, through more than one line, at least one of the lines must be active for connections to be completed.) Thus, a remote terminal is treated as active (that is, connected to or available for connection to an application program) only if all nodes in a valid path to the terminal have been activated.

Deactivating the NCP, the line (or lines in the case of switched networks), the physical unit, or the terminal (logical unit or local 3270, BSC, or start-stop terminal) effectively

deactivates the terminal by making it unavailable for connection to any program using VTAM. Each of these minor nodes is deactivated by using VTAM's VARY command with the deactivate option. To reactivate a terminal, the VARY command with the activate option must be issued for the minor node that was deactivated.

Activation and deactivation are essentially the same for all remotely attached terminals, whether they are attached to a local communications controller or to a remote communications controller. The only difference is that a terminal attached to a remote communications controller depends on the status of a greater number of nodes to complete an active path to an application program. In addition to the status of the NCP of the local communications controller, a terminal attached to a remote controller depends on the status of the following nodes associated with that remote controller:

- The remote communications controller itself
- The line or lines and for all SNA terminals, the physical unit connecting the terminal to the remote controller

Although there are more nodes in a path between an application program and a terminal attached to a remote communications controller, the same factor controls the active status of that terminal: all nodes in the path must be active.

Example of Deactivating and Reactivating a Remotely Attached Logical Unit

Assume an active logical unit in a nonswitched network is connected to an active physical unit which is in turn accessible on an active line. The logical unit can be deactivated by using the VARY command to deactivate any of the following:

- The NCP
- The line
- The physical unit
- The logical unit

Reactivating the logical unit depends on the node that is specified in the deactivate request. To reactivate the logical unit, a VARY activate request specifying the same node must be entered so that:

- If the NCP was specified in the deactivation request, the NCP is activated.
- If the line was specified in the deactivation request, it is activated.
- If the physical unit was specified in the deactivation request, the physical unit is activated.
- If the logical unit was specified in the deactivation request, the logical unit is activated.

If a physical unit or logical unit is effectively deactivated by deactivating the NCP, the physical unit or logical unit can only be reactivated automatically when the NCP is reactivated if they are defined by VTAM definition statements as being initially active when the NCP is loaded. A physical unit or logical unit not automatically activated must be explicitly reactivated.

Activating and Deactivating Switched Major Nodes

Switched major nodes can be activated or deactivated by using the VARY command for the entire switched major node or portions of the switched major node. The VARY command can control the entire switched major node by specifying the name on the switched major node definition statement (VBUILD). The portions of a switched major node that use the dial-out facility can be controlled by activating and deactivating an individual path by specifying the path identifier (PID), or a group of paths by specifying the group identifier (GID). For example, all paths that use WATS lines for dial-out operations can be associated with a GID and activated or deactivated as a unit.

Dial-in operations can be controlled by activating or deactivating a line's ability to answer a dial-in request. Lines used for dial-in operation can be activated or deactivated; lines used for dial in or out operation can be changed to dial-out only.

For additional information on controlling switched major nodes, see "Network Control Program Requirements," in Chapter 7.

Activating and Deactivating SNA Physical Units

SNA physical units supported by VTAM must be loaded during initial program loading before they can be activated by VTAM. A 3601 controller, for example, must be loaded before it can be activated, and the controller must be active before any associated logical unit can be activated.

Special Considerations for Activation

The activation of nodes has special implications that, in part, result from the way VTAM controls its system. When VTAM activates a major node, it builds a segment of a table containing entries that describe the minor nodes defined for that major node. (When the major node is deactivated, the table segment for that node is deleted from main storage.) VTAM uses the entries in the table to represent the actual minor nodes. It is these entries that VTAM allocates to users; VTAM does not allocate the node the entry represents, but it retains the ownership of all the nodes. In most instances, the status of the minor node and of its representative table entry is the same; thus, no distinction is usually necessary between a node and its table entry. The activation of application programs and the activation of local 3270, BSC, and start-stop terminals are two examples of this distinction.

The implications of an active application program and of an active terminal are discussed below. No distinction is made in this discussion between activation that occurs automatically as a result of VTAM definition options or dynamically because of network operator requests.

An Active Application Program: VTAM activates only major nodes for application programs. Once an application program major node is active, each table entry for a minor node within it is available to form a connection between VTAM and an actual application program. This connection is made when an ACB (pointing to one of these entries) is successfully opened. A table entry for an application program minor node can be used to form a connection with VTAM by only one application program at a time; that is, only one open ACB at a time can point to it.

VTAM's VARY command and the start options for activating nodes activate only the major nodes for application programs. The installation must start the application program and open the ACB. When the ACB is open, VTAM treats it and its associated table entry as an active application program.

An Active Terminal: When VTAM activates a terminal, it indicates in the terminal's table entry that the terminal is active and either transmits an activation command to the NCP (for remotely attached logical units) or obtains the terminal from the operating system (for locally attached terminals). When an application program connects with a terminal, VTAM connects the application program to this table entry, while retaining the ownership of the terminal itself.

In the case of start-stop and BSC terminals, the physical status of the terminal can differ from that of its table entry; for example, a table entry can be marked active even though the terminal it represents has not been turned on or is not even physically in the network. But, because VTAM allocates the table entry to the application program when completing a connection request, an application can connect to a terminal that is not physically part of the network. Connection is possible in this case if the table entry is marked active and a defined path has been activated.

If connection is requested to a nonexistent (but defined) terminal, the connection is completed, but no data transfer can be completed. These conditions enable an installation to include, in the definition of the network, resources that are not physically available but will be added at a later date. By doing this, an installation can avoid redefining the network or recoding application programs for the addition of minor nodes to the telecommunication system.

For logical units on nonswitched lines, the status of the table entry and the terminal it represents must agree. Thus, when a logical unit is activated, it must be physically active before its table entry can be activated and made available for use by an application program. When VTAM activates a local 3270, BSC, or start-stop terminal, the activate request completes even if no other node in the terminal's path is active except the NCP. (A terminal without an active path is still not available for connection to an application program; in effect, the terminal is inactive for application programs.)

Initiating Requests for Connection

In addition to its use in activating and deactivating nodes, VTAM's VARY command can initiate connection requests on behalf of terminals. This logon option is used to initiate VTAM's network operator logon.

To initiate a network operator logon, the network operator enters a VARY command containing the names of the terminal and the application program to be connected. VTAM then processes the command as though it were an automatic logon for the terminal; VTAM notifies the application program that a logon request was received from a terminal, but the application program must then accept the request before connection is completed. (See "Application Program Concepts and Facilities," in Chapter 5, for details on connecting application programs and terminals.)

In addition to initiating a logon request, this option of the VARY command alters the automatic logon specification for the terminal. For example: If the network operator initiates a logon request on behalf of terminal 1 for application program A, terminal 1 is initially logged on to that program. Thereafter, whenever terminal 1 is available, it is automatically logged back on to application program A. This automatic logon specification for terminal 1 remains in effect until the network operator either deactivates the major node containing the terminal definition or enters a new logon request on behalf of the terminal.

Because the network operator logon modifies the automatic logon specification, care should be exercised when using the logon facility of the VARY command.

Starting and Stopping VTAM Facilities

Some of the facilities in VTAM need not be active continuously. VTAM allows the network operator to start and stop these facilities selectively.

Using VTAM's MODIFY command, the network operator can start the following VTAM facilities:

- Network solicitor, a facility for local 3270, BSC, and start-stop terminals described in Chapter 8
- Trace facility
- Print-trace utility program
- Dump utility program
- Teleprocessing Online Test Executive Program (TOLTEP)

The MODIFY command can be used to stop:

- Network solicitor
- Trace facilities

The network solicitor and the trace facilities can also be activated by VTAM start options. See "Starting VTAM," earlier in this chapter, for more information on activating these two facilities at start time.

Starting and Stopping the Network Solicitor

The MODIFY command can be used to start the network solicitor. This network solicitor must be the IBM-supplied network solicitor or the network solicitor that was modified by the NETSOL macro instruction (with the name NETSOL).

Activating the network solicitor automatically logs on all appropriate available terminals that meet all the following requirements:

- They are active but not connected, nor queued for connection, to another program.
- The automatic logon specification in their node definition indicates the network solicitor.

As long as the network solicitor remains active, it continues to accept appropriate available terminals. It attempts to solicit logon requests from these terminals and passes valid logon requests to the appropriate application programs.

Deactivating the network solicitor with the MODIFY command causes the network solicitor to complete handling all terminal-initiated logon requests in process and to disconnect all other terminals connected to it. No additional automatic logons are accepted by the network solicitor until it is reactivated.

Starting and Stopping Traces

The MODIFY command can be used to start or stop traces for NCP major nodes as well as for the following minor nodes:

- Telecommunication lines
- Cluster control units
- Local and remote terminals
- Components

If the node specified in the command is a telecommunication line, the network operator can only request an NCP line trace. For all other nodes, either the buffer trace or the I/O trace can be requested.

Nodes included in a start-trace request are traced only when active. Deactivating a node stops any traces for it. If the node is reactivated and the trace is still needed, the trace request must be reissued. (See "Serviceability Aids," in Chapter 6, for detailed descriptions of VTAM traces.)

Starting the Trace-Print Utility Program

In DOS/VS, VTAM provides a utility program to selectively edit and print trace records accumulated by VTAM's trace facility. This utility program is also started by an option of the MODIFY command. Once this utility has been started by the MODIFY command, it prints all trace records specified by the operator in the trace data set. Tracing is suspended for the affected nodes until the printing has been completed.

In OS/VS1, VTAM traces are printed using the operating system's HMDPRDMP service aid. Refer to the publication *OS/VS1 Service Aids,* GC28-0665, for information on

HMDPRDMP. In OS/VS2, VTAM traces are printed using the operating system's AMDPRDMP service aid. Refer to the publication *OS/VS2 System Programming Library: Service Aids,* GC28-0633, for information on AMDPRDMP.

Starting VTAM's Dump Utility Program

Another option of VTAM's MODIFY command enables the network operator to initiate the execution of the NCP dump utility program tailored for a VTAM environment. Using this option, the network operator specifies the NCP to be dumped. VTAM initiates loading of the NCP dump utility program into the communications controller. The dump utility program then provides the requested dump. An operating system print utility can print the dump.

Caution should be exercised when requesting a dump of an NCP. The dump program erases part of the NCP, with or without PEP, and deactivates it, requiring that the control program be reloaded if it is to be used after the dump is taken. Thus, in the case of an NCP with PEP, the emulation function, as well as the network control function, is affected by the dump request.

If an NCP dump is to be initiated by VTAM, the dump data set, LUB name in DOS/VS, must have been specified in the NCP's PCCU statement during VTAM definition. See "Network Control Program Requirements," in Chapter 7, for a description of the PCCU statement.

Starting TOLTEP Testing

The Teleprocessing Online Test Executive Program (TOLTEP) is a component of VTAM designed to test lines and devices in the network. TOLTEP testing can be initiated from the network operator's console by using VTAM's MODIFY command or by using the logon option of the VARY command.

TOLTEP testing can also be requested by an operator at an active terminal in the VTAM network. The network operator is notified of the request and must authorize the request before the testing can continue.

See "Serviceability Aids," in Chapter 6, for a detailed description of TOLTEP and how it is controlled by the network operator.

Suppressing Network Operator Messages

VTAM allows an installation to suppress certain classes of network operator messages. Suppression can be specified when starting VTAM, either as a predefined start parameter or by the operator, or when operating VTAM by using the network operator MODIFY command. Suppressing messages allows an installation to adjust the amount of information it receives from VTAM to suit the needs of the operator.

Messages are divided into these classes:

- Information
- Warning
- Serious
- Normal

Any of these classes and classes incorporated within that class can be selected for suppression. For example, if serious class messages are selected for suppression, information and warning messages will also be suppressed; only normal class messages will be sent to the network operator.

Messages that prompt information from the operator and messages that respond to a DISPLAY command cannot be suppressed.

Changing Line-Scheduling Specifications

VTAM permits the network operator to change the following line-scheduling specifications for polled, nonswitched start-stop or BSC lines:

- Polling delay
- Negative poll response limit
- NCP session limit
- Device transmission limit

The first three specifications are made on the NCP's LINE statement. The last specification is made on the NCP's TERMINAL statement. Refer to the *NCP Generation* publication for a description of these parameters and their functions.

The line-scheduling specifications are changed by the network operator by using an option of VTAM's MODIFY command.

A line-scheduling change remains in effect until the communications controller is reloaded. If the reloading employs NCP restart (as the result of an NCP error), the change remains in effect for the reloaded NCP. If the reloading does not employ restart, the line-scheduling parameters specified for NCP generation are reinstituted.

Considerations for Network Operator Control

To use VTAM's network operator facilities effectively, the installation must establish control procedures and provide information to the network operator. In general, the network operator must be familiar with the configuration of the telecommunication network, know how to manipulate VTAM to attain the installation's teleprocessing objectives, and know what actions to take when problems are encountered in the system. Planning by the installation should include the following areas:

Names of major and minor nodes and data sets: Because VTAM manipulates symbolic node names, the network operator must know the names of all major and minor nodes that are to be used in VTAM commands or that can be received in VTAM messages. The operator should also know the names and uses of all VTAM data sets. See "Major and Minor Node Structure," in Chapter 3, for a discussion on naming nodes. (See "VTAM Data Sets," in Chapter 7, for a description of the data sets used by VTAM.)

Relationship between the names of VTAM application programs and job names: Application programs are identified to VTAM through APPL statements and ACBs. For VTAM, the name of an application program is the name of an APPL statement; for the operating system, the job name or the step name is the program name. The network operator needs to know how each application program name in VTAM relates to job names or step names.

Relationship between the symbolic and the physical network: Because VTAM commands function differently for major nodes than for minor nodes, the network operator should know which nodes are major and which are minor. Also needed is a knowledge of the hierarchical structure of the NCP nodes and of the relationship between the symbolic names and the physical units they represent.

Impact of VTAM on emulation in an NCP with PEP: Although VTAM does not use emulation mode, the access method can have an impact on the operation of emulation mode in an NCP with PEP. (See "Network Control Program Requirements," in Chapter 7, for a description of VTAM's impact on emulation mode.)

Switched network support: Controlling a switched network requires an understanding of how such a network is defined. See "Network Control Program Requirements," in Chapter 7, for details on defining and controlling switched lines and terminals that use them.

Storage sizes: When the operating system is loaded, the VTAM region or partition size can be selected. VTAM also allows the network operator to make storage pool specifications. If these two specifications are not predefined to the system or to VTAM, the network operator needs the detailed storage information.

Procedures: An installation may also want to establish procedures for the operator to follow. The procedures can be for both planned normal operations and contingencies. The procedures might cover: when and how to start and stop VTAM, application programs, and the network solicitor; when, how, and which terminals and other nodes to activate and deactivate; and what actions to take when network errors are encountered, including what traces to activate, how to activate when and how to dump an NCP.

The above list is not meant to be exhaustive because each installation has different requirements. The list calls attention to the need for setting up procedures and familiarizing the operator with the network.

CHAPTER 5. WRITING VTAM APPLICATION PROGRAMS

The purpose of this chapter is to introduce VTAM application program concepts and facilities so that an installation can understand what is involved in writing a VTAM application program.

The first section of this chapter provides an overview of VTAM application programs and a summary of VTAM macro instructions. The second section explains VTAM programming concepts and facilities. The third section explains in more detail the VTAM language and its use. The final section shows and discusses the general logic of two sample application programs.

The reader planning to write VTAM application programs may wish to go directly to *VTAM Macro Language Guide*, GC27-6994, rather than reading this chapter, to become acquainted with VTAM application programs. *VTAM Macro Language Guide* to some extent duplicates the information provided in this chapter and expands upon it.

Many of the VTAM application program concepts and facilities described in this chapter for logical units apply as well for local 3270 terminals and supported BSC and start-stop terminals. An installation considering communicating with these terminals using VTAM should read this chapter first, and then read Chapter 8.

A VTAM Application Program Overview

This section provides an overview of the VTAM application program from the perspective of an installation's telecommunication system. It summarizes VTAM macro instructions, and it relates VTAM application programs to those of TCAM and BTAM.

The VTAM Application Program in Relation to the VTAM System

Figure 5-1 shows where a VTAM application program fits into a teleprocessing system. The numbers in Figure 5-1 refer to major elements in the system. VTAM application-program connection to start-stop or BSC terminals is possible but is not shown. The following discussion is keyed to Figure 5-1.

The VTAM Application Program (1)

In a teleprocessing system, the two main activities are the processing of data and the transmission of data. A VTAM application program uses VTAM macro instructions to transmit data between the host CPU and the telecommunication network, and it uses other instructions to process data. VTAM permits the separation of the processing part of an application program from the telecommunication part. (If the parts are separate, an interface between them needs to be defined.) This separation of function allows each part to be created separately and means that changes or additions to one part do not affect other parts. The processing part of a VTAM application program may be written in a higher level language, such as PL/1; the telecommunication part, using VTAM macro instructions, is written in assembler language. This chapter, describing the concepts, facilities, and language of VTAM, concerns itself mainly with the telecommunication part of the VTAM application program.

As shown in Figure 5-1, more than one VTAM application program can concurrently share the resources of a VTAM system. For example, each of the VTAM application programs in Figure 5-1 can communicate with different logical units on the same transmission line; because the actual transmission is managed for the programs, neither program is aware that line sharing takes place or of the line's identity.

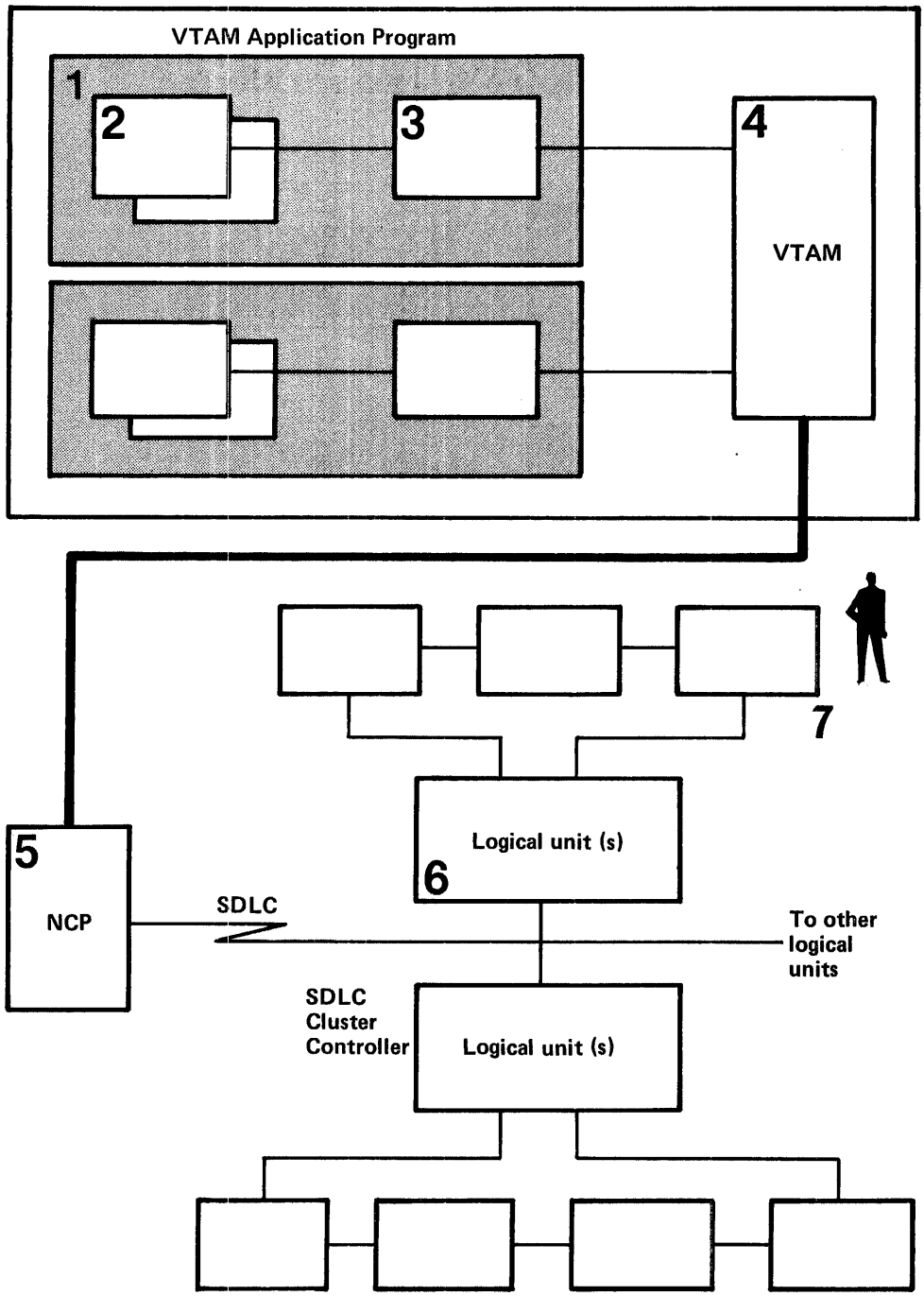


Figure 5-1. VTAM Application Programs in Relation to the Telecommunication System

The Processing Part (2)

This part of the program can be written in assembler language or in a higher-level language, such as PL/1 or COBOL. If this part is written in assembler language, it can be interleaved with the telecommunication part of the program. As shown in Figure 5-1, it is separate and requests telecommunication services by calling or branching to the telecommunication part of the VTAM application program.

The processing part of the application program manipulates data in response to a request from a remote location. Such data manipulation might include updating a data base, obtaining information from a data base, or formatting information to be displayed for an operator at a terminal. If both the logical unit and the application program in the host CPU are programmed, it is possible to distribute the data processing function between these two nodes. For example, the application program in the host CPU can obtain data from the data base while the logical unit can format the data for display to the terminal operator.

For programmable logical units, each installation decides which processing functions are to be performed by the processing part of the VTAM application program and which are to be performed by an application program that associated with a logical unit, such as an application program in a 3601 Finance Communications Controller.

The Telecommunication Part (3)

This part of the VTAM application program contains macro instructions and associated control blocks used to connect and communicate with logical units that have been defined to VTAM. Chapter 5 is concerned primarily with this part of the VTAM application program.

VTAM (4)

VTAM controls the VTAM telecommunication system. Logical units are defined as part of the VTAM system during VTAM definition and are activated by start procedures or network operator commands. The VTAM application then requests connection (on its own initiative or as the result of a logon request) to one or more active logical units. Once connected, the program requests that VTAM perform data transfer with logical units. In addition to building channel programs, VTAM performs such services as input and output data buffering, automatic scheduling of application program exit routines, and sequence numbering of outbound messages. (These facilities are discussed further in this chapter.) VTAM requests that the operating system execute channel programs it has built; the channel programs result in communications with locally attached logical units or with remote logical units through a locally attached communications controller.

The Network Control Program (5)

The network control program (NCP) in the 3704 or 3705 Communications Controller receives input and output requests and associated data or information, and communicates with logical units on telecommunication lines. The communications controller performs many functions previously performed by the access method or application program. For example, the communications controller schedules line activity, retries transmission-errors, and collects error statistics. Communication with logical units is performed using SDLC.

The Logical Unit (6)

The VTAM application program communicates with a logical unit. A logical unit, depending on the type of SNA terminal product, can be:

- Logic in a device used directly by a terminal operator.
- Logic in a terminal system application program. This logic can be associated with a particular work station, a particular series of transactions, or have other meanings.

VTAM provides application programs in the host CPU with a set of macro instructions to communicate with logical units. Although not shown in Figure 5-1, VTAM application

programs also communicate with certain terminals on start-stop and BSC lines. VTAM provides another set of macro instructions to communicate with these terminals.

Additionally, VTAM application programs can communicate with BSC and local 3270 terminals as though the terminals were logical units. Thus, the same set of macro instructions can be used to communicate with BSC and local 3270s and logical units.

In general, for programmable SNA terminals an installation defines which processing functions take place in the terminal system and which in the VTAM application program in the host CPU. An installation must coordinate the terminal system programs and the VTAM application programs so that they can work together.

The Terminal Operator or Batch Function (7)

The VTAM application program may communicate with a logical unit that is directly associated with a terminal operator. Since a VTAM application program communicates with a logical unit rather than with a terminal operator at a physical device, the VTAM application program may not need to know about terminal operator actions. The logical unit will determine whether and how data received from a terminal operator goes to the VTAM application program and whether and how data received from the VTAM application program goes to the terminal operator.

A VTAM application program may also be communicating with a batch function such as the 3791 batch function. The VTAM application program need not be aware of the source or disposition of data received from or sent to a subsystem batch function.

A Summary of VTAM Macro Instructions

The telecommunication part of a VTAM application program uses IBM-provided macro instructions to request VTAM services. These macro instructions, summarized below, are referred to later in this chapter in "Application Program Concepts and Facilities" and discussed in more detail in "The VTAM Language."

Connection Macro Instructions

These macro instructions connect and disconnect an application program with VTAM and with logical units and start-stop and BSC terminals.

OPEN: Connects a VTAM application program to VTAM.

CLOSE: Disconnects a VTAM application program from VTAM.

OPNDST: Connects a logical unit or BSC or start-stop terminal to a VTAM application program.

CLSDST: Disconnects a logical unit or BSC or start-stop terminal from a VTAM application program.

Communication Macro Instructions

These macro instructions request and control the transmission of data between an application program and its logical units. (Communication macro instructions for BSC and start-stop terminals are not included here; they are discussed in Chapter 8.)

RECEIVE: Requests that input received from a logical unit be passed to the VTAM application program.

SEND: Requests that output be sent to a logical unit from the VTAM application program.

RESETSR: Changes the mode of receiving input from a particular logical unit. The modes are continue-any mode (have input from the logical unit satisfy an outstanding RECEIVE

that specifies input from any logical unit) and continue-specific mode (have input satisfy a RECEIVE that specifies only that particular logical unit). RESETSR also cancels outstanding RECEIVEs that request input from the specific logical unit.

SESSIONC: Initializes and controls the flow of information between the VTAM application program and a logical unit.

Control-Block Macro Instructions

The two following categories of macro instructions build and manipulate VTAM control blocks. Figure 5-2 shows the relationship between the VTAM control blocks when a logical unit is being connected to an application program.

Declarative Macro Instructions: These macro instructions assemble control blocks when the application is assembled. The macro instructions are also the names of the VTAM control blocks themselves.

ACB: Defines the characteristics of a VTAM application program to VTAM

EXLST: Defines the list of exit routine addresses that VTAM uses to notify the program when certain conditions occur, such as a logical unit request for logon (connection) to the program

NIB: Defines information pertaining to a particular logical unit; VTAM stores this information for use throughout the time that the logical unit is connected

RPL: Defines the parameters and contains the results of a particular request (for example, OPNDST or SEND)

Manipulative Macro Instructions: These macro instructions dynamically build and manipulate control blocks during the execution of the application program.

GENCB: Builds and can initialize an ACB, EXLST, NIB, or RPL control block during program execution

MODCB: Changes the contents of a control block field to a designated value

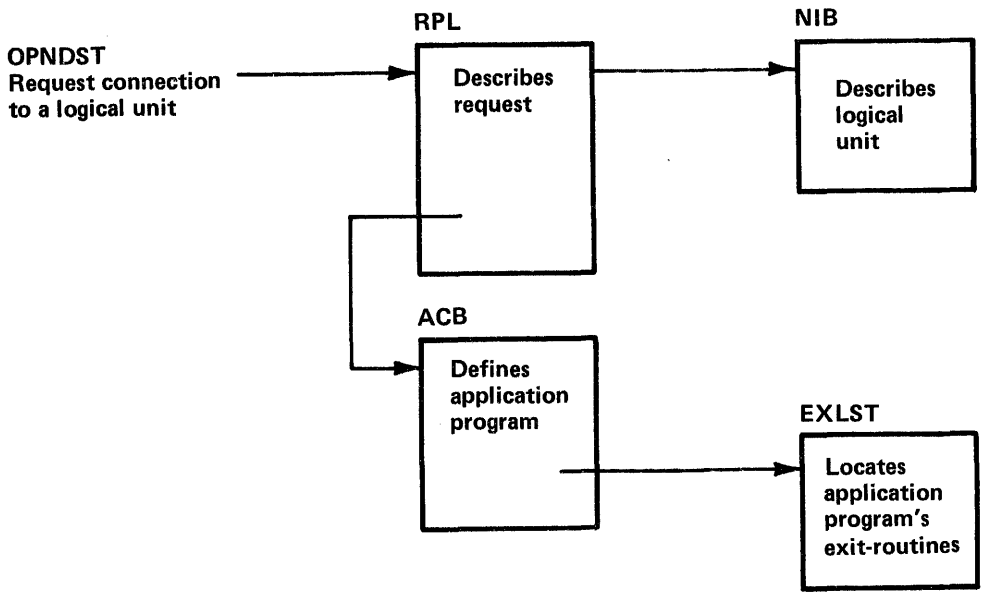


Figure 5-2. Relationship of VTAM's Control Blocks in an Application Program

SHOWCB: Moves the contents of a control block field to a designated area

TESTCB: Tests the contents of a control block field and sets the program status word (PSW) condition code accordingly

The VTAM user can also gain access to VTAM application program control blocks by using DSECT and other assembler language instructions.

Support Macro Instructions

These macro instructions request additional VTAM services.

EXECRPL: Requests that VTAM perform an operation specified in a designated RPL. It is designed primarily for reissuing previous requests from an error-recovery routine. EXECRPL can also be used instead of other RPL-based macro instructions such as OPNDST, SEND, and RECEIVE, or, after an unsuccessful operation

CHECK: Checks and, if necessary, awaits completion of a requested operation

INQUIRE: Requests information from VTAM, such as the status of another application program

INTRPRET: Obtains information from an installation-defined interpret table

SETLOGON: Tells VTAM when to permit logon requests for the VTAM application program

SIMLOGON: Allows the VTAM application program itself to initiate a logon request. If necessary, it also requests the current owner of the logical unit to release it

A Brief Comparison with TCAM and BTAM

The following is a brief comparison of programs that use VTAM macro instructions with those that use TCAM or BTAM macro instructions.

VTAM Compared with TCAM

TCAM provides services for handling telecommunication input/output by the telecommunication part of a program (a TCAM message control program, including its message handler) and by the processing parts of a program (TCAM application programs). A TCAM programmer uses TCAM macro instructions to write a message control program. A message control program queues messages until another application program requests them or until they can be sent to a telecommunication device. Queuing can be in main storage or on a direct access storage device. The programmer uses GET/PUT and READ/WRITE macro instructions in the TCAM application program to request the services of the message control program.

VTAM provides only direct telecommunications services for the VTAM application program; the VTAM application programmer must create his own control blocks and logic for the queuing of data, if necessary, and must provide his own interfaces between the telecommunication and processing parts of the program. Figure 5-3 shows how TCAM relates to VTAM.

VTAM services are available to application programs using TCAM; this is described in "Other Telecommunication Access Methods," in Chapter 7.

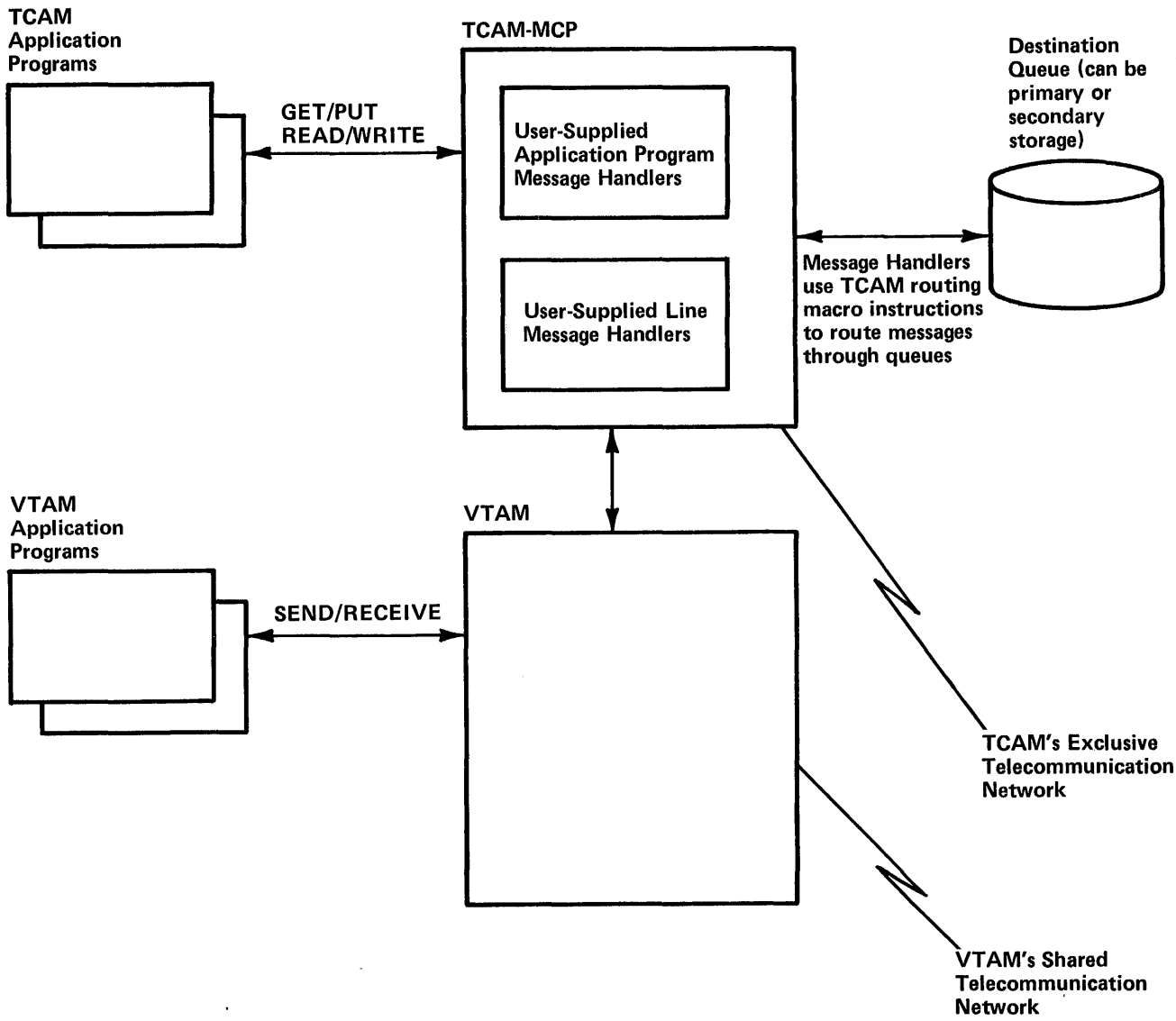


Figure 5-3. Relationship of TCAM to VTAM

VTAM Compared with BTAM

Figure 5-4 shows some of the major similarities and differences between VTAM application programs and BTAM application programs. VTAM application program characteristics shown in Figure 5-4 are discussed in more detail further in this chapter.

Application Program Concepts and Facilities

This section introduces and explains VTAM programming concepts and facilities and relates them to the VTAM language (macro instructions and operands).

VTAM's primary purpose is to provide communication between the application program and logical units, local 3270 terminals, and supported BSC and start-stop terminals. This section discusses logical units, but the facilities apply, except where noted, to local 3270, BSC, and start-stop terminals in the telecommunication network. VTAM also provides a means by which the installation can establish a system of dynamically allocating logical units to the application programs. The concepts and facilities described in this chapter all deal with application programs. The concepts and facilities described in this chapter all deal with these two aspects of an application program using VTAM—the application program's ability to have a logical unit allocated to it, and its ability to communicate with that logical unit.

Characteristic	VTAM Application Program	BTAM Application Program
General Characteristics of the Access Method		
Terminal-sharing	Yes	No
Line-sharing	Yes	No
Line-scheduling logic required	No	Yes
Polling required on part of application program	No	Yes
Exit-routines scheduled for special conditions such as a logon request	Yes	No
Local/remote 3270 handling	Can be communicated with in same mode as logical units; no coding distinction between local and remote	Requires different coding for local 3270 than for remote 3270
Primary type of terminal communication	Communicates primarily with logical units (program logic). Can also communicate with start-stop and BSC terminals	Communicates primarily with start-stop and BSC terminals
Direct-control or queued access method	Direct-control	Direct-control
Language Characteristics	VTAM macro instructions	BTAM macro instructions
	Keyword operands	Positional operands
	Macros common for DOS/VS and OS/VS	Macros different for DOS/VS and OS/VS
	Two sets of I/O macro instructions: record-mode (RECEIVE-SEND) for logical units and basic-mode (READ-WRITE) for BSC and start-stop devices	One set of I/O macro instructions
	Destination-oriented	Line-oriented
Program Organization	Telecommunication normally separate from processing	Telecommunication normally separate from processing
	Can have VTAM post an ECB or schedule an exit-routine when I/O event completes	BTAM posts an ECB when I/O event completes

Figure 5-4 (Part 1 of 2). Major Similarities and Differences Between VTAM and BTAM Application Programs

Characteristics	VTAM Application Program	BTAM Application Program
Functions Provided		
Define line groups		Use DCB or DTFBT macro instruction
Define terminals	All resources are defined during VTAM definition	Use DFTRMLST macro instruction
Initialize program	Use OPEN macro instruction	Use OPEN macro instruction
Connect logical units dynamically to application program	Use OPNDST macro instruction	Not available, because terminals are statically connected when application's job step is initiated
Release control of a logical unit to another program that requests connection to it	When requested, use CLSDST with OPTCD=RELEASE specified	No comparable function
Receive input	Use RECEIVE macro instruction	Use READ macro instruction
from any connected logical unit/ device	Specify input can be from any logical unit or device	Must poll each line separately
from a specific logical unit/ device	Specify input is to be received from a specific logical unit or device	Specify terminal list entry
Send output	Use SEND macro instruction	Use WRITE macro instruction
Have data scheduled for output from access method buffers (output buffering)	Specify that the data is to be scheduled for output	No comparable function
Test, display, or modify control block fields	Use manipulative macro instructions: TESTCB, SHOWCB, MODCB. Or use assembler language instructions	Use assembler language instructions
Record transmission errors	Performed by NCP and VTAM	Use error recording macro instructions

Figure 5-4 (Part 2 of 2). Major Similarities and Differences Between VTAM and BTAM Application Programs

The facilities described first are those that apply generally to all of the application program's interactions with VTAM, whether they involve logical unit allocation (connection), or communication, or some request that is only indirectly related to allocation or communication. After a brief discussion of application program initialization, concepts and facilities relating to connection and to communication are discussed in detail.

General Concepts and Facilities

The application program issues macro instructions to request that VTAM perform some operation. VTAM communicates with the application program by setting register return codes, control blocks, and parameter lists, by posting ECBs, and by invoking special user-written routines. The manner in which VTAM communicates with the application program is to a considerable degree controlled by the application program. The following facilities deal with the various ways that VTAM can be directed to handle the application program's requests and ways that the application program can be notified that particular events in the network have occurred.

Overlapping VTAM Requests with Other Processing

Each VTAM macro instruction request issued by the application program can be handled *synchronously* or *asynchronously* by VTAM.

Synchronous request handling means that VTAM returns control to the next sequential instruction only after the requested operation has completed. Program execution is halted until VTAM determines that the operation has been completed. This type of request handling is appropriate for applications that cannot continue processing until a particular request has completed. Figure 5-5 illustrates synchronous request handling.

Asynchronous request handling means that VTAM returns control to the next sequential instruction as soon as VTAM has accepted the request, not when the requested operation has been completed. Accepting a request consists of screening the request for errors and scheduling the parts of VTAM that will eventually carry out the requested operation. While the operation is being completed, the application program is free to initiate other I/O transactions or processing. For example, an application program might issue a RECEIVE macro instruction and indicate that it is to be handled asynchronously; while the input operation is being completed, the application program could begin to write to a direct access storage device or communicate with another logical unit.

When asynchronous request handling is used, there are two ways that VTAM can notify the application program that the requested operation has completed. If the application

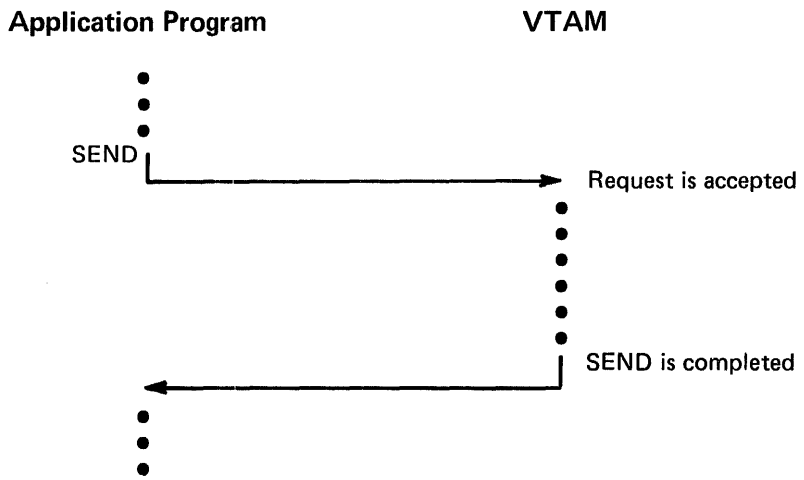


Figure 5-5. Processing Pattern for a Synchronous Request

associates an event control block (ECB) with the request, VTAM posts the ECB when the operation is completed. The application program can use a CHECK or WAIT macro instruction to determine that the ECB has been posted. Alternatively, the application program can associate an RPL exit routine with the request. When the operation is completed, VTAM invokes the routine. Figure 5-6 illustrates asynchronous processing in an application program using ECBs; Figure 5-7 illustrates the use of the RPL exit routine to control processing.

By using ECBs, the application program can use one WAIT macro instruction for a combination of VTAM requests and any non-VTAM requests that use ECBs. For example, an application program could issue three VSAM requests and three VTAM requests; by issuing one WAIT for all six ECBs, the application program can continue processing when any one of the six operations is completed.

ECBs also give the application program the freedom to determine during program execution when the program should stop processing and wait for a given operation to be completed. An application program might, for example, request data from a logical unit and then later determine that it should not stop and wait for that data (perhaps the application program has in the meantime begun to request data from another logical unit whose input is of a much higher priority and must be handled immediately). ECBs also allow the application program to assign priorities to requests by checking some ECBs before checking others.

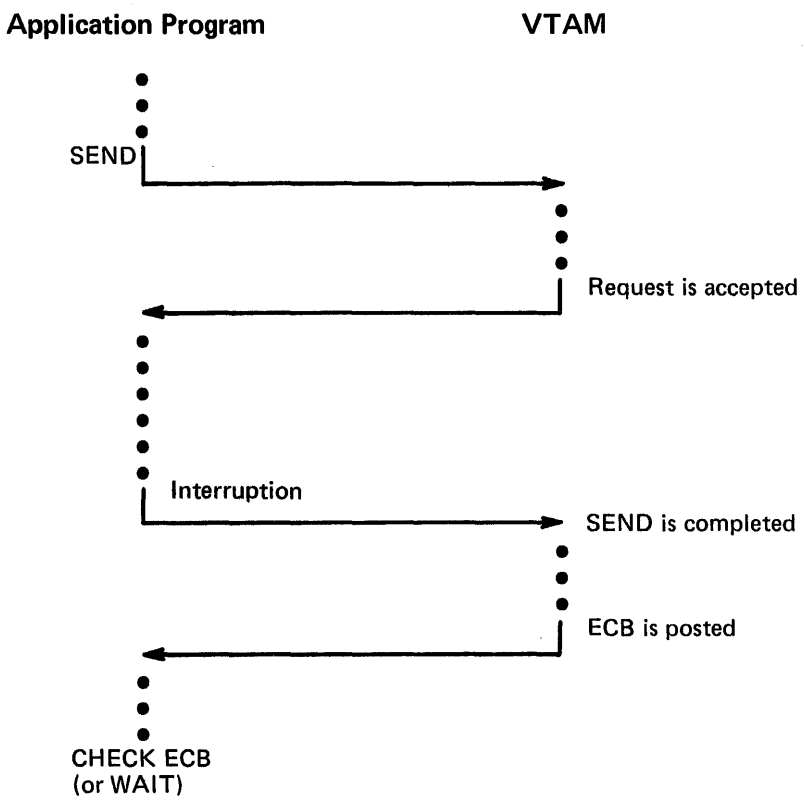


Figure 5-6. Processing Pattern When an ECB is Used with an Asynchronous Request

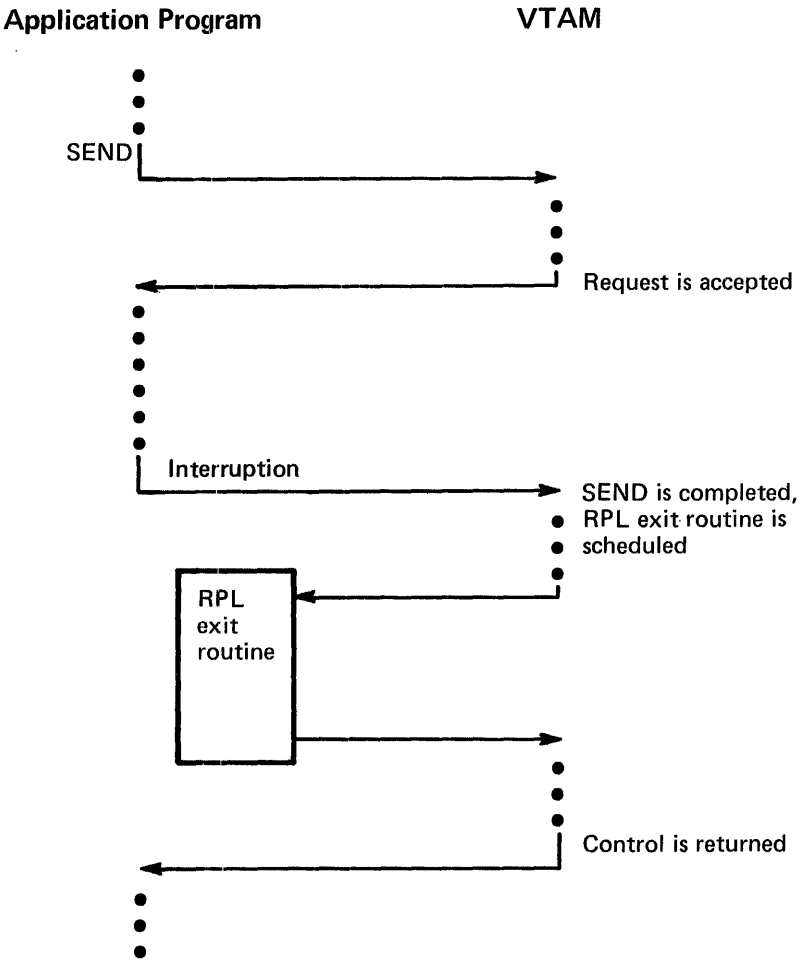


Figure 5-7. Processing Pattern When an RPL Exit Routine is Used with an Asynchronous Request

The distinction between ECBs and RPL exit routines rests primarily on the fact that the RPL exit routine is *automatically* scheduled when the requested operation is completed, thereby saving the application program the trouble of checking ECBs and branching to subroutines. ECBs provide greater control, while RPL exit routines provide greater convenience.

VTAM requests issued in the RPL exit routine can also be handled asynchronously, although an exit routine is not scheduled if another exit routine has not completed. Figure 5-8 illustrates how an application program might use this facility.

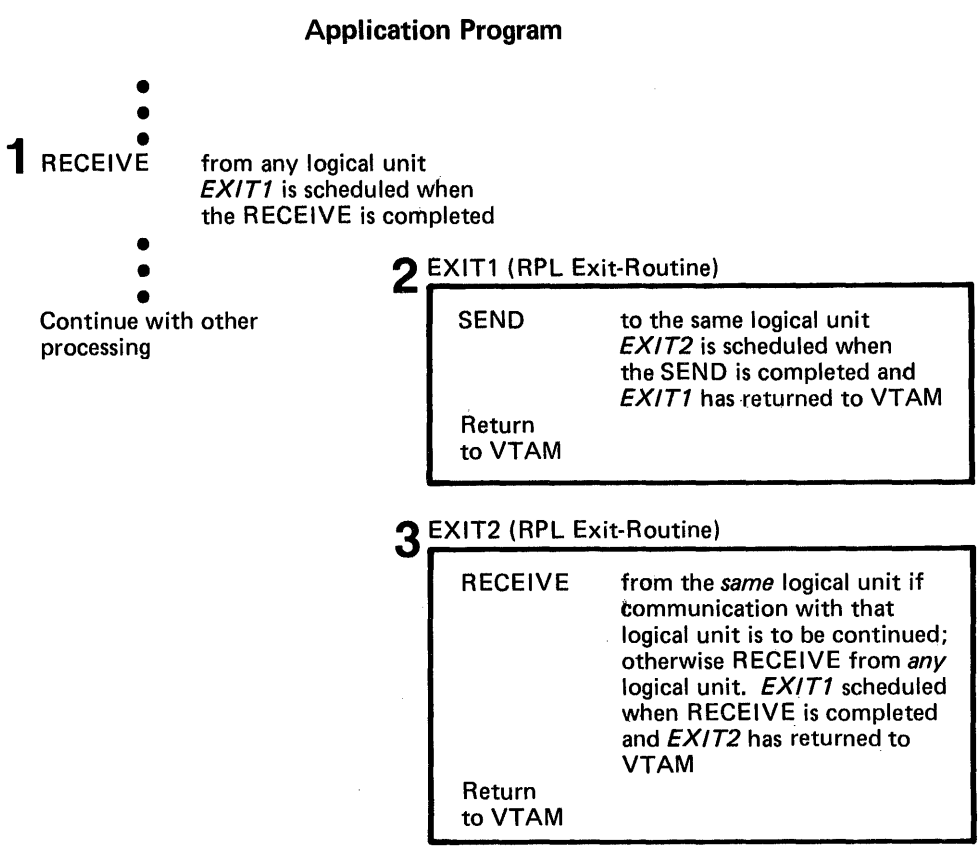


Figure 5-8. A Possible Processing Pattern When Asynchronous Requests Are Issued in RPL Exit Routines

Application Program Exit Routines

The application program can use the EXLST macro instruction to identify a variety of exit routines that VTAM schedules when particular events occur:

Event	Routine
A logical unit is waiting to be connected to the application program	LOGON
One of the logical units has withdrawn (or been withdrawn) from the network	LOSTERM
One of the logical units is wanted by another application program	RELREQ
The network operator is shutting down the network	TPEND
A start-stop terminal has caused an attention interruption	ATTN
A special type of input has arrived in the CPU (the types of input are discussed later)	DFASY RESP SCIP

When one of these events occurs, the execution of the application program is interrupted and the appropriate exit routine is given control. If another event occurs while the exit routine is still processing, the next exit routine is not invoked until the first has completed (this applies as well for RPL exit routines).

Unlike the installation exit routines (discussed in Chapter 3), which are included as part of the system during VTAM definition, the application program exit routines are included as part of the application program. The addresses of these routines are placed in an EXLST control block by the application program.

Exit routines other than the LERAD and SYNAD exit routines need be reenterable only if two or more application programs share the same exit routine.

In OS/VS, each exit routine is usually scheduled under an interruption request block (IRB); in DOS/VS, each exit routine is scheduled by changing the program information block (PIB) save area address. Any processing in the routine that places the routine in a wait state should be used with caution, since the application program's entire task waits while the exit routine waits.

Error Notification

When synchronous request handling is used, error conditions are reported when the request has been completed and control returned to the application program. When asynchronous request handling is used, error conditions are reported in two stages. When control is first returned to the application program, VTAM indicates whether it has accepted or rejected the request. When an accepted operation completes, VTAM posts an ECB or schedules a designated RPL exit routine, and, on the program's issuing a CHECK macro instruction, returns information about the completion of the requested operation. Figure 5-9 shows how errors are reported during asynchronous processing.

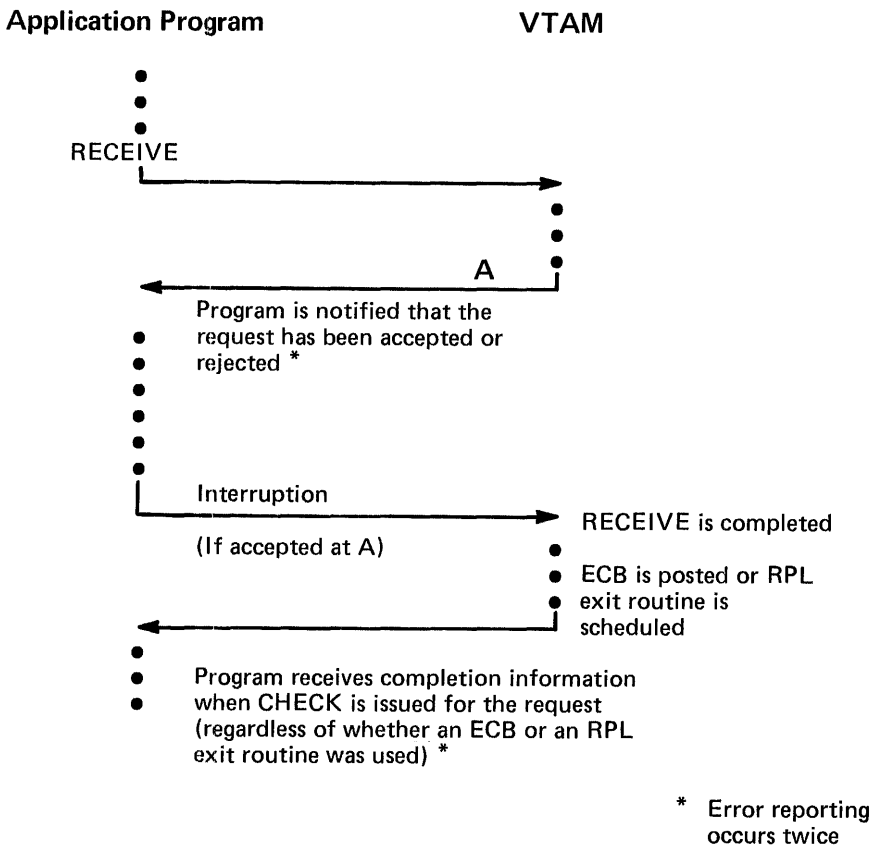


Figure 5-9. Processing Pattern For Reporting Errors During an Asynchronous Operation

Information about success or failure is sent to the VTAM application program in register 15 and, under some circumstances, in register 0. In addition, if the request or operation was not successful, VTAM will normally have placed additional information in return code fields of the RPL and will have attempted to enter one of the two types of error-handling exit routines that the user furnishes.

The application program can code two error-handling routines that VTAM attempts to invoke as a result of all physical, environmental, and logical errors. The routine that handles logical errors is called the LERAD routine, and the routine that handles other errors detected by VTAM is called the SYNAD routine.

Should an error occur during an RPL-based operation that is specified as synchronous, the LERAD or SYNAD routine is invoked as part of the processing. Should an error occur in conjunction with an operation that is specified as asynchronous, the LERAD or SYNAD exit routine will be entered at either of two times: as a result of the request being rejected or if the request is accepted, as the result of a CHECK being issued.

When the LERAD or SYNAD routine receives control, it is provided in register 0 and in the RPL with information regarding the specific cause of the error.

The VTAM application program in the main part of the program or in a LERAD or SYNAD exit routine can associate a class of completion information with a particular action. VTAM organizes its setting of register 0 and the RTNCD field of the RPL into these completion categories:

- Extraordinary completion: This requires further analysis of the RPL to determine the course of action required.
- Retriable completion: This indicates the request can be reissued. The EXECRPL macro instruction can be used.
- Damage completion: This indicates that, in addition to reissuing a request, some input or output data may have to be recovered or corrected.
- Environment error completion: This indicates that the program should call for external intervention.
- User logic error completion: This indicates that the program status should be saved, perhaps by requesting a dump. Depending on the seriousness of the error, the program can continue or terminate.

In some cases, simple determination that the return code (in register 0 and the RTNCD field of the RPL) specifies one of these completion categories can determine the course of action. In other cases, additional analysis of the RPL and other information (such as program flags) may be required to determine action.

LERAD Processing: Since most logical errors result from flaws in the design of an application program, the handling of logical errors is most important during the development and debugging of the application program. The handling of the error may consist of little more than recording the attributes of the request that failed, requesting a dump, and causing an abnormal termination.

SYNAD Processing: Physical and environmental errors usually require more extensive treatment. The application program should, during program execution, determine the nature of the error, assess the extent of the problem, and attempt remedial action. If the application program determines that the error occurred because incoming data exceeded the capacity of the NCP buffers, for example, the application program could inform the logical unit that the data should be resent in two transmissions. If the problem is a recurring hardware check for the logical unit, however, the application program may have to take whatever action is required to continue without the logical unit.

The error-handling routine can set register 0 or register 15 and return through VTAM to the instruction following the synchronous request or CHECK macro instruction. If the exit routine handled the situation successfully, register 15 and, in some cases, register 0 can be set to zero to indicate that the requested operation completed normally. The main part of the program continues normally, unaware that an error or special condition occurred and that the LERAD or SYNAD exit routine was entered. If, however, the exit routine was not able to complete successfully, it sets a value in one or both registers 15 and 0 before returning, so that the main program can take any action that the LERAD or SYNAD exit routine did not take.

Note: *The LERAD and SYNAD routines must be coded to be reenterable unless all RPL-based requests are in the main program or all RPL-based requests are in asynchronous exit routines. Unlike other exit routines, the LERAD and SYNAD exit routines can be interrupted in progress and reentered as the result of RPL-based requests within themselves or in other parts of the program.*

Opening the Application Program

In other telecommunications access methods such as BTAM, the application program first associates itself with the access method and with the line groups needed to support communication. The VTAM application program also must issue an OPEN macro instruction to establish an association with the access method.

A VTAM application program, like a BTAM program, typically begins with an OPEN macro instruction and ends with a CLOSE macro instruction. That is, the association between the application program and VTAM normally lasts for the duration of the application program's execution.

VTAM's OPEN, however, does not associate the application program with any logical units (the VTAM application program communicates directly with logical units and is not aware of line groups). Connection associates logical units with the application program.

The application program also uses the OPEN macro instruction to supply VTAM with the addresses of its error-handling routines (LERAD and SYNAD) and its event-driven exit routines (such as LOGON, LOSTERM, and RELREQ).

Connection

An application program must issue an OPNDST macro instruction to establish connection with a logical unit before communication with that logical unit can begin.

The OPNDST (open destination) macro instruction causes VTAM to "allocate" the logical unit to the application program. OPNDST initializes VTAM's and the application's control blocks to indicate that the logical unit and the application are connected. OPNDST also ensures that an active path exists between the two nodes before connecting them. Unlike the effect of an OPEN macro instruction, the effect of an OPNDST often does not last for the duration of the application program's execution. Because of VTAM's terminal-sharing facility, connections to the network's logical units can be made, broken, and remade innumerable times throughout the duration of the application program.

An application program can establish connection either by *accepting* the logical unit or by *acquiring* the logical unit.

Acceptance

When an application program accepts a logical unit, it does so because a logon request was received for the logical unit. A logical unit cannot be accepted unless (or until) a logon request has been issued for it. Although there are several possible sources of the logon request—the network operator, another application program, the logical unit itself or automatically as a result of VTAM definition—the request usually originates *outside* of

the application program. (Logon requests can also be generated *within* the application program; however such logon requests are essentially a form of acquisition, and are discussed in "Acquisition" later in this chapter.)

Acceptance is suitable for applications that do not require access to a specific logical unit or specific set of logical units in order to function, but instead are designed to service various logical units that require access to the application. If the installation, for example, wants the logical units themselves to designate which application they wish to use, the installation could allow each logical unit to initiate logon requests so that the application could accept the logical unit.

Note: *When VTAM notifies an application program of a logon request, the logical unit and its logon request is queued to the application. As long as the logical unit is queued to the application program, it is not available for connection to other applications; it is available for connection only to the application to which it is queued. The application program and its queued logical unit are unable to communicate with each other until the connection is completed by the application's acceptance of the logical unit. Because a queued logical unit is effectively eliminated from the system until accepted or disconnected by the application, the installation should ensure that application programs avoid leaving logical units on this queue any longer than necessary.*

Accepting Logical Units with an Exit Routine: The application program can maintain a LOGON exit routine which VTAM schedules whenever a logon request for the application program is made. VTAM provides the exit routine with the identity of the logical unit associated with the logon request. The application program can either accept the logical unit (using an OPNDST macro instruction) or reject it (using a CLSDST macro instruction).

The application program does not have to use an exit routine in order to determine when a logon request has been made. The application program can instead issue a connection request that VTAM completes as soon as a logon request has been made. Although this method is simpler to use than an exit routine, the application program does not have the opportunity to decline the logon request prior to actually establishing connection with the logical unit. (A logon request is declined by issuing a CLSDST macro instruction.)

Preventing Logon Requests: Logon requests cannot be directed to an application program until the application program uses the SETLOGON macro instruction, to notify VTAM that it is ready to accept them. At this point, any logon requests that are directed at the application program are queued until the application program either accepts or rejects them. Any time during its execution, the application program can notify VTAM that it is no longer accepting logon requests.

Types of Acceptance: The application program can issue a connection request to accept a *specific* logical unit, or to accept *any* logical unit for which a logon request has been issued.

To accept a specific logical unit, the application program must tell VTAM the identity of the logical unit; connection is not made until a logon request has been issued for that particular logical unit. This is the type of acceptance that an application program uses in the LOGON exit routine. Outside of the exit routine, the application program can accept a logon request from any logical unit in the network. After connection is established, VTAM passes the identity of the logical unit to the application program.

Acquisition

When the initiative for connection originates from the application program, the application program establishes connection by acquiring the logical unit. No logon request need exist; if the logical unit is active and not being used by another application program, the logical unit is connected (or queued for connection if the application is simulating a logon request to itself on behalf of the logical unit). The installation must authorize the application program's use of acquisition.

Types of Acquisition: Some or all of a set of active logical units can be acquired at once. As many as are available are connected. This type of connection (called the CONALL option) can be used when the application program is willing to proceed with as many active logical units as are available (that is, not already connected to another program). VTAM provides information so that the program can determine which logical units were connected. (If there is a possibility that one or more logical units of a group to be acquired may not be active, each logical unit will have to be acquired individually; the list method will not be successful.)

Another type of acquisition allows the application program to acquire any *one* logical unit of a specified set. The first available logical unit of the set is connected. This type of acquisition (called the CONANY option) is useful for applications that require a logical unit, but for which one logical unit is as good as another.

The application program specifies the logical unit set by building a series of control blocks (NIBs) each containing the installation-supplied name of the logical unit. The set can be limited to one logical unit.

A third type of acquisition can be used by application programs that both accept and acquire connections or which wish to ask the current owner of a logical unit (through its RELREQ exit routine) to release it. An application program can itself generate a logon request for a logical unit, and let the part of the program that accepts logon requests establish connection with the logical unit. The application program can generate logon requests for an entire set of logical units, or for any one of a set of logical units.

The use of such logon requests, called simulated logon requests, is a form of acquisition since the initiative for connection lies with the application program. Like the other forms of acquisition, its use must be authorized by the installation. Simulated logon requests are generated with the SIMLOGON macro instruction.

Acquiring Connected Logical Units: One application program cannot take another application program's logical unit from it. If an application program attempts to acquire a logical unit that is already connected, no reconnection is possible until the current owner of that logical unit disconnects it. VTAM provides a means by which the owning application program can be notified that another application program is requesting one of its logical units.

The requesting application program can indicate whether its attempt to reacquire a connected logical unit should or should not cause the owning application program to be notified. The requesting application program should request this notification when it needs the logical unit regardless of its connection status. Notification should not be indicated when the requesting application program only needs the logical unit if it is unconnected.

The owning application program also controls whether it can be notified when another application program issues a connection request to acquire one of its logical units. Notification can therefore only occur when both application programs so indicate.

VTAM notifies the owning application program by scheduling the RELREQ exit routine which the application program maintains for this purpose.

The RELREQ exit routine is provided with the identity of the contested logical unit. The application program can elect to disconnect the logical unit immediately, disconnect it later, or ignore the request entirely. If the logical unit is disconnected, the previous owner can immediately attempt to re-acquire the logical unit from the new owner (using a queued connection request as described below) so that the logical unit will be returned when it is no longer being used. When the logical unit is disconnected, it is reconnected to the acquiring application program that has waited the longest. This may not necessarily be the application program that was the previous owner of the logical unit.

By controlling which application programs release contested logical units and which do not, the installation can cause some application programs to be able to obtain and keep logical units more readily than other application programs. Or, the installation could establish a policy that all application programs release sharable logical units that are not being used; this would make the logical units available more frequently.

Establishing Logon Modes (Sets of Session Parameters)

An application program establishes connection by an OPNDST macro instruction and specifies a set of *session parameters* (rules that an application program and a logical unit agree to follow when commmunicating). Some of the parameters specified by these session parameters are: message/response conventions, acceptable types of chaining, error-recovery responsibility, and bracket usage. Some of these parameters are discussed later in this chapter. *System Network Architecture Format and Protocol Reference Manual*, GA27-3112 discusses all session parameters available to VTAM application programs and their use. To specify a set of session parameters, an application program can:

- Supply a logon mode name to specify a predefined set of session parameters
- Supply the name of an area in the application program's storage that contains a set of session parameters

VTAM uses these values to construct the SNA command (called a Bind command) that completes the connection requested by the OPNDST by causing the application and the logical unit to be in an SNA session.

An application program requests a predefined set of session parameters by specifying a logon mode name in the node initialization block (NIB) associated with the OPNDST macro instruction. The logon mode name can be the value passed to the application program as a result of a connection request from a terminal (in a character-coded LOGON command or field-formatted Initiate-Self command), as the result of a VTAM-initiated (automatic logon) request, as the result of a network-operator-initiated request, or the logon mode name can be a value supplied by the application program. VTAM searches the logon mode table associated with the logical unit to obtain the predefined set of session parameters for that logon mode.

An application program can dynamically define a set of session parameters by setting the contents of a specified area of storage to indicate the individual session parameters desired. IBM supplies a DSECT, ISTDBIND, for this purpose.

When an SNA logical unit makes a logon request, it specifies a logon mode name. VTAM processes the logon request and notifies the requested program of the pending logon request. Included in the information passed to the application program is the logon mode requested by the logical unit (if no logon mode is specified, blanks are used), and the name of the logical unit initiating the request. If the application program wishes to examine the session parameters associated with the logon mode name, it issues an

INQUIRE macro instruction specifying OPTCD=SESSPARM, which causes VTAM to search the logon mode table associated with the logical unit for that logon mode name. If the application program specifies zero in the LOGMODE field of the NIB associated with the INQUIRE instruction, a search is made for the logon mode name specified on the first connection request queued to the application program. (See "Queuing Connection Requests", below, for a description of the queuing process.) If a logon mode (as specified by the queued connection request or by a name in the LOGMODE field of the NIB) is all blanks, VTAM uses the first entry on the logon mode table for the logical unit. Having found the session parameters associated with the logon mode, VTAM moves them to the area specified by the INQUIRE macro instruction (called the bind area). The application program then examines and changes any fields in the bind area using the IBM-supplied DSECT.

When the application program issues an OPNDST macro instruction to complete the connection with the logical unit, it indicates the session parameters by specifying the desired logon mode name in the LOGMODE field of the NIB or a bind area name in the BNDAREA field of the NIB. (Whichever field is specified, the other should be set to zero. Figure 5-10 shows the action taken by OPNDST when each combination is specified.)

Queuing Connection Requests

Application program requests for connection always fail if the logical unit is inactive. If the logical unit is active but unavailable, however, the application program designates whether its connection request should fail or whether the request should remain pending (queued) until the logical unit does become available.

Although the definition of an "available" logical unit differs between acquired and accepted connections, the option of queuing or not queuing the communication request applies to both. When an application program attempts to *acquire* an active logical unit, the logical unit is available if it is not connected (or queued for connection as the result of a logon request) to another application program. When an application program attempts to *accept* a logical unit, the logical unit is available if a logon request for it has been directed at the application program. (Note the distinction between queuing an application program's request for connection—described here—and queuing a logical unit to an application program as the result of a logon request—as described in "Acceptance" earlier in this chapter.)

Figure 5-11 lists the effects of queuing a connection request on the various types of connection requests.

If the NIB specifies:		An OPNDST ACCEPT will use the session parameters specified by the:	An OPNDST ACQUIRE will use the session parameters specified by the:
LOGMODE=	BNDAREA=		
0	0	Pending logon mode	Default logon mode
0	area	Bind area	Bind area
logon mode	0	Logon mode	Logon mode
logon mode	area	Bind area	Bind area

Figure 5-10. OPNDST Action Taken Depending on Whether a Logon Mode Name or a Bind Area (or Both) Is Specified

Type of Connection Request	Meaning When Connection Is To Be Queued	Meaning When Connections Not To Be Queued
Accept		
a specific logical unit	Connect the logical unit if a logon request has been received from it; otherwise connect the logical unit when a logon request is received from it	Connect the logical unit (only if a logon request has already been received from it)
any logical unit	Connect any logical unit from which a logon request has been received (if more than one has been received, connect the logical unit that has waited the longest); otherwise wait for a logon request and then connect the logical unit	Connect any logical unit from which a logon request has already been received (if more than one has been received, connect the logical unit that has waited the longest); otherwise connect none
Acquire		
as many as are available in a set (CONALL)	Schedule a LOGON exit routine as each logical unit becomes available *	Connect all the logical units that are available
a set of one (CONANY)	Schedule a LOGON exit routine when the logical unit becomes available *	Connect the logical unit if it is available
any one of a set (CONANY)	Schedule a LOGON exit routine for the first logical unit in the set that becomes available *	Connect the first logical unit in the set that is available; otherwise connect none
	* Request queueing is available for SIMLOGON only. It is not available for OPNDST with OPTCD=ACQUIRE.	

Figure 5-11. Queued and Unqueued Connection Requests

Disconnection

To disconnect a logical unit, an application program can *release* the logical unit or it can *pass* the logical unit to another application program. The logical unit is released by disconnecting it without regard to which application program (if any) is to receive the logical unit. The logical unit is passed by disconnecting it and designating which application program is to receive the logical unit. Passing must be authorized by the installation.

Passing and releasing is accomplished by using the PASS and RELEASE options of a CLSDST macro instruction.

If the logical unit is released, VTAM connects the logical unit to any application program that has attempted to acquire the logical unit (and has indicated that its connection request should be queued). If more than one application program has attempted to acquire the logical unit, VTAM connects the logical unit to the application program that first issued the connection request. If there are no queued requests to acquire the logical unit, VTAM generates an automatic logon request for the logical unit; if no automatic logon request has been specified by the installation, the logical unit remains unconnected.

When a logical unit is passed, VTAM disconnects the logical unit, generates a logon request, and directs the logon request to the designated application program. The logical unit is not reconnected until the receiving application program accepts the logon request.

A logical unit should be passed only when it is imperative that it be connected to a specific application program and to none other. For example, an installation might

maintain several application programs that each require the same information from the logical unit before any of them can be used. Although each application program could conduct its own interrogation, it might be simpler for the installation to maintain a single application program that converses with the logical unit to obtain the initial information and then passes the logical unit to the appropriate destination.

When the application program passes a logical unit, it can also pass data to the receiving application program. In the example above, the application program might pass along the results of the preliminary conversation.

Communication

Communicating with logical units requires an understanding of concepts related to:

- What is communicated between a VTAM application program and a logical unit (messages and responses)
- VTAM facilities for sending and receiving messages and responses
- SNA-defined protocols that may be used to control the exchange of messages and responses

Messages and Responses

Communication between a VTAM application program and a logical unit consists of an exchange of *messages* and *responses*.

A message consists of data and control information or sometimes control information alone. For example, a message can consist of data (perhaps an answer to an inquiry previously forwarded) furnished by the VTAM application program in a designated data area, and other information, such as the kind of acknowledgment wanted, that is specified symbolically to VTAM when it requests VTAM to send the message. Or, in some cases, a message can contain no data, and be intended to control the further exchange of data with a logical unit. For example, the program could specify that VTAM send a message on its behalf to the logical unit indicating that the logical unit should stop sending until released to send by the VTAM application program.

A response indicates whether or not a message arrived successfully. A response is positive or negative. Figure 5-12 illustrates messages and responses.

The length of the data in a message is defined by the VTAM application program and the logical unit.

A message or a response is sent by the VTAM application program by issuing a SEND macro instruction. For a message, the SEND specifies a data area (if there is data) and control information. For a response, the SEND specifies the nature of the response (positive or negative) and possibly control information for a response. A message or a response is received by a VTAM application by issuing a RECEIVE macro instruction. (In some cases, input can also be received as the result of VTAM scheduling a special-purpose exit routine; in this case, the input is present when the exit routine is entered and a RECEIVE does not have to be issued.)

When VTAM receives a SEND macro instruction request, it takes the data from the VTAM application program into its own buffers and, together with the control information specified in the SEND macro instruction, formats the message into a structure defined by systems network architecture (SNA).

Message Flow: Normal and Expedited

The VTAM application program can send and receive messages in two different message streams or flows. *Normal-flow* messages are messages that contain data and certain control commands; *expedited-flow* messages are messages that contain control commands

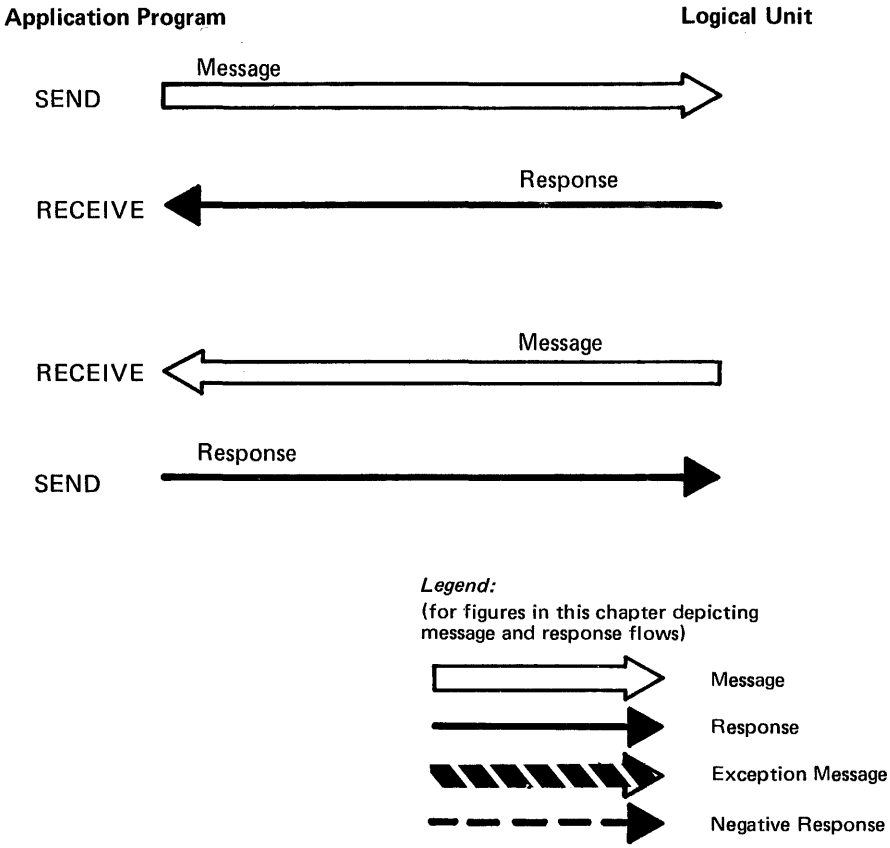


Figure 5-12. Exchanging Messages and Responses

that require that they be handled ahead of the normal-flow messages. For example, if a VTAM application program wants to signal to a logical unit to shut down its operations for the day, it can send a Shutdown command message; this message will be expedited over other messages from the VTAM application program that may already be scheduled by VTAM for sending to the logical unit.

Levels of Message Control: Session Control and Data Flow Control

Ordinarily, a VTAM application program and a logical unit will be in a level of message control in which data and related control information and responses are exchanged. Using special commands, however, a VTAM application program can stop and restart the exchange of data messages and related control information and responses. While this flow is stopped, the VTAM application program and the logical unit, using special *session control* commands, can exchange information about *sequence numbers* with which each data message is associated. (Each normal-flow message is assigned a sequence number by VTAM; this number is one greater than the sequence number assigned the previous normal-flow message that the VTAM application program requested be sent to the logical unit.)

The SESSIONC macro instruction is used to start, stop, and exchange information about sequence numbers. Figure 5-13 shows an example of SESSIONC used to stop and restart the data flow.

Responses: Requesting a Response

When the VTAM application program or a logical unit sends a message, it specifies the conditions under which it wants a response to that message returned. The message sender can request a definite response (it wants either a positive or a negative response as

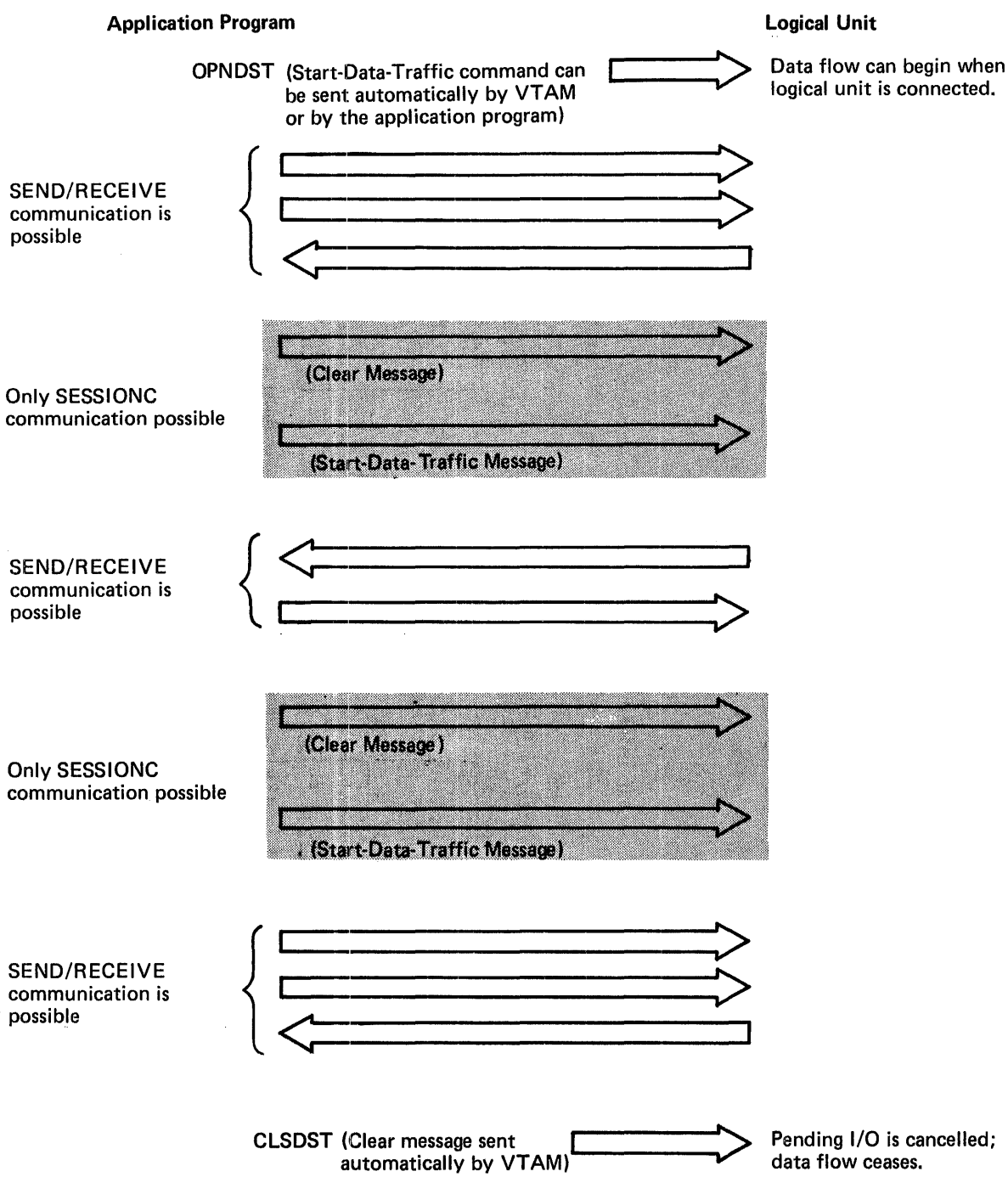


Figure 5-13. An Example of Start-Data-Traffic and Clear Messages

appropriate) or an exception response (it wants a response only if a negative response is appropriate). The receiver of the message must respond as appropriate. The VTAM application program requests a definite response, an exception response, or no response by specifying a parameter in the RESPOND (either FME or RRN or both and NEX for a definite response; EX for an exception response; and NFME, NRRN, and NEX or NFME, NRRN, and EX for no response). The VTAM application program determines the type of response required after it has received a message by examining the RESPOND field of the RPL.

Figure 5-14 shows a case where a definite response is requested and a case where an exception response is requested.

Requesting definite responses, rather than just exception responses, results in greater control over error conditions and provides an opportunity for quicker error recovery--but requires more programming and more traffic in the network. A request for only exception responses can be used within a group of related output messages if the entire group is to be discarded when an exception condition occurs. Requesting no responses of either sort is appropriate when there is no concern for error recovery for that transmission.

The application program can specify the same options in the messages that it sends to a logical unit. The logical unit can examine the message received from the application program and determine whether a positive or a negative response (or no response) should be returned to the application program.

Responses: Positive or Negative

When a definite response has been returned to a message, the response receiver determines whether the response was positive or negative. If only negative responses were requested, the receipt of any response would indicate a negative response.

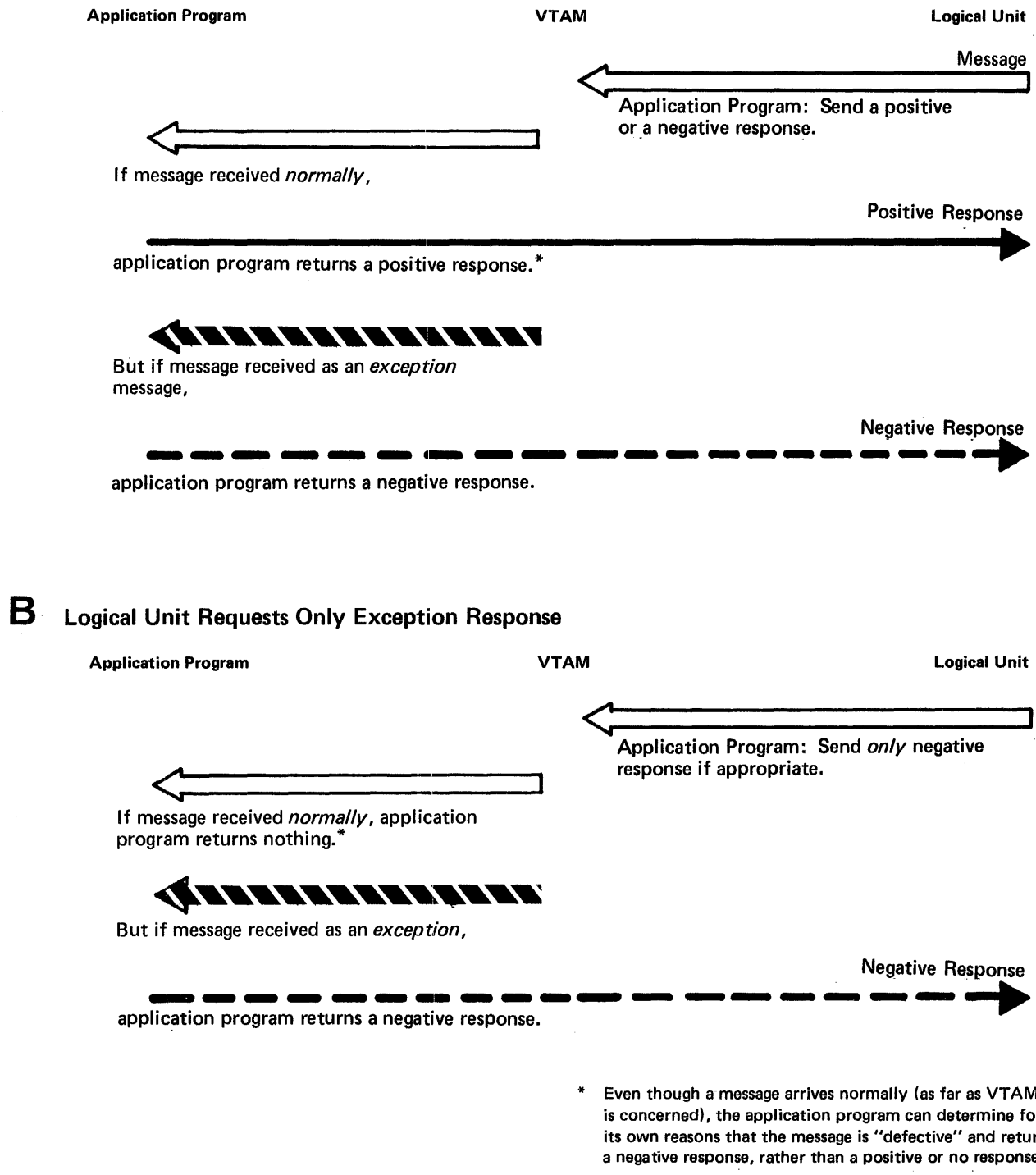
A VTAM application program receives a response either as the result of a RECEIVE macro instruction that allows a response or as the result of a RESP exit routine being scheduled. The type of response received can be determined by examining the RESPOND field of the RPL associated with the RECEIVE or provided by VTAM in the case of the RESP exit routine.

The logical unit or application program can send a negative response to a message for reasons other than whether or not it was received successfully. For example, a negative response might indicate that the data in the message was not in a prescribed form.

Responses: Definite Response 1 and 2

Every response, positive or negative, is designated by its sender as a definite response 1 (FME) or definite response 2 (RRN). Although some applications may not need to distinguish between two different kinds of positive response, other applications may; the distinction may be made for any purpose understood by the installation.

If a distinction is made between definitie response 1 and 2, it works like this: The sender of a message requests a response be returned; it may specify that either definite response 1, definite response 2, or both be returned. The receiver of the message, in returning either a positive or negative response as appropriate, also indicates the corresponding response type (definite response 1, definite response 2, or both) that was requested in the message. The receiver must send back the same type or types of definite response that were requested.



* Even though a message arrives normally (as far as VTAM is concerned), the application program can determine for its own reasons that the message is "defective" and return a negative response, rather than a positive or no response.

Figure 5-14. An Example of a Logical Unit Requesting (A) A Definite Response and (B) Only an Exception Response

Sequencing and Chaining

VTAM assigns a sequence number to each data message sent to a logical unit. The numbering begins with the first message sent after connection. The number is increased by one for each subsequent message. This process continues until the logical unit is disconnected, unless interrupted earlier by the application program.

Should a message arrive out of sequence (that is, its sequence number is not one greater than that of the last record received), VTAM considers this a transmission error and replaces the missing message with an exception message.

When a response is sent (either positive or negative), the sender assigns to it the sequence number of the message being responded to. This provides the sender of a message with a means of matching the response with its message. For example, an application program might send a group of messages, with each message indicating that only negative responses should be returned. Should a negative response be returned, the application program could use the sequence number to determine where in the group the error occurred. Sequence numbers are also useful for logical units that log each received or sent message.

Application programs (or logical units) can group any number of messages into a set called a message *chain.* The sender can indicate which part of a chain is being transmitted–the first message of the chain, the last message of the chain, neither (the message is somewhere in the middle) or both (the message is the sole element of the chain).

The sender of a chain can at any time send a *cancel command* to the receiver (the sender might send this indicator because a negative response has been returned). The receiver can interpret this as an indication that all received messages of the current chain should be discarded.

The actual unit of work that the chain represents is determined entirely by the application program and the logical unit.

Figure 5-15 illustrates a possible use of chaining. In this example, a logical unit has submitted an inquiry to the application program. The application program can obtain various pieces of information from data files and send them to the logical unit as each becomes available. By chaining the output requests, the application program has a convenient way of telling the logical unit whether any given piece of data represents the beginning, middle, or end of a reply to an inquiry and of checking all the reply before displaying any part of it.

Send Messages: Scheduled or Responded Output

Messages can be sent to a logical unit with one of two options:

- The application program can indicate that as soon as the message has been scheduled for transmission and the output data area is free, VTAM is to consider the output operation completed (by returning control, posting an ECB, or scheduling an RPL exit routine). This is called *scheduled* output and is illustrated in Figure 5-16.
- The application program can indicate that VTAM is not to consider the operation complete until the message has been received by the logical unit and a response has been returned. This is called *responded* output and is illustrated in Figure 5-17.

These options define the completion of an output operation, and should not be confused with synchronous and asynchronous request handling, which indicate the action to be taken when the completion occurs.

Responded output requires that the output data area and RPL not be reused until a response has been received. If the response indicates that an error occurred, the data is

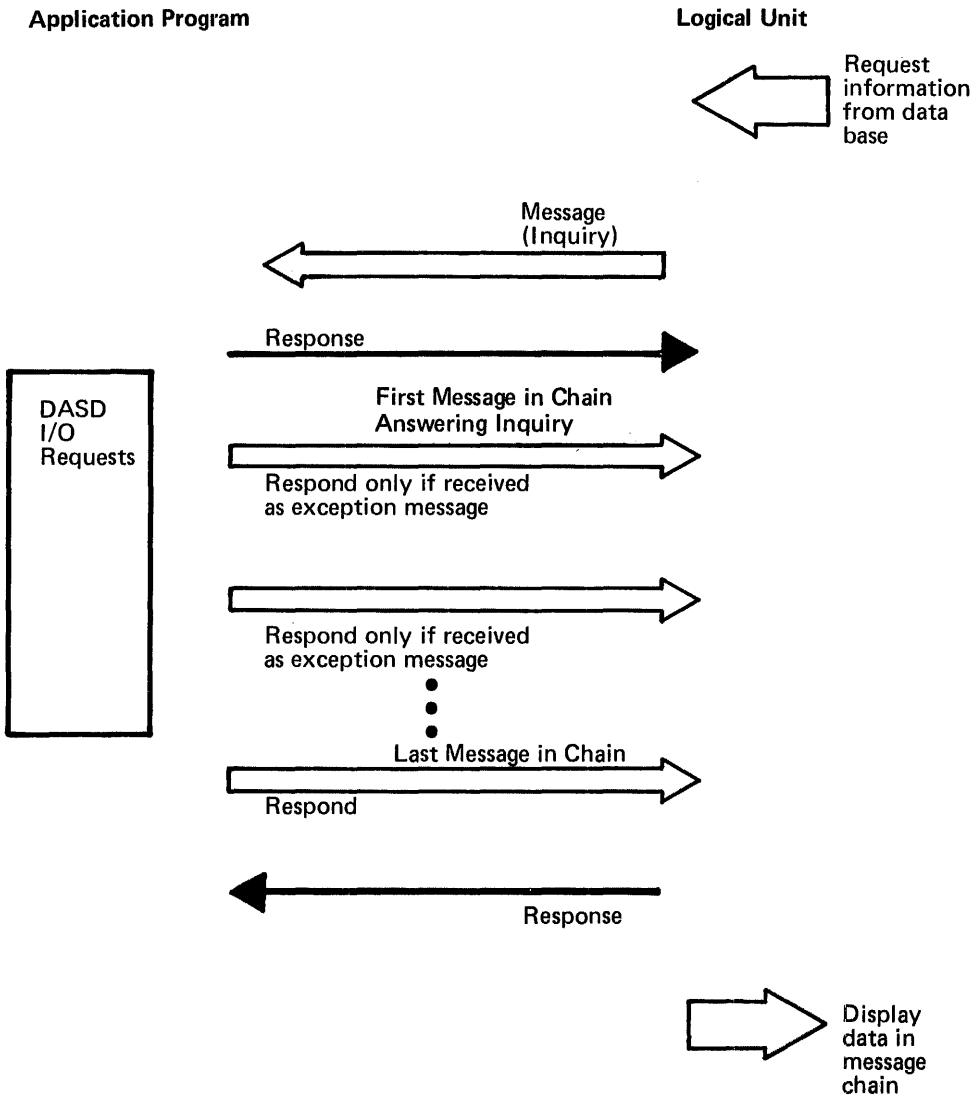


Figure 5-15. An Example of Message Chaining

still available for retransmission. Scheduled output allows the application program to send a series of messages that all use the same I/O area and RPL.

With responded output, completion status information is returned when the output request is completed. But with scheduled output, the output request is completed before any completion status information is available. To determine how the output operation completed, the application program must issue an input request to obtain the completion status information. This is why the application program in Figure 5-16 issues three input requests in addition to the three output requests.

If an error occurs during the transmission of the message, the receiver is passed a substitute message that indicates the error condition. This message is called an *exception message.* The node transmitting the message becomes aware of the error condition when the other node returns a *negative response.*

When the logical unit sends a message to the application program, the logical unit indicates in that message the type of response it expects. The terminal can tell the application program to:

- Send a response regardless of whether the message arrives normally or not; that is, if an exception message is received, send back a negative response. If the message arrives normally, send back a positive response. Part A of Figure 5-14 is an example of responding to both normal and exception messages.
- Send only a negative response. That is, if the message arrives normally, send no response. If an exception message arrives, send a negative response. Part B of Figure 5-14 is an example of responding only to exception messages.

Receiving Input

Data, responses and data flow control information can be received by the application program separately or with one RECEIVE macro instruction. When the macro instruction is issued, any combination of the following types of input can satisfy the request (any combination can be specified):

- Normal-flow messages
- Expedited-flow messages
- Responses

Normal- and expedited-flow messages are terms used to group all messages into two distinct types of input. Two types of input results are necessary due to these characteristics of data transmission:

- If messages are sent at a faster rate than the receiver receives them, the messages are queued for the receiver.

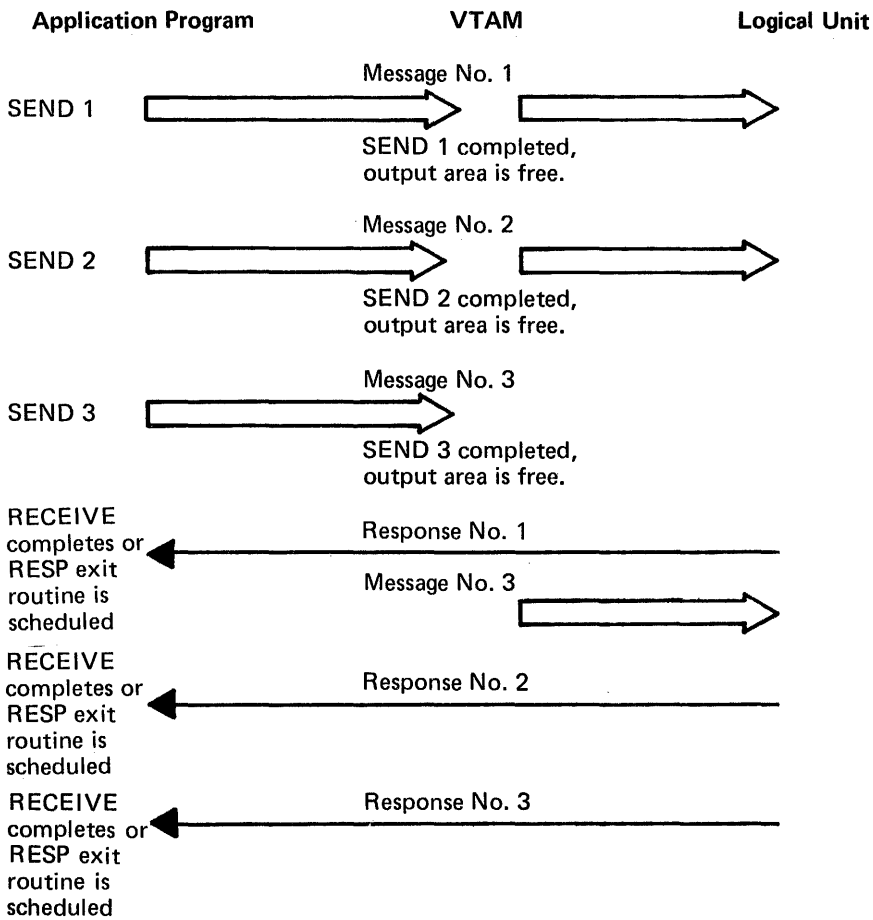


Figure 5-16. Scheduled Output

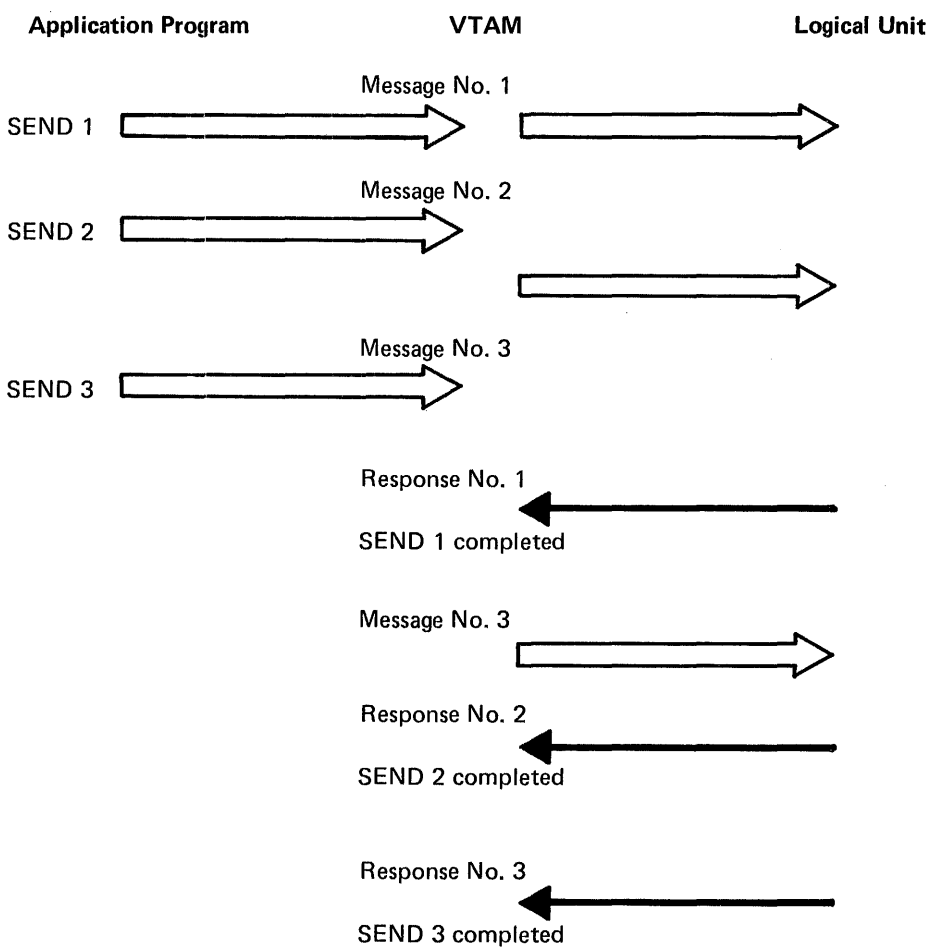


Figure 5-17. Responded Output

- Some messages should not be queued with the other messages but should be available to the receiver separately and immediately.

Thus, VTAM provides two "priority levels" for messages. Data messages are always treated as normal-flow messages. Certain flow-control information is treated the same way. One example is the Quiesce Completed message, which must keep its place in the queue of data messages; if it were to be received prematurely, the bypassed data messages might be lost.

Other data-control information must be made more immediately available to the receiver and is therefore made available to the receiver as expedited-flow messages. One example of an expedited-flow message is the Quiesce at End of Chain (QEC) command. This type of message is not meant to stay within a queue of data messages, waiting until the receiver eventually obtains it.

Figure 5-18 Shows the types of information exchanged between an application program and logical units.

See Appendix D for an indication of which messages are normal-flow and which are expedited-flow.

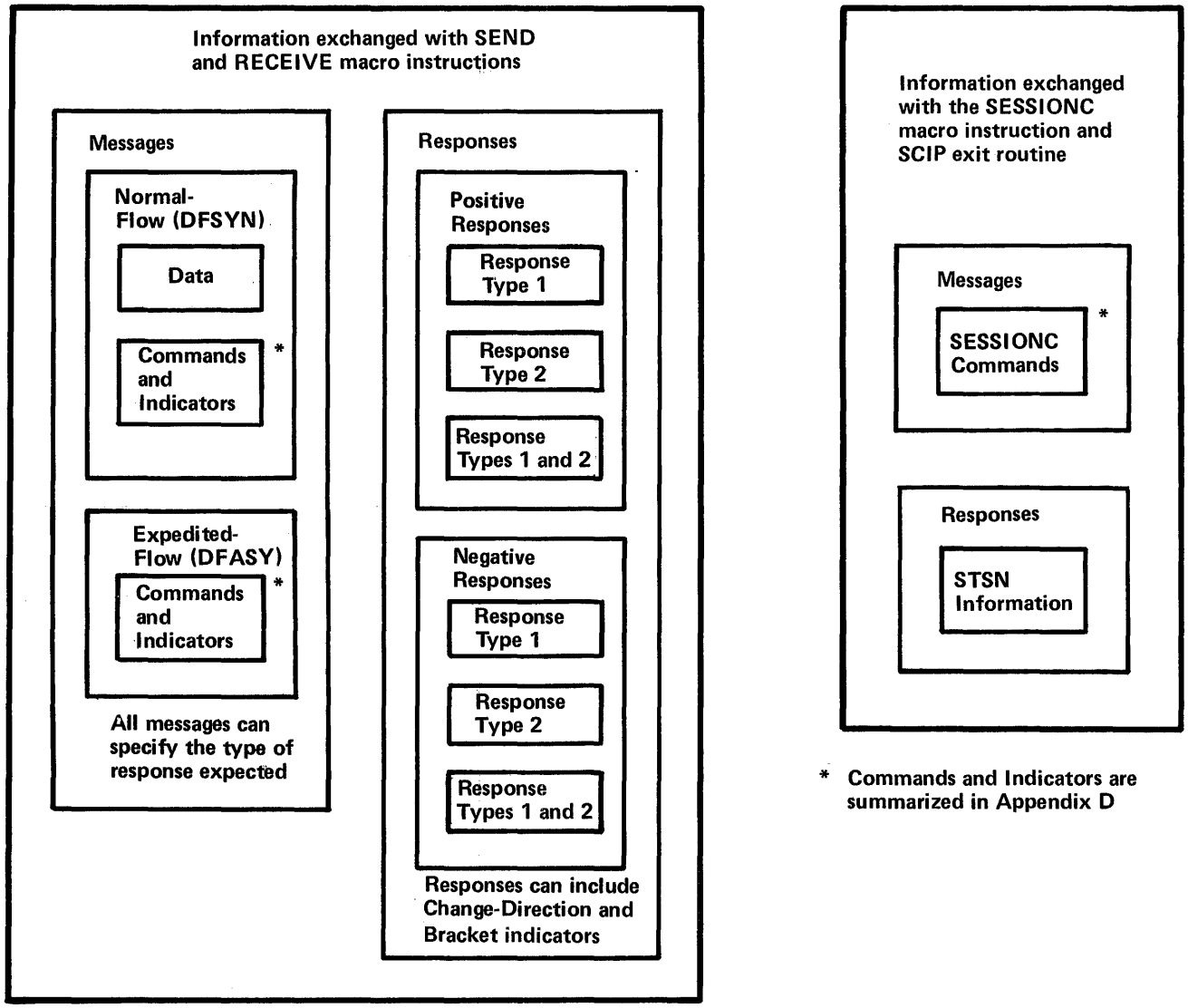


Figure 5-18. Types of Information Exchanged Between an Application Program and Logical Unit

The application program can maintain three types of exit routines that VTAM schedules whenever one of the following types of input become available:

- Expedited-flow messages (DFASY exit routine)
- Responses (RESP exit routine)
- Request-recovery (RQR) commands (SCIP exit routine)

Except for the SESSIONC command, the types of input that can cause the exit routine to be scheduled correspond to the type of input that causes a particular type of input request to be completed. Unless a SCIP routine is maintained for this purpose, the application program has no means of receiving an RQR command.

These exit routines operate in the same manner as those described in "Application Program Exit Routines," earlier in this chapter. One difference, however, is that the application program need not use one exit routine to handle a particular kind of input from *all* logical units. A given exit routine can service input from a limited set of logical units, or a separate exit routine can be maintained for each logical unit.

Specific-Mode and Any-Mode

To obtain data from its logical units, the application program can request data from a *specific* logical unit, or it can request data from *any* one of its connected logical units. The application program designates the desired mode–specific or any–with each RECEIVE macro instruction. These two modes are called, respectively, the *specific-mode* and the *any-mode*.

In general, an application program initially requests input from its logical units in the any-mode, and then communicates with each logical unit in the specific-mode until the transaction, inquiry, or conversation is completed. While specific communication proceeds with one logical unit, the application program keeps a RECEIVE macro instruction (issued in the any-mode) pending so that new transactions can be handled.

A RECEIVE issued in the any-mode is analogous to a BTAM READ Initial macro instruction except that (1) the concept of polling does not apply to logical units, and (2) the RECEIVE can be satisfied by any logical unit connected to the application program, rather than by just those attached to one line.

In the any-mode, the application program does not know the identity of the source logical unit until the data has been moved into its input area and the RECEIVE has been completed. Since the logical unit is initially unknown, the amount of incoming data may also be unknown. This means that the application program must either reserve an input area large enough to hold the largest possible amount of incoming data, or execute additional instructions to handle overlong data. On the other hand, the any-mode allows the application program to use just one input area for data from all of its logical units, rather than using a separate input area for each of its logical units.

With the specific-mode, the application program must specify the identity of the logical unit supplying the data. Since the identity of the source is known, the size of the input area is much more predictable than with the any-mode. But a logical unit may not supply data for some time, so the application program may have to contend with unused data areas.

Input data areas can be managed by a combination of specific-mode and any-mode. As an example, consider an application program that obtains an inquiry from any of its logical units, handles that inquiry with a series of SEND and RECEIVE macro instructions, and then obtains a new inquiry. Part of such a program is illustrated in Figure 5-19.

Continue-Any and Continue-Specific Modes

In the example of Figure 5-19, synchronous request handling for the I/O requests is assumed. The application program handles each inquiry serially, never accepting a new inquiry until it has completed the previous one. Although this procedure might be suitable for application programs processing short inquiries and few logical units, most applications would probably work more efficiently if the inquiries were handled in parallel.

An application program designed to handle more than one inquiry concurrently (Sample Program 2 at the end of this chapter is such a program) might use asynchronous request handling and issue new RECEIVE macro instructions in the any-mode before the previous inquiry has been completed. This, however, raises the possibility that both a RECEIVE for a specific logical unit and a RECEIVE for any logical unit (which includes the specific logical unit as well) might be awaiting data at the same time. Consequently, data for the subroutine's RECEIVE might instead be intercepted by the RECEIVE in the main program, which is meant only to receive new inquiries.

To eliminate this sort of problem, VTAM allows the application program to indicate when a RECEIVE macro instruction issued in the any-mode can receive data from a particular logical unit, and when a RECEIVE macro instruction issued in the

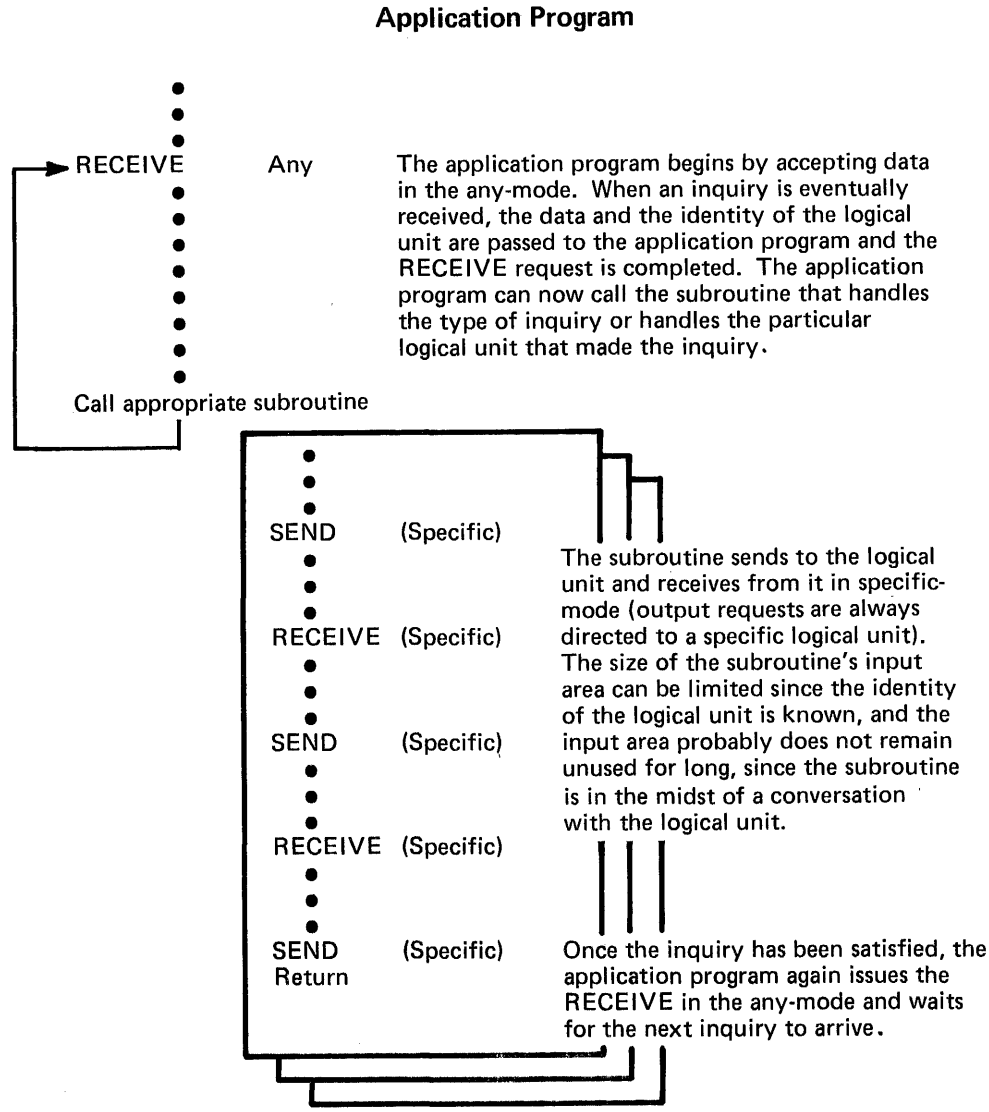


Figure 5-19. Using a Combination of Any-Mode and Specific-Mode to Obtain Data

specific-mode must receive the data. The former is called *continue-any mode*, and the latter is called the *continue-specific mode*. These modes are designated when an I/O request is issued, but do not become effective until the I/O operation is completed. Alternatively, a logical unit can, at connection, be designated as always in either continue-any or continue-specific mode. If the logical unit is always in one of these modes, the mode specified in the I/O request is disregarded. This facility makes it possible to treat logical units that are expected to always enter one-line input (always in continue-any mode) differently from those that may enter multiple-lines of input (only the first line received in continue-any mode).

Figure 5-20 illustrates how the various modes described above relate to one another.

Identifying Logical Units

Before an application program begins to communicate with a logical unit, it has the logical unit's installation-supplied name. This is an eight-byte symbolic name created for the logical unit during VTAM definition. When connection is established with the logical unit during program execution, the application program is also provided with a

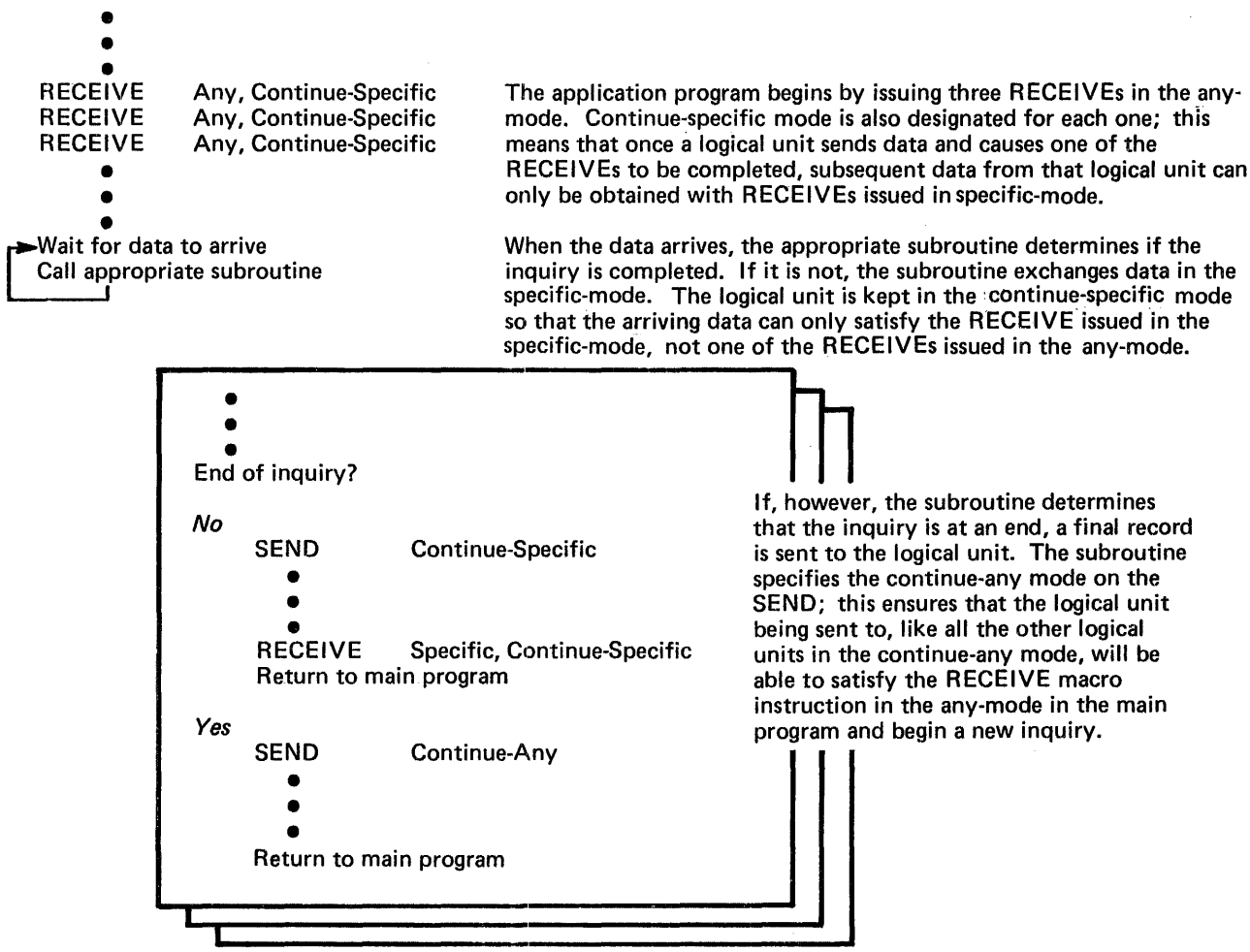


Figure 5-20. Using The Continue-Any and Continue-Specific Modes to Handle Concurrent Inquiries

VTAM-supplied network-oriented name (called a communications identifier, or CID) for that logical unit. The application program uses the CID for all I/O requests issued in the specific-mode.

When a RECEIVE macro instruction issued in the any-mode is completed, VTAM provides the identity of the logical unit that sent the data. Since the application program will probably communicate with the logical unit in the specific-mode, VTAM supplies the CID, rather than the symbolic name, to the application program. Should the identity be significant, the application program has three ways to relate the CID to the logical unit's symbolic (installation-supplied) name:

- The application program can ask VTAM to translate the CID into its symbolic name. This is done with an INQUIRE macro instruction.
- The application program can maintain a table of CIDs and their symbolic equivalents.
- When the application program establishes connection with the logical unit, the application program can initially assign to the logical unit a four-byte value that VTAM returns each time that logical unit's data satisifies a RECEIVE. The four-byte value can be anything the application program chooses to associate with the logical unit and is known as the user field. It could be used to identify the logical unit, or it could contain the address of a subroutine that is to handle that logical units's data.

Handling Overlong Input Data

When an application program issues a RECEIVE macro instruction, the length of the incoming data is often unpredictable. As noted above, this is particularly true of RECEIVE macro instructions issued in the any-mode. VTAM provides two ways of handling data that is too large for the input area:

- VTAM can discard the overlength data. This facility is called the TRUNC (truncate) option. This option would be useful in applications that must impose rigid size limitations on input data. For example, an inventory-control application might require the logical unit to supply an account number no longer than 10 bytes.
- VTAM can keep the data. VTAM fills the input area, saves the remainder, and completes the input request. Additional input requests must be issued to obtain the excess data. This facility is called the KEEP option.

The application program selects the appropriate option on an individual logical unit basis when the logical unit is connected and can override the option as required by a request.

Quiescing

VTAM provides a set of indicators and commands that the application program can use to request that logical unit stop sending normal-flow messages to the program. The VTAM application program can receive similar requests from a logical unit for the application to stop sending.

Quiescing can be used to occasionally stop the logical unit or application program from sending because of some temporary circumstance. For example, a logical unit may be receiving a long chain being sent from a VTAM application program. Elements received are stored in a buffer associated with the logical unit. When a certain point in the buffer is reached (a point that normally is not reached), the logical unit sends a *quiesce-at-end-of-chain* (QEC) *command.* The VTAM application program can interpret this command to mean "quiesce immediately." It can either end the chain at this point, allowing the logical unit to clear its buffer, or it can send back a quiesce-completed command, indicating that it is temporarily ceasing to send until the logical unit clears its buffer and sends back a *release-quiesce* (RELQ) *command.*

In another example of the use of quiescing, a terminal operator at a device to which a printout was being sent could notify the subsystem application program to interrupt the printout to enter an inquiry. The logical unit would send a QEC command, and the VTAM application program would temporarily hold sending and prepare to receive input from the logical unit.

Quiesce, Change-Direction, and Bracket Protocols

Messages can be sent to a logical unit whether or not the logical unit is at that moment sending messages to the application program. The nature of some applications, however, may prohibit such unrestricted exchange of data, or the application program may have been developed when no capability for unrestricted data exchange existed, and the installation does not wish to rewrite the program just to use this facility. For such application programs, three methods of communication are available that allow the application program and the logical unit to control each other's ability to send data. These three methods (which are essentially three sets of commands and indicators with "rules" about how to use them), are called *quiesce protocol, change-direction protocol,* and *bracket protocol.*

Quiesce Protocol: The application program informs the logical unit that it is to stop sending data when the logical unit has completed sending its current chain. It does so by sending a *Quiesce at End of Chain* (QEC) *command.* When the logical unit replies, by sending a *Quiesce Completed* (QC) *reply,* it must stop sending and prepare to receive. The logical unit cannot again send data until it receives a *Release Quiesce* (RELQ) *command* from the application program, as shown in Figure 5-21.

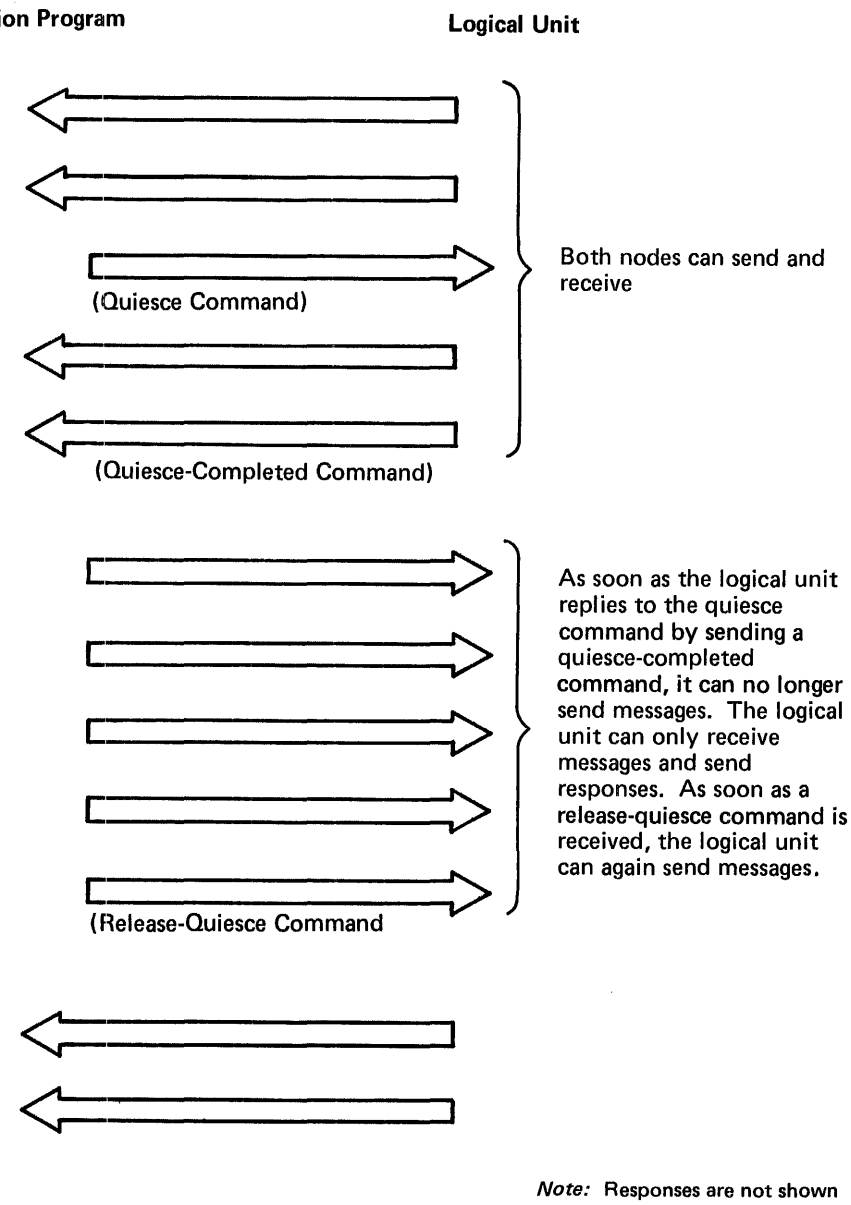


Figure 5-21. An Example of Quiesce Protocol

This convention is not enforced by VTAM; it is the user's responsibility to conform to the convention.

The application program is not necessarily the primary node (that is, the node entitled to "quiesce" the other node).

Change-Direction Protocol: The first node that sends data (following the application program's indication that data flow can begin) can continue to send data. When the first node is through sending data, it sends a *Change Direction Command indicator* to the other node. The other node sends data until it relinquishes its ability to send by returning another change-direction-command indicator. The nodes continue to alternate in this fashion, as shown in Figure 5-22.

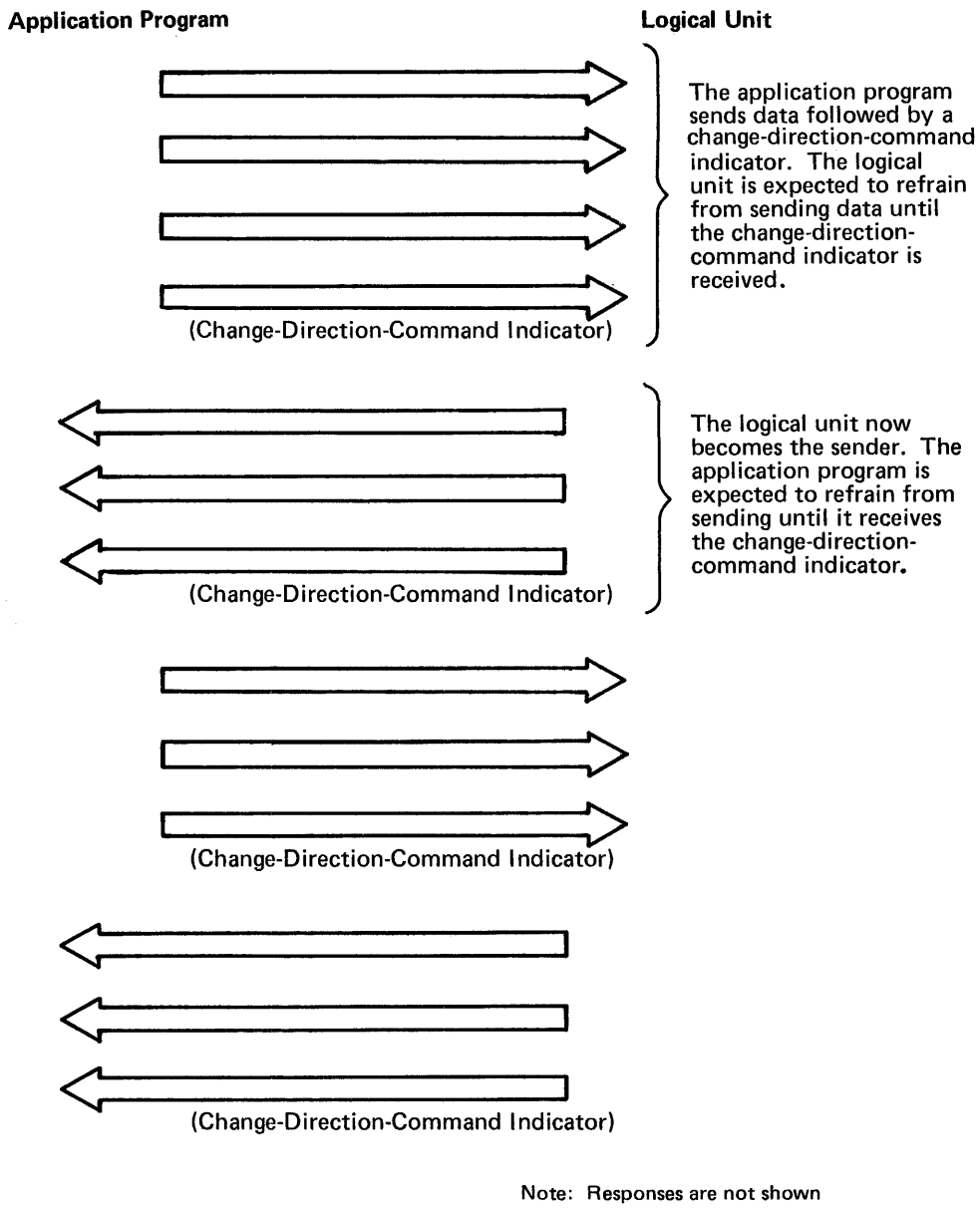


Figure 5-22. An Example of Change-Direction Protocol

While the receiver is awaiting the Change Direction Command indicator, it can transmit (as part of a response) a prompting indicator to the other node that in effect says "I would like that Change Direction Command indicator now." The prompting indicator, called a *Change Direction Request indicator* can be honored or it can be ignored.

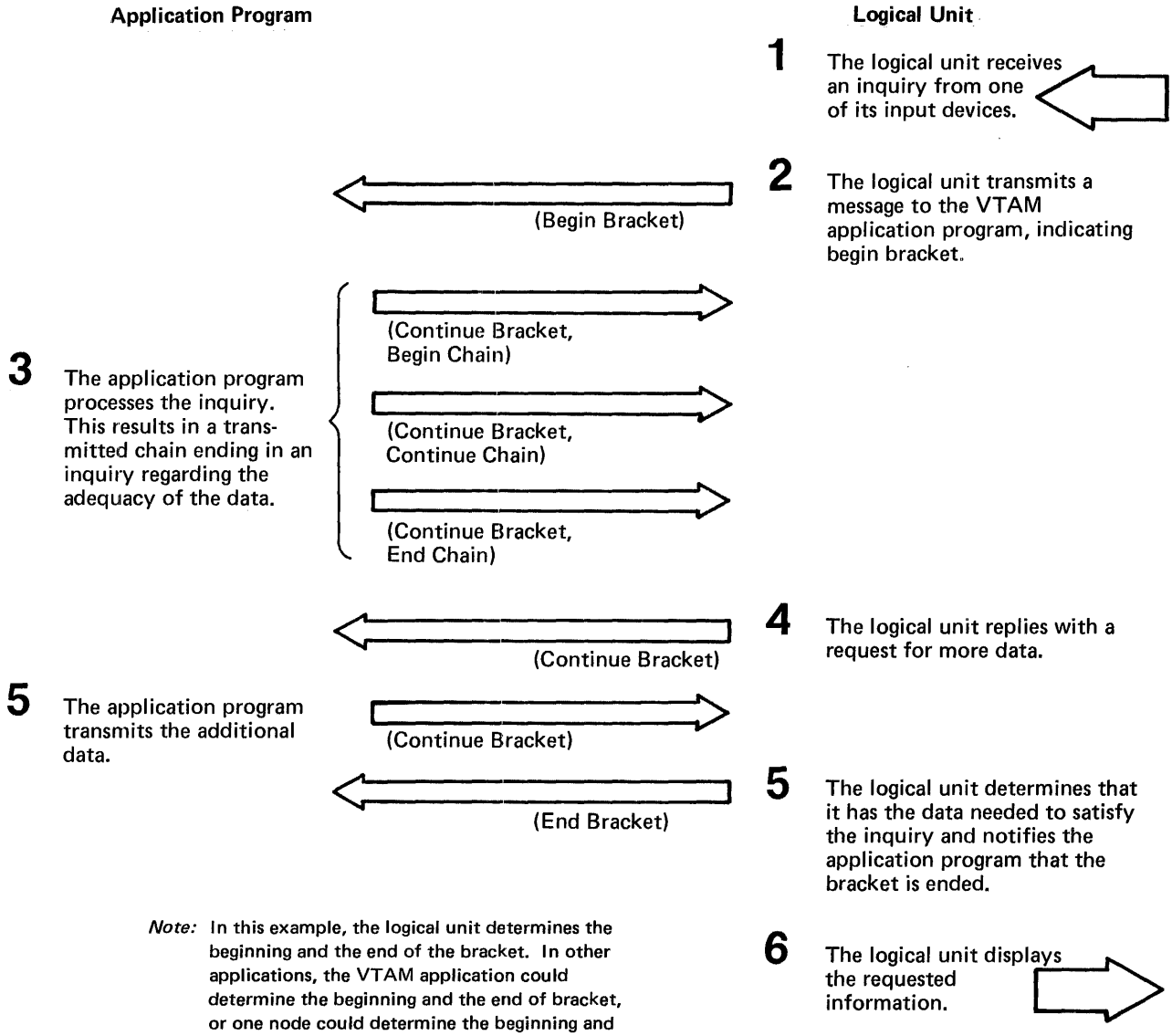
Change-direction is not enforced by VTAM. Should the node waiting for a change-direction indicator begin sending data anyway, VTAM does not prevent the transmission of the data. The successful use of this method of communication rests on the assumption that neither the application program nor the logical unit violates the "rules."

The node that is awaiting a change-direction indicator, like the node that is in a quiesced state, is free to send responses. Only the sending of messages is restricted.

Bracket Protocol: A *bracket* is any unit of work for which the application program and the logical unit have been programmed. Each bracket consists of input operations or output operations (or both) that do not necessarily follow a fixed pattern. Data-base inquiry and data-base update transactions are typical examples of brackets.

Bracket communication is used when one of the nodes cannot process a new bracket until the previous one has been completed. Nodes using this method of communication note on each transmitted message whether that message is the beginning or end of a bracket. These delimiters allow the receiving node to determine whether or not a new bracket can be started. Figure 5-23 illustrates a use of bracket communication.

Because either connected node can initiate a bracket, a *Bid command* can be used to avoid situations in which both nodes attempt to initiate a bracket at the same time. A Bid command requests permission to start a bracket. Upon receipt of a Bid command, the receiving node can reject it, give permission for a bracket to be initiated, or reject the Bid



Note: In this example, the logical unit determines the beginning and the end of the bracket. In other applications, the VTAM application could determine the beginning and the end of bracket, or one node could determine the beginning and the other node determine the end.

Figure 5-23. An Example of Bracket Protocol

temporarily. If the Bid is rejected temporarily, the node that received it transmits a *Ready to Receive* (RTR) *command* when a bracket session can be permitted. Upon receipt of an RTR command, the node that originally sent the Bid can initiate the bracket by sending a Begin Bracket indicator in a message.

To use the bracket convention effectively, only one node should be permitted to initiate a bracket without the use of the Bid. The other node should be required to use the Bid before initiating a bracket.

Like quiesce protocol and change-direction protocol, bracket protocol is not enforced. Any message marked "begin bracket" that is sent before the previous bracket has ended is not rejected by VTAM. Bracket communication is, however, enforced in 3270 communication.

Figure 5-24 lists the three sets of indicators and commands discussed above.

Communicating with the 3270 Information Display System

A VTAM application program can communicate with the 3270 Information Display System in *record mode*. That is, the application program uses SEND and RECEIVE macro instructions as described above.

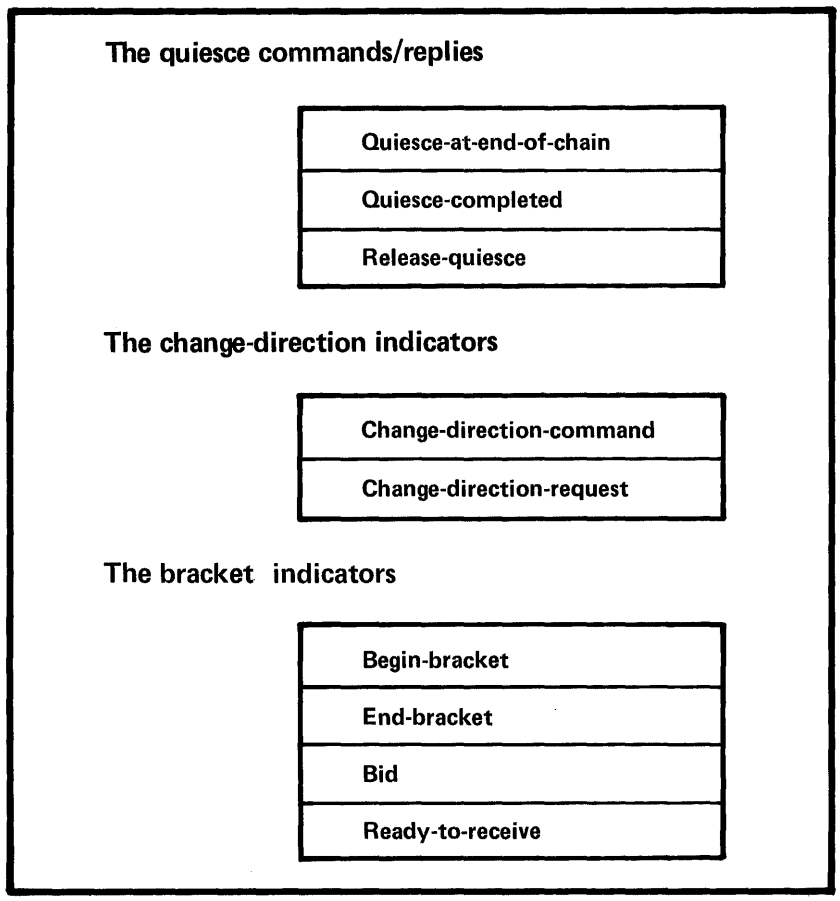


Figure 5-24. Indicators Used to Direct the Flow of Data

The following restrictions apply to session parameters for all SEND and RECEIVE communication with a 3270 (all other aspects of communication apply as described previously):

- All commands and orders for the 3270 must be placed in the output data by the application program. Data, therefore, includes 3270 commands and orders.
- No responses should be sent to a 3270. All incoming messages indicate that no response of any type is expected.
- Messages sent to a 3270 can contain only data and 3270 commands and orders. No quiesce, change-direction, bid, chase, or cancel commands and indicators should be sent. Bracket indicators can be set, but chaining indicators should always mark the message as the sole element of a chain (all incoming messages are so marked).
- No SESSIONC commands can be received from a 3270. Only the clear command can be sent to it. The effect of the clear command is to reset both incoming and outgoing sequence numbers to 0.
- The bracket convention must be used. If the application program has no use for brackets, the entire interval between the first I/O operation of a connection and disconnection can be considered to be one bracket. Both the application program and the 3270 can begin a bracket.

  The first input from a 3270 that begins an NCP session is marked as the beginning of a bracket. All subsequent messages received from the 3270 during the NCP session indicate that the bracket is being continued. The 3270 cannot end a bracket; this can only be done by the application program.
- The application program should request definite response 2 for each message sent to the 3270 that begins or ends in a bracket.

Chapter 8 discusses additional restrictions that apply to these terminals. Appendix D discusses compatibility considerations for communicating so that a program can communicate with both BSC and local 3270 terminals and SNA 3270 terminals in the same way or so that BSC 3270 terminals can be replaced with SNA 3270 terminals with a minimum amount of difficulty. See the chapter on VTAM support in *Introduction to Programming the IBM 3270*, GC27-6999, for a summary of how to use VTAM when programming the 3270.

The VTAM Language

This section defines the functions of each VTAM macro instruction. It also explains how macro instructions and control blocks are related.

Introduction to the VTAM Language

The application program facilities described above are obtained by using VTAM macro instructions and by specifying desired values as parameters in these macro instructions.

VTAM assembler language macro instructions request that VTAM:

- Connect or disconnect the application program to or from the VTAM network and then to specific logical units
- Transfer messages and responses between the application program and a logical unit
- Create control blocks to hold information that is passed between the application program and VTAM and to modify and interrogate these control blocks
- Provide support for connection and communication activities

The operands specified in these macro instructions are in keyword rather than positional format. Some of the operands must be specified; most operands are optional.

Most of the VTAM macro instructions rely upon parameters in the RPL. In addition to being able to modify these parameters with the MODCB macro instruction or assembler language instructions, the application program can change parameters as part of each individual connection, communication, or support request to VTAM.

In general, the VTAM language complements the VSAM language. Both VTAM and VSAM use ACB, EXLST, and RPL control blocks (although the formats of these control blocks differ in the two access methods). Both VTAM and VSAM have macro instructions (GENCB, MODCB, TESTCB, and SHOWCB) that manipulate these control blocks.

Another common feature is the ability to code and specify the scheduling of exit routines; these exit routines are scheduled and executed independently of the other parts of the application program.

The VTAM Macro Instructions

The following provides a description of the macro instructions that were summarized earlier in this chapter. The macro instructions are grouped here, as they were earlier, into the following categories:

- Connection macro instructions
- Communication macro instructions
- Control-block macro instructions
- Support macro instructions

The Connection Macro Instructions

These macro instructions tell VTAM that a particular application program is in operation and, subsequently, request VTAM to connect the application to one or more logical units prior to transferring data. Macro instructions also request VTAM to disconnect the program from one or more logical units and to disconnect the program from the VTAM system.

OPEN: Identifies an application program to VTAM. Once the program is opened, VTAM can schedule any exit routines associated with the program. (The scheduling of a LOGON exit routine also requires the issuance of a SETLOGON macro instruction.)

CLOSE: Indicates to VTAM that an application program is terminating its connection with VTAM.

OPNDST: Requests that VTAM connect the application program to a designated logical unit or list of logical units. Connection is required before communication macro instructions can be used to request data transfer with the logical unit.

CLSDST: Requests that VTAM terminate the connection between the application program and a designated logical unit.

The Communication Macro Instructions

These macro instructions request the transfer of messages and responses between the application program and a logical unit:

RECEIVE: Requests that VTAM transfer an available message or response from a specific logical unit or any one of a group of logical units to the application program's data area (if the input is data) or to one or more fields of the RPL (if the input is indicators or response information).

SEND: Requests that VTAM transfer a message: or a response to a specific logical unit. Data is transferred from an area in the application program. indicators or response information are specified sumbolically in the SEND macro instruction in the RPL, and no output area is required.

SESSIONC: Requests that VTAM send to a logical unit a SESSIONC command which either (1) starts or stops the possibility of exchanging messages with the SEND and RECEIVE macro instructions or (2) assists in synchronizing the message sequence numbers being kept by a logical unit.

RESETSR: Requests that VTAM change the continue-any/continue-specific mode of the logical unit. It also cancels any outstanding RECEIVE macro instructions that request input from the specific logical unit.

The Control-Block Macro Instructions

These macro instructions build and manipulate control blocks required by VTAM application programs. The first part describes the control blocks and lists the macro instructions used to assemble each when the application program is assembled. The second part describes the macro instructions that generate and manipulate the control blocks during program execution.

Declarative Macro Instructions: VTAM provides macro instructions to create these control blocks in the application program:

Access Method Control Block (ACB): Contains information the application program provides VTAM about the application program in its entirety. Primarily, it names the application program and the list of exit routines associated with the application. The ACB contains information about the *application program.*

Exit List (EXLST): Contains the addresses of special exit routines that VTAM schedules when certain conditions occur (for example, when a logon request is received from a logical unit). The EXLST contains the names of *exit routines.*

Node Initialization Block (NIB): Contains information the application program provides VTAM about general communication characteristics between the application program and a particular logical unit. This information is provided to VTAM as part of a connection request; it remains in effect for the duration of a connection. The NIB contains information about a *logical unit.*

Request Parameter List (RPL): Contains information (parameters) that an application program provides VTAM when requesting connection to a logical unit, input/output, and any action except the opening and closing of an ACB or the manipulation of a control block. On completion of the requested action, the RPL contains information that VTAM passes to the application program. The RPL contains information about a *request.*

The ACB, EXLST, NIB, and RPL control blocks can be assembled in the application program by using the macro instruction associated with each control block, or they can be created and initialized during program execution, using the GENCB macro instruction. The type of control block that GENCB is to generate is specified in one of the operands in the macro instruction. The GENCB macro instruction is discussed below.

The Manipulative Macro Instructions: Like VSAM, VTAM provides a group of macro instructions that build control blocks or manipulate control-block fields. These macro instructions provide a more convenient release-independent way to do this than by using assembler language instructions. They refer to fields symbolically rather than by specific control block location. The manipulative macro instructions are:

GENCB: Builds an ACB, EXLST, NIB, or RPL during program execution and can initialize designated fields with specified values.

SHOWCB: Obtains the value or values from one or more fields of a control block and places them in an area in the application program where they can be examined. In addition to fields that are set by the application program's use of macro instruction keyword operands, a number of control block fields can be shown that are set by VTAM but that cannot be directly modified by the application program.

TESTCB: Tests the contents of a field against a value and accordingly sets the condition code in the program status word (PSW).

MODCB: Changes the contents of one or more fields by inserting specified values in the fields.

There are several different forms of the manipulative macro instructions. In addition to the standard form, there is a list form, a remote list form, a generate form, and an execute form. The nonstandard forms can be used for programs that must be reenterable or that share the parameter lists that are assembled when the macro instructions are expanded.

Instead of using the manipulative macro instructions, a VTAM application can manipulate values in these control blocks by using DSECT and other assembler language macro instructions. While less convenient to code, fewer instructions will be executed.

The Support Macro Instructions

VTAM provides these additional macro instructions to support connection and communication acitvities:

CHECK: Checks and, if necessary, awaits completion of a previously requested RPL-based operation, marks as inactive the RPL associated with the request (thus freeing it for further use), and, if a logical or other error is detected and a LERAD or SYNAD exit routine exists, causes the appropriate routine to be called.

EXECRPL: Requests that VTAM perform an operation that is currently specified in a designated RPL. EXECRPL can be used:

- To reissue a specified request. When a SYNAD exit routine is entered with a "retriable completion" return code, EXECRPL can be issued with assurance that the RPL contains the desired values. Other use of EXECRPL to reissue a request requires that any RPL fields that may have been changed as a result of the original request be reset. For example, on reissuing a SEND that resulted in an exception response, if a definite response is required, the RPL RESPOND field must be reset from EX to NEX.
- Instead of using other RPL-based macro instructions, such as OPNDST, SEND, and RECEIVE. Prior to issuing an EXECRPL, the operation to be performed must be set in the RPL; this requires the use of the IBM-supplied RPL DSECT. Other parameters may either be set in the RPL or specified with keyword operands when the EXECRPL macro is issued. While less convenient to code, the use of EXECRPL can result in fewer instructions being executed and/or in less storage being taken up by macro expansions. Note that EXECRPL cannot be used to issue or reissue a CHECK request that has failed, since CHECK does not alter the operation field of the RPL.

INQUIRE: Obtains certain information that the application program may need and places it in a specified area of the program. The information that can be requested using INQUIRE includes: the logon message associated with a logon request, the logon mode (set of session parameters) associated with the request, the number of logical units currently connected, or queued for logon to, the application, the physical address of a 3270 screen (for use in a copy operation), and whether another application program is active or inactive and whether it is accepting logon requests.

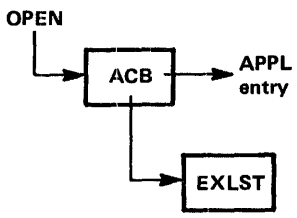
INTRPRET: Provides access to an installation-defined interpret table. For example, INTRPRET can be used to obtain the real symbolic name of an application program when the program is identified with an alias in a logon message. INTRPRET can be used by special programs written to receive logon messages and reconnect logical units to the appropriate application program.

SETLOGON: Tells VTAM to activate the application program's LOGON exit routine and to begin queuing any logon requests. The user can also temporarily halt the queuing of logon requests until more logical units can be handled or can permanently halt the queuing of logon requests in preparation for a close-of-day operation.

SIMLOGON: Allows the application program to acquire a logical unit by initiating the logon request on behalf of one or more logical units to which the application wants to be connected.

Relating VTAM Control Blocks and Executable Macro Instructions

The control blocks created by VTAM macro instructions are used to pass information between the application program and VTAM. One control block, the EXLST, is used simply to pass the names of exit routines in the application program to VTAM. The other control blocks, the ACB, RPL, and the NIB, are used both to pass information to VTAM and to contain information that VTAM returns to the program.



The most frequently used control block is the RPL. An RPL is required each time the program makes a request for a connection, for an I/O operation, or for any service except opening and closing an ACB. The other control blocks, the ACB, EXLST, and NIB, may be used as little as once during the execution of the program.

Each RPL-based macro instruction refers to an RPL that contains information about the operation to be performed, such as the address of the input or output area, the length of the area, and the logical unit to be communicated with. This information is placed in the RPL initially when the control block is created or subsequently, either by using a MODCB macro instruction or by providing the information in the request macro instruction itself. For example:

```
RECEIVE     RPL=RPL1,AREA=INFO,AREALEN=200
```

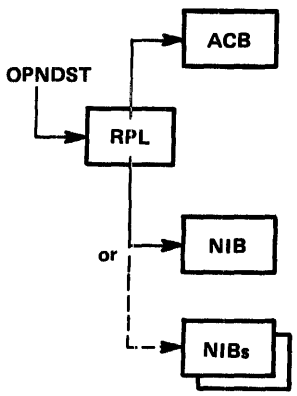
changes the value in RPL1's AREA and AREALEN fields to INFO and 200 respectively before initiating the input operation. These values would remain in these fields for all subsequent requests that used RPL1 until they were changed again.

The relationship of the VTAM control blocks to each other and to the macro instructions that refer to them can be described in the context in which they are used:

- Opening the application program—that is, identifying itself to VTAM as operational
- Connecting to logical units with which it will communicate
- Communicating with connected logical units

These activities as they relate to the VTAM language are discussed below.

Opening the Application Program



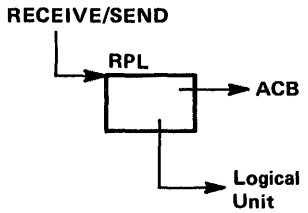
The OPEN macro instruction indicates to VTAM that a program is now an operational part of the VTAM network. The OPEN specifies an ACB; the ACB in turn points to a location in the program that contains the name of the APPL program as defined in an APPL statement during VTAM definition. The ACB may also point to an EXLST control block containing the names of exit routines that are to be associated with the application program. (An EXLST can also be pointed to when a logical unit is connected; see "Connecting Logical Units", below.) When the open process is complete, any exit routines that have been specified are eligible for scheduling by VTAM.

A single OPEN macro instruction can open more than one ACB at a time. This means that a program that performs related functions (for example, communicating with both logical units and other terminals) may be defined so that it is viewed by VTAM as more than one application program. Probably most VTAM users will find it satisfactory to open only one ACB for each program.

The CLOSE macro instruction notifies VTAM that an application is detaching itself from the VTAM system. As a result of issuing a CLOSE macro instruction, any logical units still connected to the program are disconnected.

Connecting Logical Units

Before communicating with a logical unit, an application program must have itself connected to the logical unit. Either the logical unit or the application program can initiate connection; in either case, it is the application program that formally requests connection, by using an OPNDST macro instruction. The OPNDST macro instruction specifies an RPL that is associated with the request. The RPL contains the address of a NIB. The NIB contains information that applies to subsequent communication with the logical unit; this information can include a unique storage address to be associated with the logical unit. If a number of logical units are to be connected, a single OPNDST can be used, with the RPL pointing to a list of NIBs instead of to a single NIB.



Optionally, a NIB can point to a list of exit routine names in an EXLST control block. For the logical unit being connected, these exit routines are used in preference to any equivalent exit routines identified for the entire application program when the ACB was opened.

When a logical unit is connected as the result of an OPNDST macro instruction, VTAM returns information about the logical unit in the RPL and the NIB. In both the RPL (if only one logical unit is connected) and the NIB, VTAM places a communications identifier (CID) that it has assigned to the logical unit. The CID contains an origin (application program) identifier and a destination (logical unit) identifier; it is shorter and requires less space and search time for VTAM than the symbolic name of the logical unit specified during VTAM definition. On all subsequent I/O requests for the logical unit, the application program must be sure that this CID is located in the RPL. In addition to the CID, VTAM also places the logical unit name, characteristics, and other information in the NIB; if desired, the application program can use this information to determine how to communicate with the logical unit.

Once a NIB has been used to connect one logical unit, it can be reinitialized and reused for connection with other logical units.

Disconnecting Logical Units

Once a series of communications between the application program and a logical unit is complete, the program can use the CLSDST macro instruction to disconnect the logical unit.

If the program is terminating and all logical units are to be disconnected at the same time, the CLOSE macro instruction, used to close the ACB, also disconnects all connected logical units and the CLSDST macro instruction need not be used.

Communicating with Logical Units

Having opened the application program's ACB and having connected one or more logical units to the program, the program can communicate with each connected logical unit by issuing SEND and RECEIVE macro instructions. VTAM obtains the name of the application program that made the request and the identity of the logical unit (if a specific logical unit is being addressed) from the RPL. The communication macro instruction specifies an RPL; the RPL contains the address of an ACB and the identity of the logical unit.

Only data is written from or read into an application program data area. Indicators and response information are sent as a result of being specifed symbolically in a SEND macro instruction or in an RPL. Indicators and response information that are received are not read into a data area but are detected by analyzing fields in the RPL associated with a

RECEIVE macro instruction or in an RPL associated with the scheduling of an exit routine specified to handle the receipt of indicators or response information.

VTAM provides two ways to write to a logical unit. A SEND macro instruction can be issued that specifies POST=RESP, in which case VTAM handles any response signals that are returned and notifes the application program when the requested operation is complete; this is called responded output. (If necessary, the application program can obtain response information from the RPL.) A SEND macro instruction can also be issued that specifies POST=SCHED, in which case VTAM considers the SEND complete as soon as the output operation has been scheduled; this is called scheduled output. The application program thus has the responsibility for determining completion either by having a RECEIVE macro instruction that specifies response information is to be received or by having a RESP exit routine that VTAM schedules each time response information arrives.

Handling Sessions

A VTAM application program may be required to handle only one type of session, such as an interactive session or a batch session, or it may be required to handle several different types of session. A type of session is distinguished by a set of session parameters that are agreed upon by the logical unit and the VTAM application program when the logical unit is connected. Defining the type of session (that is, the specific set of session parameters) consists of specifying the particular rules or protocols (for example, whether or not chaining will be permitted) agreed to by both the logical unit and the application program. The logical units asssociated with certain IBM terminals must conform to the rules of a particular type of session (for example, the IBM 3767 Communication Terminal must use an interactive type of session); the logical units associated with other terminals can request any of several types of session. The VTAM application program must be written with an awareness of what types of session may be requested.

The VTAM application program can either use the set of session parameters that has been initially defined for a logical unit during VTAM definition or specified by the logical unit in its logon request or it can itself specify the set of session parameters. For example, if the VTAM application program is acquiring a logical unit itself, it may know whether it is going to want an interactive or a batch session. If the logical unit is initiating connection, the application program can determine the type of session requested by issuing an INQUIRE macro instruction that specifies OPTCD=SESSPARM. An IBM-supplied DSECT (ISTDBIND) can be used to examine the session parameters and determine the appropriate program logic. If the program wants to attempt to override the requested set of session parameters or is itself the initiator of the connection request, the program can use the DSECT to set the desired parameter values in an area pointed to by the BNDAREA field of a NIB. The NIB is then specified in an OPNDST macro instruction. If the session parameters are not agreeable to the logical unit, the OPNDST fails.

Handling Control Blocks, I/O Areas, and Work Areas

The application program can handle control blocks in a number of ways. It can:

- Define RPLs, NIBs, or EXLSTs in the application program during assembly or generate them, using the GENCB macro instruction, during program execution
- Assign one RPL or NIB to a specific logical unit during assembly, or either assemble or generate RPLs and NIBs that are to be available for any logical unit as the need arises
- Retain the RPL used in connecting the logical unit for all further communication with the logical unit
- Use one RPL for all connection requests and use another RPL or group of RPLs for all communication requests
- Define the RPLs, NIBs, and any other required control blocks or work areas to be associated with logical units as a pool so that a limited amount of control block storage is not exceeded

In application programs that must handle many logical units concurrently, it may be useful to have a control block work area other than the RPL or NIB associated with a particular logical unit or logical unit conversation. The queuing of input or output by the application program between its processing routines and logical units may also require a logical unit-related control block work area. VTAM provides a user field in the RPL that may be used to associate storage with a particular logical unit. The application program initially associates the storage with the logical unit when the logical unit is connected by specifying an address in the USERFLD of the NIB; thereafter, when input is received from the logical unit VTAM provides the address associated with it in the RPL's user field.

Coordinating I/O Activity

The VTAM user has several ways to coordinate the transfer of data to and from logical units with the preparation and processing of data by the program. The simplest way is to make the program wait each time an I/O operation is requested until that operation is completed. This is called synchronous request handling and is specified by coding OPTCD=SYN in the I/O request macro instruction or in the associated RPL. This method is satisfactory for the transmission of data between an application program and one or a few logical units, or for reading a message from a master logical unit where all further program action depends on analysis of that message.

Most programs, however, require asynchronous request handling. Asynchronous request handling is specified by coding OPTCD=ASY in the I/O request or in the associated RPL. In this type of operation, the program continues to be executed after an I/O request is accepted by VTAM. On completion of the requested operation, the program specifies whether VTAM schedules an exit routine associated with the request (specified by coding the EXIT operand in the macro instruction or associated RPL) or posts an ECB (specified by coding the ECB operand). If an RPL exit routine is specified, it is scheduled automatically for the application program. An ECB, if specified, must either be checked periodically to see if it is posted, or the application program can be put into a wait state, using either a system WAIT or WAITM macro instruction of VTAM's CHECK macro instruction. A WAIT or WAITM macro instruction can be for one or any one of a list of ECBs; a CHECK can wait only for the completion of the operation associated with one RPL. (Whether an RPL exit routine or ECB posting is used, a CHECK must be used to make the RPL available for reuse.)

In OS/VS2, performance can be further improved by specifying execution of individual SEND, RECEIVE, and RESETSR macro instructions in a path that requires fewer instructions. An installation must authorize a VTAM application program to use this facility. See "Using Authorized Path in OS/VS2" further in this chapter.

Sample Programs

This section shows and discusses the general logic of two sample application programs. Following the discussion of the sample programs is a discussion on VTAM alternatives to programming techniques used in the samples.

Introduction to the Sample Programs

The sample programs in this section show how VTAM macro instructions are used in VTAM application programs. Many, but not all, VTAM facilities and language features are illustrated in these sample programs. In the text that explains the programs, some alternative facilities or techniques are discussed, but they are not shown in the figures.

Sample Program 1 shows how simple a VTAM program can be; it is not designed to portray a typical program.

Sample Program 2 illustrates additional VTAM programming concepts and techniques and contains more detail than Sample Program 1. Sample Program 2 communicates with logical units associated with a 3600 Finance Communication System and SNA 3270 Information Display System terminals. It communicates with both types of logical units using the record mode macro instructions (SEND and RECEIVE).

The logic of each program is displayed in flowcharts that show the major decision points but do not show all program instructions. However, the key macro instructions and operands are shown, and they are discussed in notes that are keyed to each figure. Programming examples are provided throughout this section. Language options are explained in the text as they are introduced.

To follow the sample programs, the reader must be familiar with programming techniques for the acquiring and handling of control block areas, the posting of ECBs, and the scheduling of processing routines by a telecommunication program. Following Sample Program 2, in "Choosing Programming Alternatives Discussed in the Sample Programs," is a discussion of alternatives to some of the programming techniques used in the sample programs.

Sample Program 1

Figure 5-25 shows a VTAM application program that would handle a request for logon (connection) for a logical unit, connect it, read input from any connected logical unit, process the input, prepare a reply for output, and then write the output to the logical unit.

Sample Program 1 demonstrates use of:

- A LOGON exit routine and the acceptance of a logon request
- A request to receive input from any connected logical unit
- Synchronous I/O requests
- Continue-any and continue-specific modes
- A SEND macro instruction to send a response rather than data
- A request to *schedule* output
- A RESP exit routine to handle a response received from the logical unit each time it receives data
- A TPEND exit routine

For simplicity, error recovery routines and other special routines are omitted; these are discussed in Sample Program 2.

Sample Program 1 might be feasible for a VTAM application program for which each transaction between the application and the logical unit consisted of a short inquiry and a short reply. Because the application program waits for the processing for one logical unit to complete before reissuing a request for input from any connected logical unit, the application might not adequately serve a large number of logical units communicating with the application program at the same time.

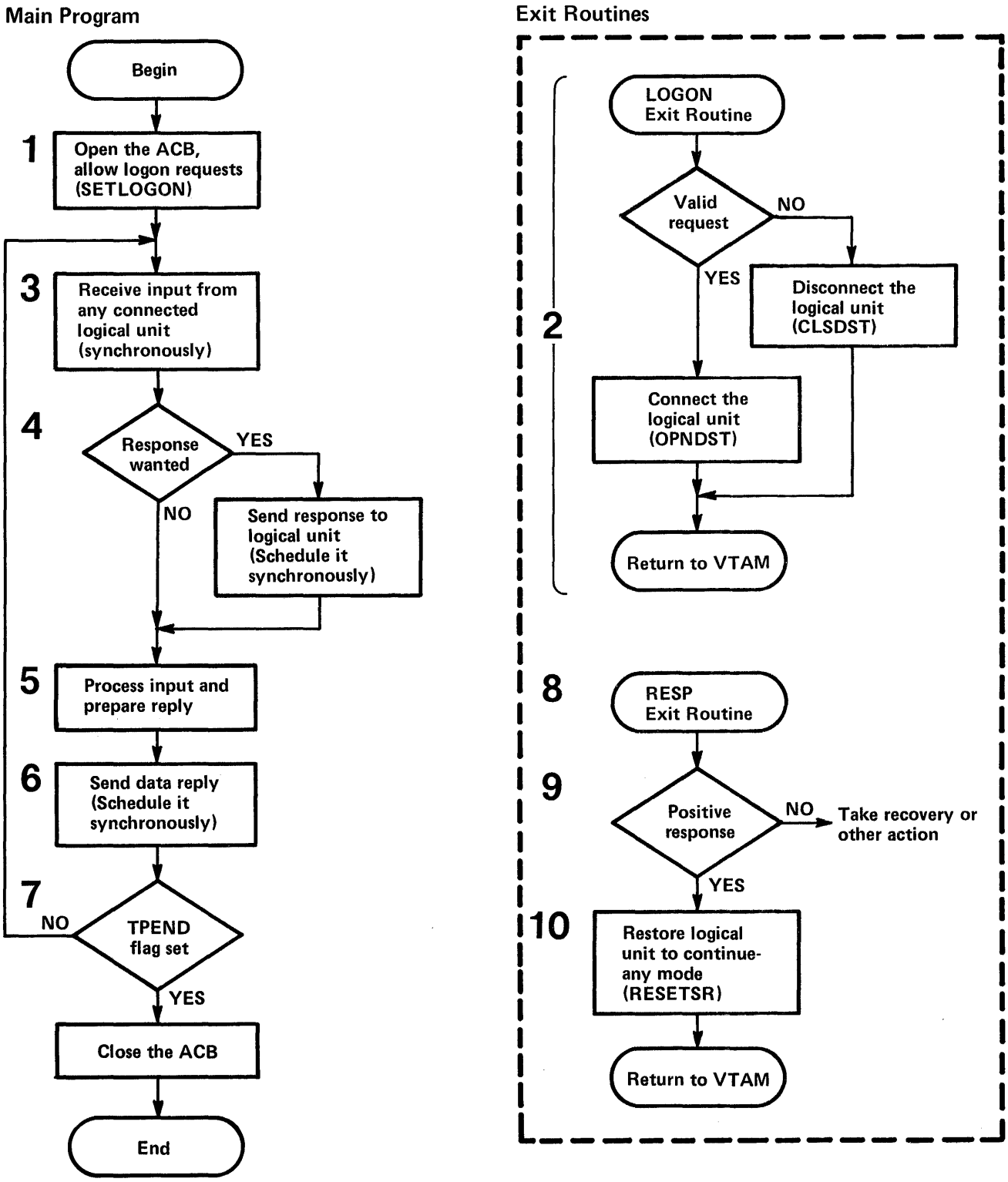


Figure 5-25. The Logic of Sample Program 1

The Logic of Sample Program 1

These notes are keyed to Figure 5-25.

1 The OPEN macro instruction opens the ACB, defined in the program with the ACB macro instruction. For example, this might be coded:

```
OPEN        ACB1
```

ACB1 contains:

```
ACB1        ACB     AM=VTAM,APPLID=APPL1,EXLST=EXLST1,
                    MACRF=LOGON
```

APPL1 contains:

```
APPL1       DC      X'05'
            DC      C'PROG1'
```

PROG1 is the name of the APPL statement used to define the application during VTAM definition.

EXLST1 contains the names of the exit routines shown in Figure 5-25. MACRF=LOGON indicates logon requests are accepted.

After the OPEN macro instruction, a SETLOGON macro instruction is used to tell VTAM to begin processing logon requests for the application. The macro instruction might be coded:

```
SETLOGON    RPL=RPL1,OPTCD=START
```

2 The LOGON exit routine, whose address is specified in the LOGON operand of the EXLST macro instruction (whose address is in turn specified in the ACB macro instruction), is scheduled when VTAM receives a request for connection from or on behalf of a logical unit. If no other exit routine is currently being executed, the LOGON exit routine is given control. When the LOGON exit routine is completed, either the next-scheduled exit routine receives control or the main program regains control at its next sequential instruction.

An INQUIRE macro instruction could be used to obtain the logon message when there is one. By examining the logon message, the exit routine determines whether the logical unit should be connected to the program. If so, an OPNDST macro instruction like this might be issued:

```
OPNDST      RPL=RPL1CONN,OPTCD=(ACCEPT,SPEC),NIB=NIB1
```

An RPL and a NIB (RPL1CONN and NIB1, respectively, in this example) are required for the OPNDST request; the same RPL and NIB can be reused by the OPNDST each time the LOGON exit routine is entered. If necessary, the type of terminal, the set of session parameters (logon mode), and other communication characteristics can also be obtained using the INQUIRE macro, and the NIB properly initialized with this information prior to issuing the OPNDST macro. (The MODCB macro instruction can be used to initialize the NIB.)

OPTCD=(ACCEPT,SPEC) specify options of the OPNDST requesting that a logon request from a *specific* terminal (identified in the NIB) be *accepted.*

If the logical unit is not to be allowed to use the program, a CLSDST macro instruction must be issued to notify VTAM that the logical unit's logon request is being rejected. Although not shown in the flowchart, it may be desirable to connect the logical unit (using OPNDST), write an appropriate message to it, and then disconnect.

3 In this example, the request to receive input from any connected logical unit is issued in the main program. In this case synchronous request handling is specifed; that is, the application program indicates that it will wait until input is received from one of the logical units. In making the request, the application identifies an RPL and an input area. The macro instruction might be coded:

```
RECANY    RECEIVE  RPL=RPL1,AREA=AREA1,AREALEN=100,
                   RTYPE=DFSYN,OPTCD=(SYN,ANY,CS)
```

or it could be coded:

```
RECANY    RECEIVE  RPL=RPL1
```

where RPL1 was coded:

```
RPL1        RPL    AM=VTAM,ACB=ACB1,AREA=AREA1,
                   AREALEN=100,RTYPE=DFSYN,
                   OPTCD=(SYN,ANY,CS)
```

The input request could have been coded with some operands appearing in the RPL and others in the macro instruction. If an operand that appears in the RPL is also coded in the macro instruction, the value in the macro instruction replaces the value in the RPL and is in effect not only for the current operation but for subsequent operations that use the RPL. Note also that values in some RPL fields are reset by VTAM in order to provide information about a requested operation. Subsequent requests may require that the values in these RPL fields be respecified.

AM=VTAM indicates the RPL is to be used for a VTAM request. ACB=ACB1 specifies that the application program issuing the request is using ACB1.

AREA=AREA1 specifies that any input data resulting from the request should be moved to the application's input area AREA1 which has a length of 100 bytes (AREALEN=100).

RTYPE=DFSYN is specified so that the receipt of data or of a normal-flow SNA control command completes the request. (The receipt of a response causes the RESP exit routine to be entered; see 8 below.)

CS is specified so that the logical unit whose input is read by the RECEIVE is put in continue-specific mode until its inquiry has been successfully answered. Thus, if the logical unit is still in continue-specific mode when the next RECEIVE with OPTCD=ANY is issued, input from that logical unit does not satisfy the any-input request. This allows input from other logical units to be processed; a single, busy logical unit does not monopolize the program. The logical unit is put back in continue-any mode when a response has been received acknowledging that the reply to the inquiry arrived successfully; in this sample program, this is done in the RESP exit routine.

Assume that a logical unit just connected sends in the first inquiry to the program. The synchronous RECEIVE completes. If register 15 contains 0, the operation was successful. If register 15 contains some other value than 0, an error or special condition has occurred. A LERAD or SYNAD exit routine in the program may have been entered, and may have returned a code in register 15 or register 0 that will indicate further action for the program to take. Whether a LERAD or SYNAD was entered, information is available in various feedback fields of the RPL for analysis. One of the errors that can occur is that the message arrived as an exception; this information is available as one of the feedback return codes in the RPL.

Assuming the operation was successful, the identity of the logical unit whose input was received is provided in the ARG field of the RPL. Data is located in AREA1, and a SHOWCB or TESTCB can be used to determine its length (by examining the RECLEN field of the RPL).

4 The logical unit that sent the data may have indicated that the program should issue a response to verify that the input has been received.

There may be some occasions when the logical unit wants a response, perhaps to verify that a data base update message has been received so that it can free its buffers. On other occasions, such as an inquiry message, the logical unit may not want a response (or want a response only if an exception condition occurs); the answer to its inquiry will be forthcoming soon and will be implicit assurance that the inquiry arrived. The VTAM application program can examine the RESPOND field of the RPL to determine whether and under what conditions a response is required. If completion information following the RECEIVE indicates that the input was received normally and the RESPOND field indicates that a normal response is required, it is sent with a SEND macro instruction that might be coded:

```
SENDRESP    SEND  RPL=(2),STYPE=RESP,OPTCD=SYN
```

If completion information following the RECEIVE indicates that an exception message was received (in which case there will be no input to process), and the RESPOND field indicates that a response is requested, it is sent with a SEND macro instruction that might be coded:

```
SENDRESP    SEND  RPL=RPL1,STYPE=RESP,OPTCD=SYN,RESPOND=EX
```

If the logical unit wanted a response only in the event of an exception, the RESPOND field will already be set to EX and will not have to be reset in the SEND. Before issuing the SEND for the exception response, the program would set up sense information in the RPL, defining the exception.

The same RPL used for the RECEIVE request can be reused for the SEND. Since a response is being sent (STYPE=RESP), no data area or length is needed. For a response, POST=SCHED completion is assumed. The request is specified for synchronous handling (OPTCD=SYN). However, because it is only being scheduled, the operation takes a relatively short time. As soon as the operation has been scheduled and the SEND completes, the RPL can be reused.

5 The inquiry is analyzed and a reply is prepared by a processing routine. Disk I/O may be required. If so, the program waits until the reply is ready. (This sample program assumes that this processing time is acceptably short.)

6 The reply is then sent with a SEND that requests the transmission of data (STYPE=REQ). In this sample program, the same area used to receive input is used for output. The macro instruction might be coded:

```
SENDDATA    SEND  RPL=RPL1,STYPE=REQ,RESPOND=(NEX,FME),
                  OPTCD=SYN,POST=SCHED
```

Since the RPL currently contains the address of AREA1 in the AREA field, AREA=AREA1 need not be respecified in the macro. So that the application program can determine whether the logical unit receives the data successfully, a response from the logical unit is requested (RESPOND=(NEX,FME)). NEX indicates that either a positive or a negative response is to be returned (one or the other). Although not shown, EX indicates only a negative response is to be returned, and NFME indicates no response at all is to be returned. The macro

instruction requests the scheduling of the operation; the request is completed synchronously (OPTCD=SYN) as soon as the sending of the output has been scheduled (POST=SCHED). Completion of the operation is determined as a result of the program's receiving the definite response 1 (FME) response.

With the reply under way, a branch is made back to the RECEIVE so input that may have been read into VTAM's buffers from another connected logical unit (that is not in continue-specific mode) can be read into the application program for processing.

7 If the VTAM network is being halted, the application program's TPEND exit routine (not shown) is scheduled and entered by VTAM. This routine can indicate to the main portion of the application program that the application is to terminate. The TPEND exit routine or later the main program can send final messages to connected logical units and do other close processing depending on whether the closedown is quick or a routine end-of-day (normal) closedown. The main program, discovering the closedown requirement, must issue a CLOSE macro instruction to disconnect the application program from the network; any logical units not yet disconnected are, as a result of the CLOSE, disconnected. Note that the CLOSE macro instruction must be issued in the main program and not in the TPEND exit routine. The application program terminates by returning control to the operating system.

8 When a response to the data sent in 6 is received by VTAM, the RESP exit routine is scheduled and entered. On entry, register 1 points to a parameter list that points to a read-only RPL in VTAM's storage, whose feedback fields can be examined to determine the kind of response received.

9 If a negative response has been received, the application program can determine from sense information in the RPL whether or not to retry the operation, disconnect the logical unit, or take some other action.

10 If a positive response is received, the logical unit can be returned to continue-any mode so that the next request for input from any logical unit includes this logical unit as one whose input can be read by the VTAM application program. This is done by issuing:

```
RESETSR     RPL=RPL1R,OPTCD=CA,RTYPE=DFSYN
```

The RESP exit routine must have its own RPL available. The identity of the terminal to be placed back in continue-any mode must be put in the ARG field of the exit routine's RPL. The RESP exit routine then returns control to VTAM.

Sample Program 2

Sample Program 2 is a more typical example of a VTAM application program than Sample Program 1. Sample Program 1 should be read first to gain an understanding of some of the elementary concepts of using VTAM macro instructions.

Sample Program 2 communicates with logical units associated with 3600 Finance Communication Systems and SNA 3270 Information Display Systems. Logical units are connected to VTAM on nonswitched lines through a 3704 or 3705 Communications Controller with a network control program. Additionally, local 3270s are attached directly to VTAM through a channel.

Figure 5-26 shows a possible configuration of terminals with which Sample Program 2 might communicate.

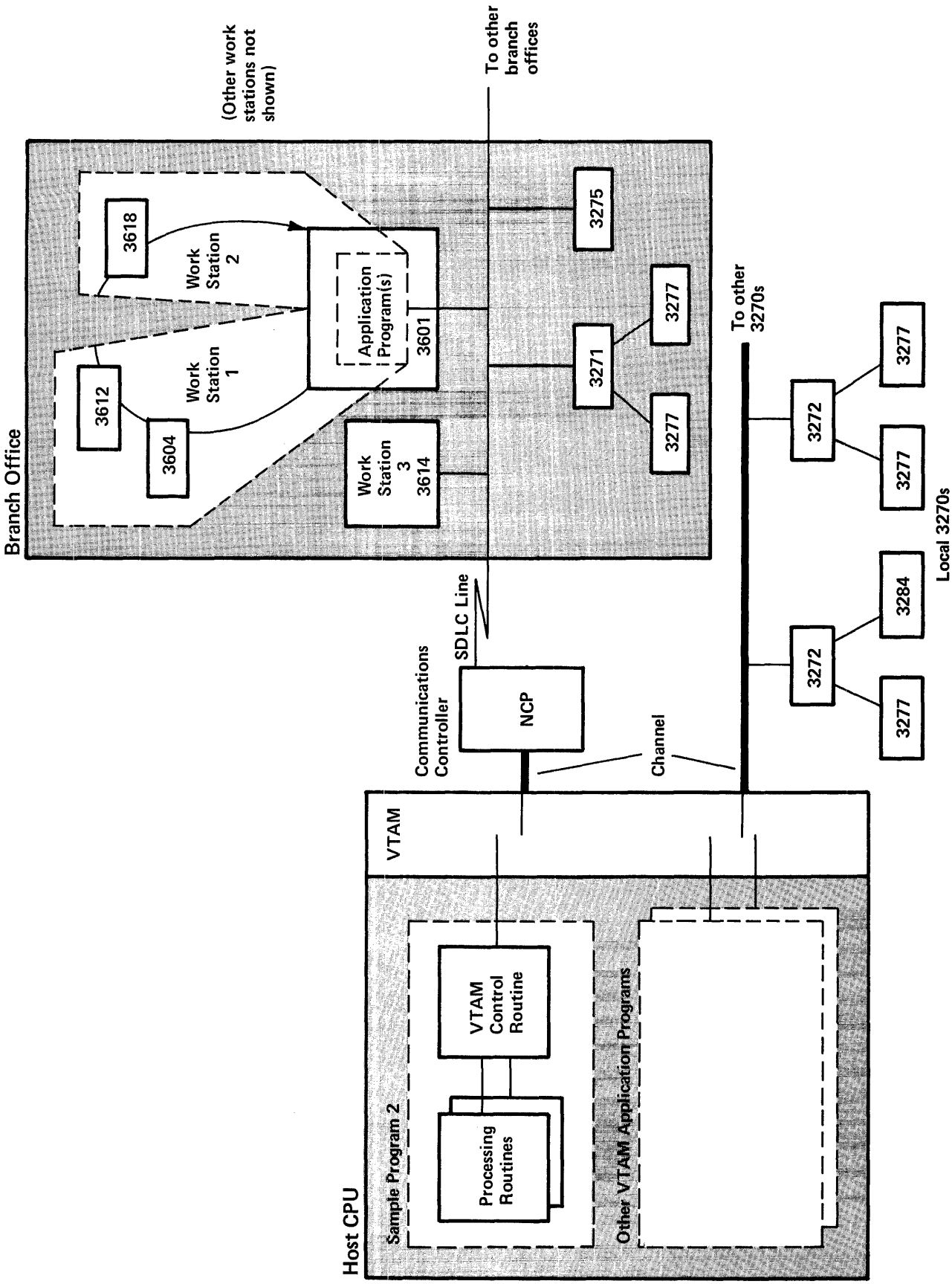


Figure 5-26. Hypothetical Network Configuration for Sample Program 2

Note in Figure 5-26 that the application program in the 3601 controller can be written to perform certain functions that would otherwise have to be performed by the VTAM application program. For example, the 3601 application program could be written to screen inquiries for correct format prior to forwarding them to the VTAM application program for processing, or it could be written to collect inquiries from several devices that form a work station and send them as one transmission to the VTAM application program.

The logic of Sample Program 2 is described at a high level in Figure 5-29 and accompanying notes. The logic of special routines is described in more detail in subsequent figures and accompanying notes.

Sample Program 2 uses the posting of ECBs, either by VTAM or within the application program, and a central wait routine that detects posted ECBs to handle a number of terminals alternately without suspending all program execution while awaiting completion of I/O operations. After a request has been issued for an operation to be performed asynchronously to the main program, control is transferred to the wait routine, which detects (or, if necessary, waits for) a posted ECB. When the ECB is posted, the wait routine determines the event that completed and branches to a point in the program to provide further processing for the event. An understanding of the details of this technique is assumed in this discussion.

Although not discussed in detail, it is likely that Sample Program 2 would use a separate work area (or control block) for each terminal that is actively using the program. This work area might include an input/output area. A separate RPL might be associated exclusively with each active terminal. The storage for the RPL and the work area might be obtained from a fixed pool or be obtained dynamically and the RPL initialized using the GENCB macro instruction. This work area and RPL area might be obtained and related to a terminal for the duration of its connection, for the duration of the program, for the duration of a transaction or conversation, or on some other basis. The ECB associated with a terminal could be located in the RPL or outside of it in some fixed relationship, perhaps just in front of it. In Sample Program 2, it is assumed that the storage for an ECB, RPL, and work area is obtained and initialized in the LOGON exit routine and retained for the duration of the terminal's connection to the application program.

The Organization and Flow of Sample Program 2

Figure 5-27 shows the principal routines in Sample Program 2; the notes below indicate how Sample Program 2 works. More detailed logic is shown and discussed in subsequent figures and notes.

Figure 5-27 shows the main portion of the application program (all parts of the application program other than the exit routines) and the exit routines as separate groups of routines. This is a logical rather than a physical separation; exit routines are distinctive because they are entered only when an event occurs that requires handling by an exit routine. VTAM suspends execution of the main program until the exit routine completes its processing and returns to VTAM. Only one exit routine can be executed at a time; if an exit routine event occurs while an exit routine is being executed, the second exit routine is scheduled for entry only after the first exit routine is completed. The LERAD and SYNAD exit routines are exceptions to this general rule; they can be entered as the result of a RPL-based request in the main program or another exit routine (in which case, they may be viewed as extensions of the exit routine that caused them to be entered) or within themselves.

Except for the LERAD and SYNAD exit routines, each exit routine must establish its own addressability, be executed, and then return to VTAM; VTAM's registers need not be saved or restored. A temporary branch to part of the main program could be made from

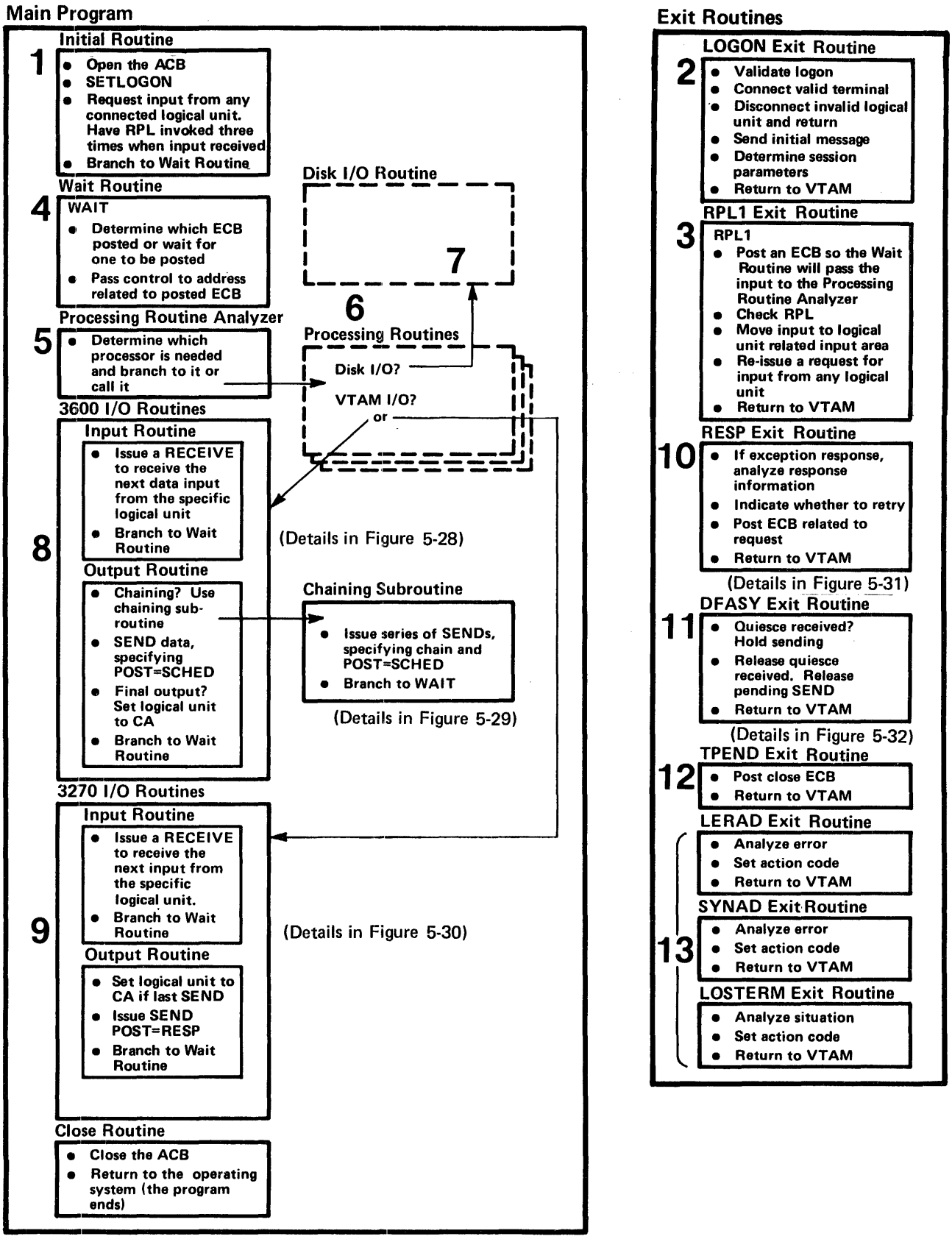


Figure 5-27. Organization and Flow of Sample Program 2

an exit routine–common code could be shared–but the exit routine is considered to be in progress until control is returned to VTAM. The LERAD and SYNAD exit routines establish addressability by loading registers (the user save-area address is passed in register 13).

Except for an RPL exit routine, whose address is specified in the RPL or request macro instruction, the addresses of the exit routines to be associated with the program are defined in an EXLST macro instruction. In addition, for DFASY, RESP and SCIP exit routines, different exit lists can be defined for different terminals or sets of terminals. In Sample Program 2, one exit list is assumed; the ACB's EXLST operand provides the address of this exit list to VTAM when the ACB is opened.

The following notes are keyed to Figure 5-27.

1 As in Sample Program 1, the ACB is opened and a SETLOGON is issued. A request to read input from any logical unit is issued; the operation is performed asynchronously to the execution of the application program, and the RPL exit routine (RPL1) is designated for scheduling by VTAM when the operation completes. The logical unit whose input is read into Sample Program 2 is to be put into continue-specific mode. Thus, subsequent requests to read input from any logical unit issued in the RPL exit routine exclude the logical unit whose input was just read and with whom the program is now in specific communication. The RECEIVE might be coded:

```
RECEIVE     RPL=RPLANY1,AREA=AREAANY1,AREALEN=100,
            RTYPE=DFSYN,OPTCD=(ASY,ANY,CS)
```

Efficiency might be improved by issuing more than one request for input from any logical unit at this point. For example, three RECEIVEs might be issued, using three different RPLs and data areas. This would mean that if one RECEIVE completed (thus causing the RPL1 exit routine to be scheduled and entered), there would be two other RECEIVEs still outstanding that could allow scheduling of RPL1. The RECEIVE that completed first would be reissued in the RPL1 exit routine.

The RPLs and input areas can be assembled in the program as fixed areas and reused each time the program issues a request to read input from any logical unit. The ASY option of the OPTCD parameter specifies asynchronous request handling.

A branch is made to the wait routine, which waits for the first input from a connected logical unit. The code in the initial routine is executed only once.

2 Both 3600 and 3270 logical units can be connected in the LOGON exit routine. The INQUIRE macro instruction is used to determine which type of logical unit is being connected. For logical units, the VTAM application program does not have to identify the particular type of SNA terminal product (such as 3600, 3790, or 3270). Instead, the VTAM application program distinguishes between types of logical units on the basis of the set of session parameters associated with the logical unit. In this example, 3600 logical units have a different set of session parameters than 3270 logical units; the program can relate a logical unit that is being connected with one of these sets and use the appropriate logic to communicate with the logical unit. Storage that is to be associated with this logical unit can be obtained from a pool or can be obtained dynamically from the operating system. (The storage can include an ECB, an RPL, and a work area for additional logical unit-related information.) The address of this storage or other logical unit-related information can be put in the user field of the NIB, using the MODCB macro; when

the logical unit is connected, VTAM saves the contents of this field and returns it to the program at the completion of each subsequent input from the logical unit.

A logical unit can be connected as the result of a logical unit-initiated, installation-initiated (automatic), network operator-intiated, or application program-initiated logon request (these logon possibilities are discussed in Chapter 3). A 3270 logical unit-initiated logon request results from a terminal-operator action (pressing a function key or entering a logon message). A 3600 logical-unit logon request could be the result of either some terminal-operator action or could be initiated solely by the 3601 application program without involving a terminal operator (in either case, the actual request would be transmitted by the 3601 application program).

The following is a possible sequence of events that might occur prior to and during a 3600 logical unit-initiated logon request:

a. The 3601 and its logical units, defined to VTAM during VTAM definition, are made an active part of the VTAM network (perhaps by a network-operator VARY command).

b. As a result of receiving an activation request for the 3601 controller, VTAM sends an activate-physical-unit command to the 3601. The 3601 acknowledges the signal and responds that it is ready for operation. This is followed by an activate-logical-unit command to one of the logical units (work stations) associated with a particular 3601 controller.

c. The 3601 application program can either wait for a terminal operator at a 3601 device to indicate that the logical unit (work station) is to be logged on, or can issue a logon request on its own initiative. This is done by sending an Initiate-Self command to VTAM specifying the name of the VTAM application program with which the logical unit is to be connected. The initiate-self request unit may also specify a logon mode and a user-defined logon message. The logon mode is specified with the name of an entry in an installation-defined logon mode table. This entry contains a set of session parameters that is to be used when communicating with the logical unit. (If no logon mode is specified in the initiate-self, VTAM uses the first entry in the IBM-supplied default logon mode table. The set of session paraemters in this entry corresponds to an arbitrary definition of the communication rules most likely to be required for an interactive mode of operation.)

d. VTAM, receiving the Initiate-Self, schedules Sample Program 2's LOGON exit routine.

e. In the LOGON exit routine, after the validity of the logon request is confirmed, the logical unit is connected, by an OPNDST macro instruction. The OPNDST causes VTAM to send a bind control signal to the 3601. When VTAM receives a response to the bind signal, it issues a start-data-traffic (SDT) command to the logical unit (assuming SDT=SYSTEM is specified in the NIB used for connection). On receipt of a response to the SDT signal, the logical unit is connected to the application program, and the OPNDST is posted complete. (If SDT=APPL is specified in the NIB used for connection, the VTAM application program can itself send the initial SDT, using the SESSIONC macro instruction.)

   The OPNDST might be specified for synchronous or for asynchronous request handling. If the latter, the LOGON exit routine can either identify an ECB to be posted or an RPL exit routine to be scheduled as soon as the connection has been made.

The same RPL and NIB could be used for each logical unit being connected in the LOGON exit routine.

It may be desirable to write an initial message to the connected logical unit; this can be done from the LOGON exit routine or in an RPL exit routine following an asynchronously scheduled OPNDST.

So that the session parameters to be associated with the logical unit can be identified (which in this program would determine whether the 3600 or the 3270 I/O routines would be used with the logical unit), an INQUIRE macro instruction specifying OPTCD=SESSPARM can be issued. This option will allow the session parameters and logon data to be inspected and perhaps saved in the storage that is to be associated with the logical unit.

As soon as the logical unit is connected and control returned to VTAM, initial input for the logical unit can cause scheduling of the RPL exit routine, RPL1.

3 As soon as the first input is received from a connected logical unit, the operation started by the request to read input from any logical unit, issued in the initial routine, is completed. RPL1 is then scheduled and entered.

RPL1 can use the user field of the RPL of the request just completed to locate the logical unit-unique storage and the identity of the logical unit. On entry to RPL1, the address of the RPL is in register 1.

A CHECK macro is required to free the RPL for reuse; it also causes LERAD or SYNAD exit routines to be entered if any error occurs.

RPL1 posts an ECB so that subsequently, the wait routine in the main program can determine that the input has been received and can pass the input to the processing routine analyzer. The RPL1 exit routine moves input to the unique input-area for the logical unit. It then reissues a request to read input from any logical unit. Because a logical unit-related RPL obtained in the LOGON exit routine is used for subsequent I/O with any logical unit just read, the RPL causing entry to RPL1 can be continuously reused by the RECEIVE in RPL1. The operation is to be asynchronous with relation to the program, and RPL1 is to be reentered each time the request is completed. Note that the logical unit whose input caused entry to RPL1 is now in continue-specific (CS) mode. The RECEIVE would be coded identically to the RECEIVE in the initial routine.

Although not shown in Figure 5-27, the RPL exit routine would send a positive response to the message that caused it to be entered if a positive response were requested by the logical unit. If a response was to be sent only on an exception condition, a negative response would probably be sent in the SYNAD exit routine after a CHECK was issued.

An alternative to an RPL exit routine for the RECEIVE with OPTCD=ANY and related logic is to program this logic in the main program and post an ECB rather than schedule RPL exit routine. In Sample Program 2, one advantage to using an RPL exit routine is that input from a RECEIVE with OPTCD=ANY is handled sooner in an RPL exit routine than if an ECB were posted (which requires waiting until the next entry to the main program's wait routine). The first input of a new transaction or conversation receives preference over transactions or conversations already in progress.

4 The wait routine first determines whether an ECB has been posted, and if it has not, issues a WAIT macro on the list of ECBs. When a posted ECB is found, the RPL associated with it is located (perhaps by having the RPL located at a fixed

displacement from the ECB), and a CHECK macro instruction is issued. The CHECK clears the RPL for reuse for the next VTAM request for that logical unit and, if necessary, causes the LERAD or SYNAD exit routine to be entered. On return from CHECK, the feedback fields of the RPL contain information provided by VTAM; in addition, the LERAD or SYNAD routine may have indicated action to be taken. If the operation was successful, the wait routine branches to an address associated with the ECB. In the case of the first input of a transaction or conversation, this address is that of the processing routine analyzer.

5 The processing routine analyzer, which might consist of separate routines for different types of logical units, analyzes the input and branches to or calls the appropriate processor. This processor may be in a higher level language, such as COBOL or PL/1. (These processor routines are depicted in 2 of Figure 5-1).

6 The processing routine processes the input and prepares the output. This may require one or more disk I/O operations, which might be performed by calling a common disk I/O routine. When output is ready, or, in a conversation, when the next input is required, the processing routine requests VTAM I/O, causing control to pass to an appropriate telecommunications I/O routine.

7 The disk I/O routine requests a disk I/O operation asynchronously and uses the wait routine to wait for completion. This allows processing for other logical units to continue while a disk I/O operation for one logical unit is under way.

8 Although not shown, a processing routine might return to the next sequential instruction in the main program from which it was called; a branch would then be made to a common I/O routine, which would branch to a 3600 or a 3270 input or output routine. A special routine might be required to edit 3270 input and format 3270 output.

If the logical unit has 3600 session parameters, an input or an output operation is requested as appropriate. The handling of the request is specified as asynchronous; completion is determined when the ECB related to the logical unit is posted. Before issuing the request, the address to which the wait routine should branch (the return address is the processing routine) is placed in the ECB-related logical unit block.

If input is requested, the input, when it arrives, is not used to satisfy the outstanding RECEIVE request in RPL1 because the terminal is in CS mode.

If output is requested, the data may be sent in a chain of transmissions. This might be useful with output that would be passed from the 3601 application program to a 3610 printer. The 3601 application program could store all elements of the chain in a buffer until the entire chain was received (or print each element as it arrived). The VTAM application program would ensure arrival of the entire chain by receiving a single positive response sent back by the 3601 application program when the last element of the chain was received.

If the output completes a transaction or conversation, the terminal is reset to continue-any (CA) mode so that input that begins the next transaction or conversation satisfies the RECEIVE with OPTCD=ANY request that is issued in RPL1.

Details of the 3600 routines are provided in Figures 5-28 and 5-29 and in accompanying notes.

9 If the logical unit has 3270 session parameters, different I/O routines are required. The size of I/O areas required may be different. The range of input that may arrive may be wider. An additional requirement is to use brackets to control the overlap of input and output with 3270.

Details of the 3270 I/O routines are provided in Figure 5-30 and in accompanying notes.

0 The RESP exit routine is scheduled and entered when a response arrives from a logical unit. VTAM receives a response is received by VTAM because the VTAM application program requested it in the RESPOND operand of the SEND macro. When the response is received, the operation, only posted as scheduled in the 3270 or the 3600 I/O routine, is now complete and the RESP exit routine can now post the ECB. If the operation was successful, the response is positive; if an error occurred, a negative response will be indicated in the RPL. After copying the contents of the RPL (which is read-only) a CHECK macro is issued. The program's SYNAD exit routine analyzes the exception and takes possible action. The ECB is posted and control is returned to VTAM.

Details of the RESP exit routine are in Figure 5-31 and accompanying notes.

11 In Sample Program 2, two kinds of expedited-flow commands can be received from a 3601 application program: a request to stop sending to the logical unit at the end of the chain that is currently being sent (a quiesce-at-end-of-chain (QEC) command) and a request to reinitiate sending after previously being requested to stop (a release-quiesce (RELQ) command). This use of these commands may be desirable if an operator wants to interrupt a long series of printing so that keyboard input can be entered. After handling the operator request, the VTAM application program could resume printing. When either of these requests is received, VTAM schedules Sample Program 2's DFASY exit routine.

This exit routine does not apply to 3270 operation. Details of this routine are shown in Figure 5-32 and accompanying notes.

12 The TPEND exit routine is scheduled and entered as the result of a request from the network operator for an end-of-day (normal) or quick closedown of the VTAM network. In addition to other possible processing, the TPEND exit routine posts a special close ECB so that, subsequently, the main program's wait routine can branch to a CLOSE macro instruction in the main program.

13 The LERAD, SYNAD, and LOSTERM exit routines handle different categories of errors or unusual situations. The LERAD or SYNAD exit routine are entered as the result of a CHECK macro or a synchronous I/O request. The LOSTERM exit routine is scheduled asynchronously when certain situations occur, such as the immediate deactivation of a connected logical unit by the network operator. Like other parts of the program, the LOSTERM exit routine might branch to the LERAD or SYNAD exit routines for problem analysis. The LERAD exit routine primarily handles logical errors; it is most likely that these would occur during the debugging stages of the program. This exit routine might gather information, format it, and save it for programmer analysis after the program ends. The SYNAD exit routine primarily handles physical errors; it determines what general action should be taken (for example, retry, disconnection of the logical unit, termination of the program, or sending a message to the network operator) and either takes the action or passes an action code to the main program or other exit routine where the action is taken. The SYNAD routine may also want to record information related to situations that it handles for later problem analysis.

A number of error situations must be perceived and analyzed as the result of receiving a response from a logical unit; the response is analyzed following a RECEIVE with RTYPE=RESP specified or after a RESP exit routine is entered. Errors or special situations that result in exception responses cause the SYNAD exit routine to be entered when a CHECK macro is issued; the SYNAD can determine the cause of the exception response by analyzing sense information in the RPL and then take appropriate action.

The Logic of the 3600 I/O Routine

This routine is entered directly or indirectly (perhaps from a common I/O branching routine in the main program) as the result of a request for input or output with a specific logical unit with which a processor is currently engaged in a transaction or a conversation.

Figure 5-28 shows the logic of the 3600 I/O routine. The following notes are keyed to this figure.

1 If the processor's request is for input, the information that must be passed to VTAM is set up, and a RECEIVE is issued. The address of the logical unit's RPL is put in a register and the address of the input area associated with the logical unit and the length of the area are put in other registers. Since data is to be read, RTYPE=DFSYN is specified. The operation is to be asynchronous, and input is to be read only from the specific logical unit (whose CID is located in the RPL's ARG field). The ECB associated with the logical unit is specified for posting by VTAM when the operation completes. After issuing the RECEIVE request, register 15 contains 0 if the request is successfully accepted, or some other return code if it was not. If the request is accepted, the wait routine is returned to, after setting the next sequential instruction in this routine as the address of the instruction to be branched to when the ECB is posted.

Note: For simplicity, most checks of register 15 are not shown in Sample Program 2.

2 When data is received from the logical unit, VTAM posts the ECB. When the wait routine detects the posted ECB, it branches to the indicated location in the 3600 I/O routine. The RESPOND field of the RPL can be tested to determine whether the logical unit wants a positive response returned so that it knows positively that the input was received. If so, a SEND is issued, indicating that a response is to be sent to the logical unit (STYPE=RESP).

The SEND that sends the response, if requested, is scheduled synchronously. VTAM assumes POST=SCHED. Since no response can be returned to a response, once the request to send the response is accepted, the VTAM application program considers the sending of the response as complete.

If input arrives unsuccessfully or out of sequence (indicating that some input was lost), VTAM completes the VTAM application program's input request with an indication that input arrived for which an exception response must be returned; no input is forwarded to the program. The VTAM application program sends an exception response. The input request should be reissued.

Although now shown, the VTAM application program could also return an exception response to input that was successfully received. This might be done where such a response would have a meaning understood by both the VTAM application program and the logical unit. The USENSEO field of the RPL could be used to convey exception information.

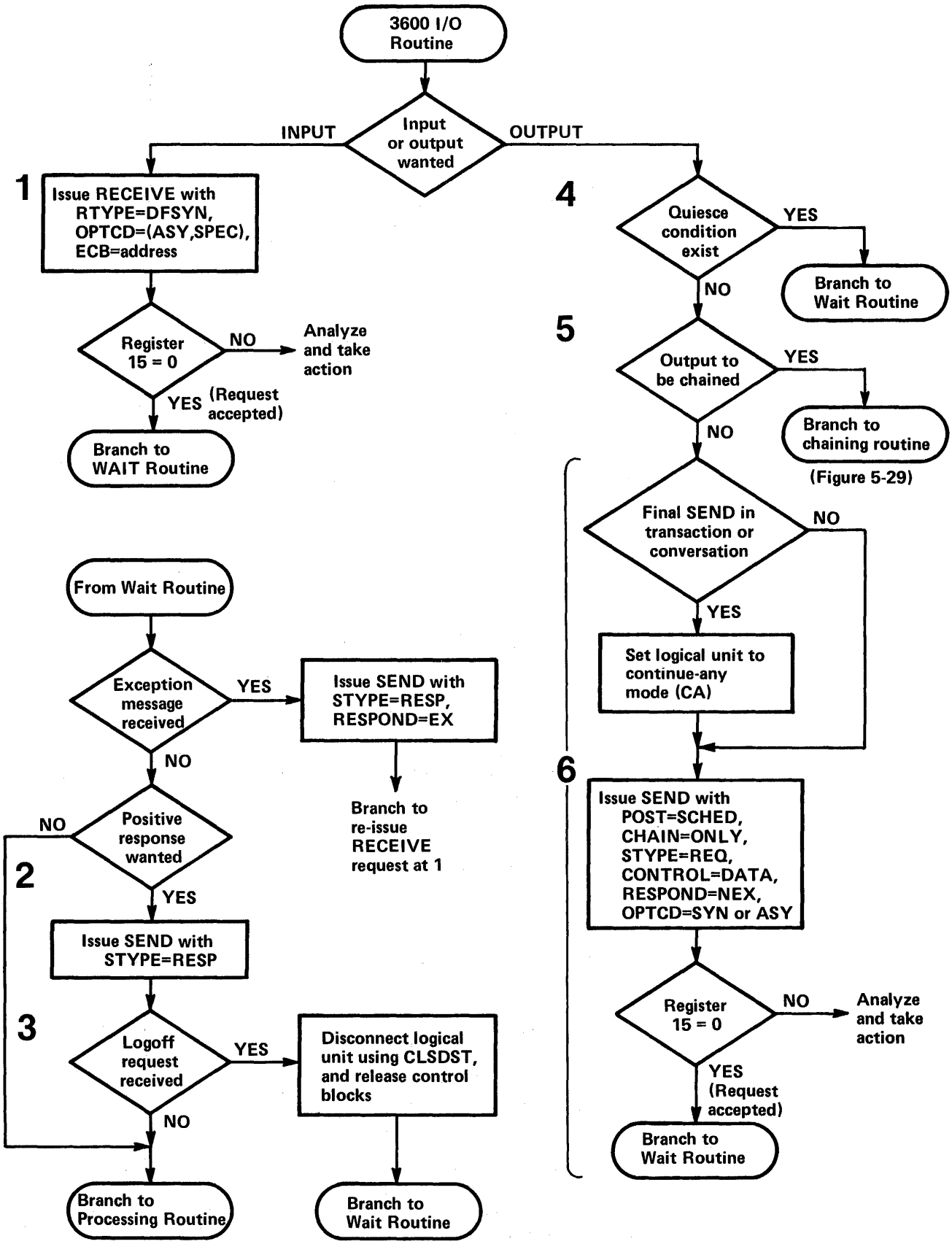


Figure 5-28. The Logic of the 3600 I/O Routine

3 If the input received contains a request to log off (to be disconnected from this program), the logical unit is disconnected via a CLSDST macro instruction, and the work areas associated with the logical unit are returned to the operating system or to a pool. Optionally, a message might be sent to the logical unit, confirming logoff, prior to issuing CLSDST.

The above description of the 3600 I/O routine assumes that a CHECK macro is issued in the wait routine upon completion of each requested input operation; if an error occurs, CHECK causes entry to the LERAD or SYNAD exit routine where appropriate analysis and action is taken. It might be noted that the input routine could issue a request to receive any kind of input: a normal-flow request unit (data or a command) (DFSYN), an expedited-flow command (DFASY), or responses (RESP). In this sample program, DFASY- and RESP-type input is handled by VTAM-scheduled DFASY and RESP exit routines, but the logic in these routines could have been branched to after determining in the wait routine or in the 3600 routine that DFASY or RESP information had been received. DFSYN means that either data or a normal-flow command can be received; although not shown in this example, normal-flow commands might be received in some applications, in which case, such commands would have to be responded to appropriately.

4 When an output request from a processor is received by the 3600 I/O routine, the I/O routine does not proceed with the request if the logical unit has quiesced the VTAM application program; instead, the I/O routine branches to the wait routine. The processor must wait for the quiesce to be released at which time the ECB for this logical unit is posted, a pending send request detected, and the I/O routine is reentered. (This logic is discussed in "The Logic of the DFASY Exit Routine.")

5 If the output request is chained to other output requests, a branch is made to a chaining routine (see Figure 5-29).

6 If output is not being chained, a SEND is issued that includes the operand CHAIN=ONLY. If the output completes a transaction or conversation, the logical unit is returned to continue-any mode; its next input satisfies the RECEIVE (with OPTCD=ANY) request issued in RPL1. The request can specify scheduling of the operation (POST=SCHED) with completion to be determined as the result of a positive or exception response (RESPOND=NEX) that causes scheduling of the RESP exit routine. (The RESP exit routine posts the ECB associated with the terminal, thus notifying the VTAM application program and the processor that the output request was completed.) The 3600 I/O routine then branches to the wait routine.

The Logic of the 3600 Chaining-Output Routine

Figure 5-29 shows the logic of the 3600 chaining output routine. The following notes are keyed to this figure.

1 Depending on the application, the number of elements in the chain might vary, or it might always be the same. Assuming that it varies in Sample Program 2, the number of chain elements must be determined so that the routine knows when to send the last element. This routine might be entered because a processor wants to send a report to an administrative line printer; this report may vary in length between 20 and 100 printer lines. Each line is sent to the logical unit as a chain element. The logical unit would determine how many lines (chain elements) it would collect before forwarding them to the administrative printer.

This chaining routine might be passed all of the data that it was to send in a chain or only part of it. In other words, the routine would not necessarily be sending an

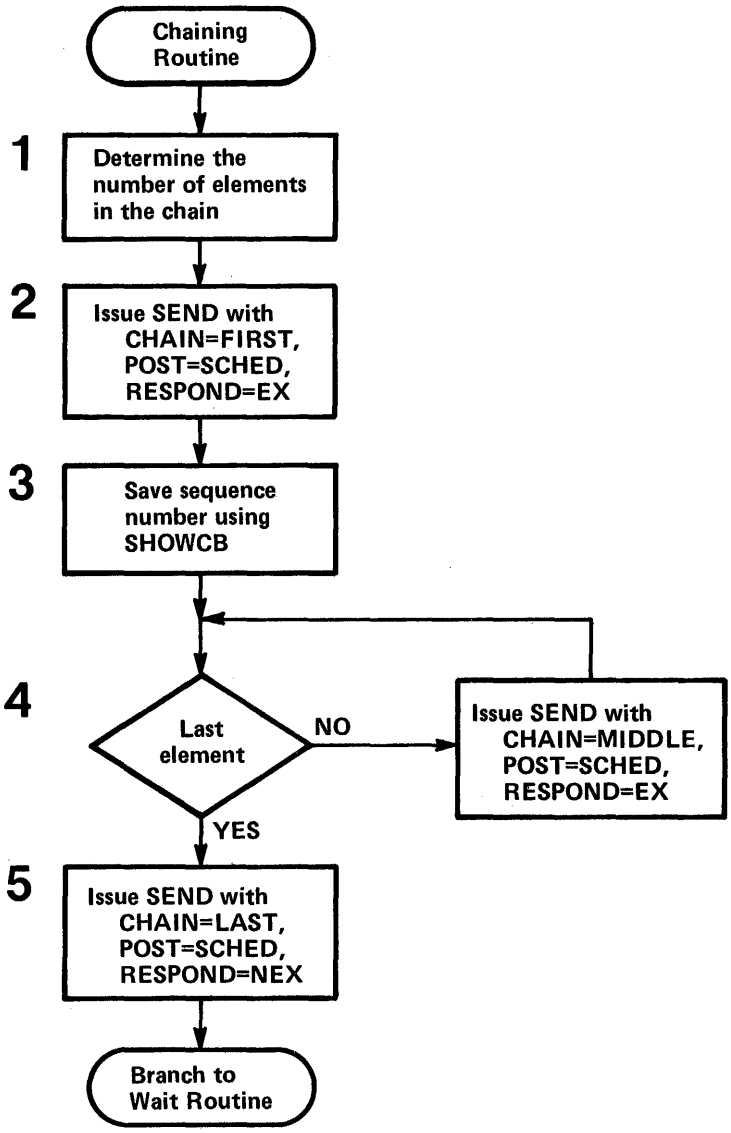


Figure 5-29. The Logic of the 3600 Chaining-Output Routine

entire chain each time it was entered. The logic discussed here, however, assumes all, or, in a retry situation, the last part of a chain is being sent.

2 Because one of the advantages of chaining output is reducing the number of required responses while still breaking output into segments that can be interspersed on the communication line, all SEND macros other than the last one specify that a response is to be returned only if an exception is noted (RESPOND=EX). When the last segment is received, a positive response is returned, and the VTAM application program recognizes that the entire chain arrived successfully. When RESPOND=EX is specified, the scheduling of output (POST=SCHED) is assumed by VTAM; it does not have to be specified.

3 In some cases, it may be necessary to save the sequence number of the first element sent in a chain. This number is available as soon as sending has been scheduled. It can be obtained from the SEQNO field of the RPL by using the SHOWCB macro. In case all or part of the chain must be resent (a negative response arrives in the RESP exit routine), the first-in-chain sequence number may be useful

in determining where to start resending from. It may also be necessary (not shown here) to reset the beginning sequence number for the logical unit that is receiving the chain; this number would be sent to the logical unit by using a SESSIONC macro. The sequence number can be saved in the work area associated with the logical unit.

4 All elements except the first and last are middle elements (CHAIN=MIDDLE).

5 For the last element in the chain, the SEND macro must identify it as the last (CHAIN=LAST) and request the return of a response (RESPOND=NEX). Either POST=SCHED or POST=RESP can be specified.

When the response is received, if POST=SCHED is specified, the RESP exit routine is scheduled; it posts an ECB and the wait routine returns to the processor that originated the output request. If POST=RESP is specified, VTAM posts an ECB or schedules an RPL exit routine as appropriate.

The Logic of the 3270 I/O Routine

Figure 5-30 shows the logic of Sample Program 2's 3270 I/O routine. With few exceptions, the VTAM application program using record-mode (SNA) macro instructions need not distinguish between locally attached, BSC, and SDLC 3270s. Data received from a 3270 begins with an AID (Attention Identifier) character. Data sent to the 3270, whether local or remote, must begin with a 3270 command character (for example, an Erase, Erase and Write, or Erase All Unprotected character); VTAM inserts an ESC character for the BSC 3270. The 3270 is different from other logical units in the following ways:

- A 3270 cannot send SNA command requests or responses. However, in some cases, VTAM will provide responses to the VTAM application program on behalf of the 3270 as a result of receiving BSC or SDLC responses to transmitted data or as a result of receiving indications that the 3270 is in a particular bracket state.
- The amount of data that can be sent to or received from the 3270 is limited by the physical characteristics of the 3270, whereas the amount of data that can be sent to or received from a logical unit is more indefinite.
- Chaining output to the 3270 is not possible.
- Responses cannot be requested by 3270 logical units.

The following notes are keyed to Figure 5-30.

1 Except that the type and length of data may be different for a 3270 RECEIVE, this request is similar to 1 discussed for the 3600 I/O routine.

2 This logic is similar to that of 2 for the 3600 I/O routine. However, because the 3270 cannot request that a response be sent to input it has provided, no check is made to determine whether to send a response.

3 If 3270 output is requested by a processing routine, the 3270 I/O routine determines whether this output completes a transaction or a conversation. If it does, the 3270 logical unit is put back into continue-any mode so that the RECEIVE (with OPTCD=ANY) specified in the RPL1 exit routine can receive input from this logical unit when the terminal operator wishes to begin a new transaction or conversation.

4 A SEND macro instruction is issued to send the output. (Although not shown, this routine may also have to determine from the processing-routine request what 3270 command character—for example, Erase and Write—is to precede the output data

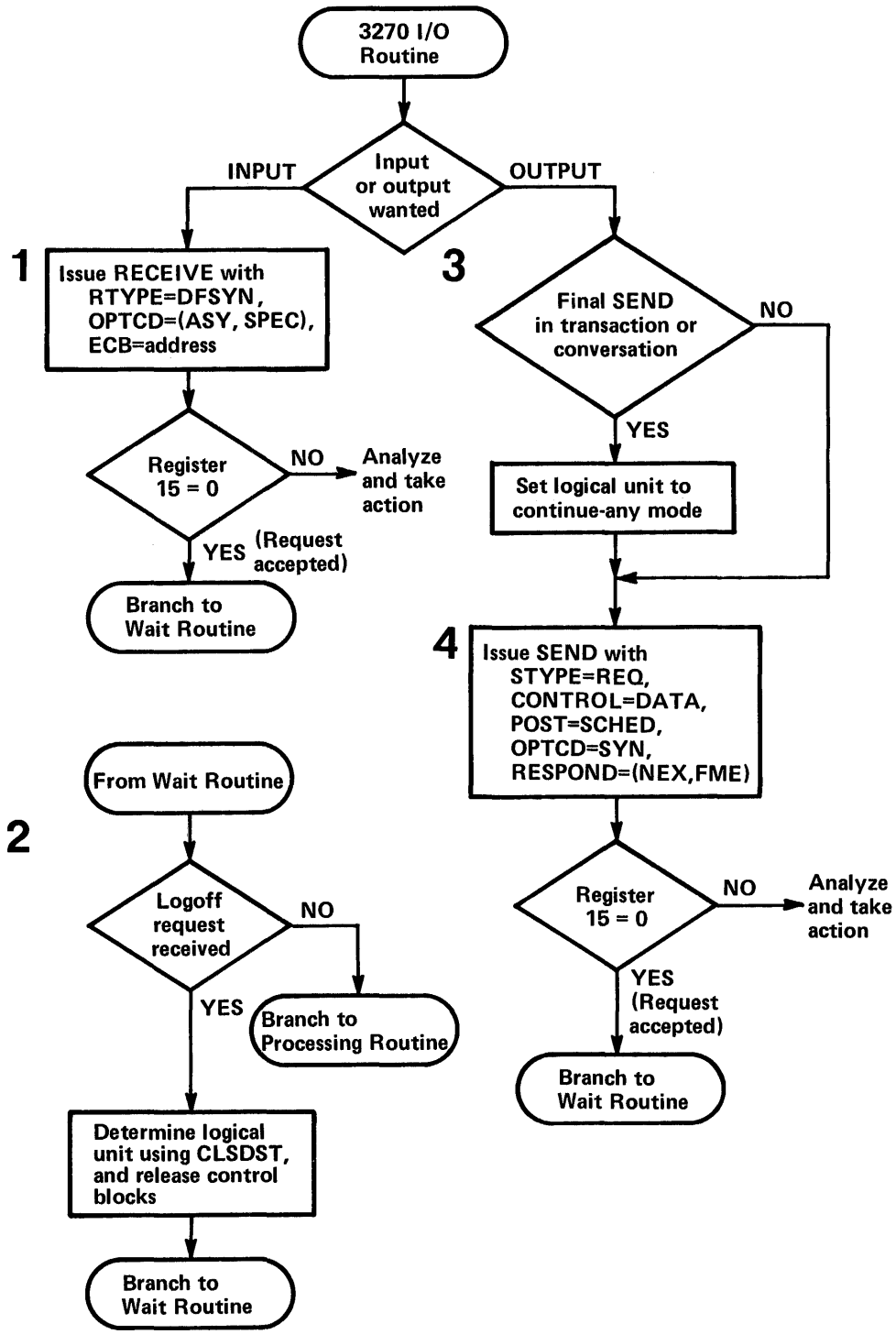


Figure 5-30. The Logic of the 3270 I/O Routine

stream that the processing routine furnishes.) The sending of the output is scheduled synchronously (POST=SCHED, OPTCD=SYN); VTAM returns control after it has scheduled the output operation. A response is requested (RESPOND=(NEX,FME)) so that the VTAM application program can determine whether or not the operation was successful. The 3270 returns information enabling VTAM to provide the appropriate response in the RPL and to schedule the RESP exit routine. The RESP exit routine (Figure 5-31) posts an ECB so that the main program's wait routine can determine that the operation completed branching back to the processing routine that requested the output. (Note that an output request to a 3270 printer requires a definite response (RESPOND=(NEX,FME)); an output request to a display can specify either NEX or EX but cannot specify that either a positive or negative response is to be returned (RESPOND=(NEX,NFME)).

After successfully scheduling output to the 3270, the 3270 I/O routine branches to the wait routine.

The Logic of the RESP Exit Routine

Figure 5-31 shows the logic of the RESP exit routine. This routine is entered when the response is received to an output request that has POST=SCHED specified in a 3600 or 3270 I/O routine. The output operations have been scheduled with responses to be returned by the terminal so that completion of each operation can be determined. (It would also have been possible for all output operations to be specified POST=RESP; in this case, the response would be received by VTAM and its nature determined by the VTAM application program after ECB posting or RPL exit scheduling.)

Note that when the VTAM application program gets control in its RESP exit routine, a RECEIVE is not issued. The nature of the response is determined by examining the RESPOND and other fields of an RPL that is in VTAM's storage (and therefore is read-only). The address of this RPL is pointed to in a parameter list whose address is in register 1 when the RESP exit routine is entered. The logical unit control area (the ECB, the RPL associated with the logical unit, and the logical unit-associated work area) can be located by the address in the USER field of the VTAM RPL. (It contains whatever was placed in the USERFLD field of the NIB when the logical unit was connected.)

The following notes are keyed to Figure 5-31.

1 If the response was positive, the appropriate ECB is posted and a return is made to VTAM. Even if other action must be taken because the response is negative, the ECB is posted so that the wait routine can determine that the operation has been completed.

2 The RESP exit routine may wish to use the SYNAD exit routine to analyze a negative response; if so, the user could set up the appropriate registers and branch directly to his SYNAD exit routine.

3 If the situation was defined by the SYNAD exit routine as recoverable, the operation would be retried. If it were part of a 3600 chaining operation, it might be necessary to save the sequence number of the output to which an exception response was returned so that the chaining routine could determine the sequence number at which it should start a retry. If a logical unit's inbound sequence number must be reset, a SESSIONC using the STSN operand could be used to synchronize sequence numbers. (This logic could also be in the SYNAD exit routine.)

On completion, the RESP exit routine returns control to VTAM.

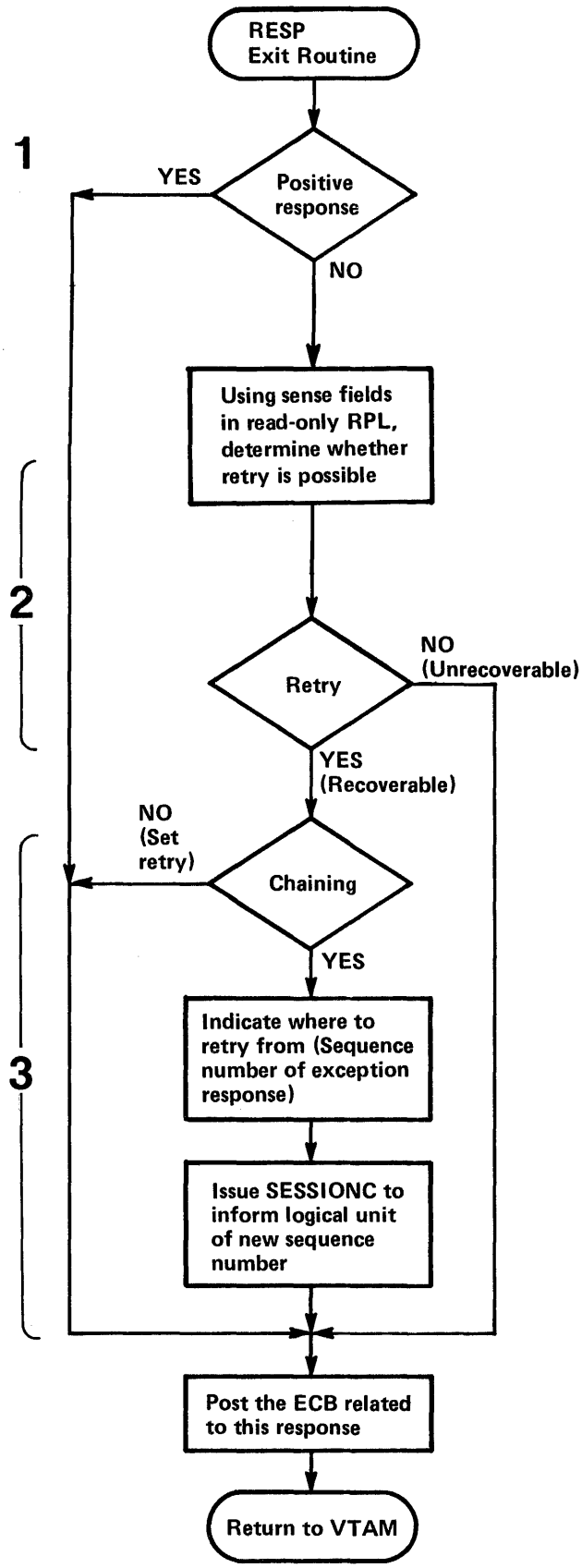


Figure 5-31. The Logic of the RESP Exit Routine

The Logic of the DFASY Exit Routine

Figure 5-32 shows the logic of the DFASY exit routine in Sample Program 2. The DFASY exit routine is entered when a request is received from the logical unit asking the program to quiesce the sending of data to the logical unit or to resume sending it, if sending was previously quiesced.

Quiescing can be done for two reasons:

- To ensure that, at a given time, only the logical unit or the VTAM application program can send. This use of quiescing is not demonstrated in this sample program. (Quiescing is only one means available to ensure that both sides do not send at the same time. Change-direction indicators may also be used. In many cases, dependence on one end of the communication link receiving an answer to a previous output is sufficient to ensure that both ends do not send at the same time.)
- To interrupt a steady flow of data input so that an output operation can be performed. This is the use of quiescing that is demonstrated here. For example, a teller at a 3600 device may wish to temporarily interrupt a long printout so that an inquiry can be submitted to the VTAM application program. As a result of a teller action, the logical unit for the teller's device sends a quiesce command to the VTAM application program, which can then agree to hold its sending in abeyance and be ready to read input from the logical unit.

The quiesce-at-end-of-chain and release-quiesce commands are sent as expedited-flow commands. VTAM schedules the VTAM application program's DFASY exit routine when one of these expedited-flow commands is received.

The following notes are keyed to Figure 5-32.

1 The type of command that caused the DFASY exit routine to be entered is available in the CONTROL field of the read-only RPL whose address is provided by VTAM on entry. If a quiesce-at-end-of-chain (QEC) command has been received, this routine sets a bit that prevents the sending of messages in the work area associated with the logical unit. The work area, as in the RESP exit routine, is located by the address in the user field of the RPL.

2 If the quiesce is to be immediate, all scheduled but incomplete output operations can be canceled by clearing all existing data flow with a SESSIONC macro instruction that specifies CONTROL=CLEAR followed by a SESSIONC that specifies CONTROL=SDT. If in the middle of a chain and not all of the chain is to be present, it may be necessary to determine where sending is to resume when the quiesce condition is released.

3 The QEC command is acknowledged by sending back a quiesce-completed (QC) command. The command is specified symbolically (CONTROL=QC) by the VTAM application program. So that the logical unit's normal-flow input (RTYPE=DFSYN) can complete the request to receive input from any logical unit, being recurrently issued in the RPL1 exit routine, the logical unit is put back into continue-any mode (OPTCD=CA). In sending other than data, a response should not be requested; VTAM does not post the operation complete until it has received a response (in other words, POST=RESP and RESPOND=(NEX,FME) are assumed).

4 This flag may be required in addition to the hold-from-sending flag to determine where to resume sending. See **2** above.

5 If the command was a release-quiesce (RELQ), the hold-from-sending flag is turned off.

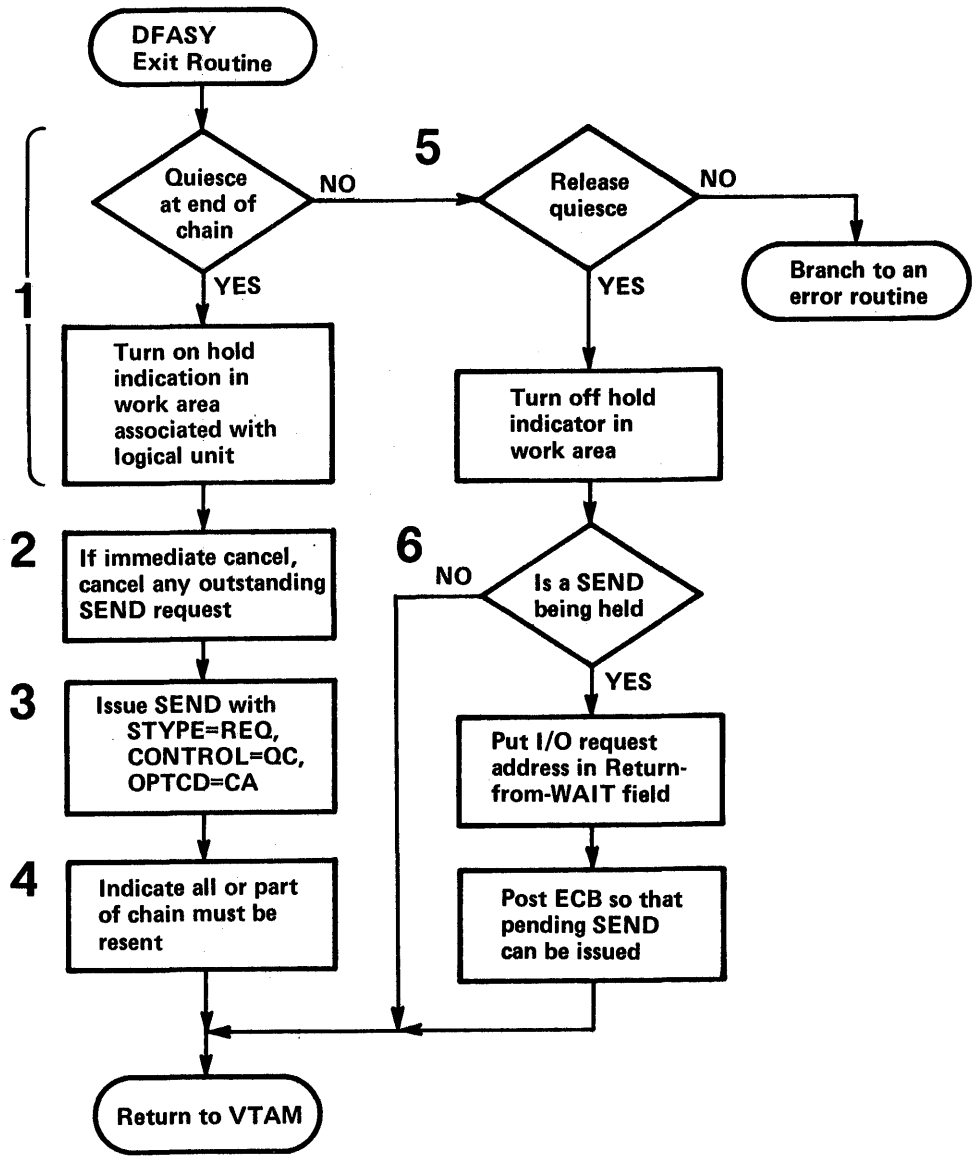


Figure 5-32. The Logic of the DFASY Exit Routine

6 If further output is being held, the address of the output request routine is rescheduled for this logical unit and an ECB is posted so that the wait routine can branch to it. Control is returned to VTAM.

Choosing Programming Alternatives Discussed in the Sample Programs

The following is a summary of some alternatives to programming techniques discussed in the sample programs. In general, an installation must decide whether programming convenience or efficiency is more important. To achieve maximum efficiency, it may be necessary to "tune" a program; for example, to vary the number of consecutive RECEIVE with OPTCD=ANYs to determine the effect on response time.

- A RECEIVE with OPTCD=ANY that specifies ECB posting can be more efficient than a RECEIVE with OPTCD=ANY that specifies the scheduling of an RPL exit routine (which might then post an ECB). However, the RPL exit routine provides some programming convenience and allows handling of an event's completion sooner than does the posting of an ECB by VTAM (which the program may not immediately detect).

- The use of more than one RECEIVE with OPTCD=ANY specified can be more efficient than a single RECEIVE with OPTCD=ANY. During the time between a single RECEIVE with OPTCD=ANY completing and being reissued, VTAM sends any input received to a pageable buffer from which the input must be retrieved when the RECEIVE is reissued. Multiple RECEIVEs require multiple RPLs and input areas, however.
- If a positive response is required to confirm the arrival of an output message, a SEND with POST=RESP specified is more efficient than the combination of a SEND with POST=SCHED and the scheduling of a RESP exit routine or use of a RECEIVE RTYPE=RESP.
- A SEND that specifies ECB posting can be more efficient than a SEND that specifies the scheduling of an RPL exit routine (in which an ECB may be posted). However, the latter may be easier to program.

Using Authorized Path in OS/VS2

In OS/VS2, a VTAM application program can specify execution of individual SEND, RECEIVE, and RESETSR macro instructions in a path that requires fewer instructions. This facility, called "authorized path," can be used to improve performance in a VTAM application program. VTAM authorized path can be used with any devices supported using record mode.

To use authorized path, the program must be authorized and in the supervisor state. *OS/VS2 System Programming Library: Supervisor*, GC28-0628, describes how to specify an authorized program. The MODESET macro instruction can be used to put the program into supervisor state.

The VTAM application program can use authorized path while running under a TCB (task control block) or while running under an SRB (service request block). To use it while running under a TCB, the authorized program, having put itself into supervisor state, specifies BRANCH=YES on any SEND, RECEIVE, or RESETSR that is to be executed using authorized path. (Subsequently, to issue any macro instruction that does not use authorized path and that uses the same RPL, the RPLBRANC flag in the RPL must be turned off either by 1) coding BRANCH=NO on a MODCB macro instruction, 2) referring to the field by using the IBM-supplied DSECT and turning it off with an assembler language instruction, or 3) by coding BRANCH=NO on the subsequent RPL-based macro instruction that does not use authorized path.)

Authorized path is always used when SEND, RECEIVE, or RESETSR is issued under control of an SRB. One way to use authorized path under an SRB is for the authorized program, while running under a TCB, to specify an RPL exit routine when issuing a SEND, RECEIVE, or RESETSR that specifies BRANCH=YES. On entry to the RPL exit routine, the program will be running under an SRB. Any SEND, RECEIVE, or RESETSR issued in this environment is automatically executed using the authorized path; BRANCH=YES need not be specified. An alternative way to create the SRB environment is to use the SCHEDULE macro instruction. No RPL-based macro instruction other than SEND, RECEIVE, RESETSR, and CHECK should be issued while running under an SRB.

An alternative way to run under an SRB is to first build an SRB and then issue a SCHEDULE macro instruction.

Figure 5-33 illustrates the logical requirements for using authorized path when running under a TCB and under an SRB. The logic associated with input/output requests in an actual program similar to the one outlined previously in Sample Program 2 would probably be more complex. The following notes are keyed to the numbers in Figure 5-33.

1 The application program begins processing as a task in OS/VS2, running under the control of a TCB (task control block). As part of normal VTAM processing, it issues an OPEN macro instruction, to open an ACB (access method control block). The OPEN might be coded like this:

```
        OPEN    AUTHACB
```

In this sample program, AUTHACB contains:

```
AUTHACB     ACB     AM=VTAM,APPLID=APPL5ID,PASSWD=APPL5ID
```

2 Next, the application program issues an OPNDST macro instruction to connect the application to logical unit. The OPNDST might be coded:

```
        OPNDST  RPL=AUTHRPL,OPTCD=SYN
```

The RPL, named AUTHRPL, contains the rest of the information needed for the OPNDST.

3 Now the application program uses the MODESET macro instruction to change into the supervisor mode. This is coded:

```
MODESET     MODE=SUP
```

4 The RECEIVE macro instruction conforms to the coding rules for authorized path running under the control of a TCB. The BRANCH=YES operand is specified. The RECEIVE macro instruction might be coded:

```
RECEIVE     RPL=AUTHRPL,RTYPE=DFSYN,AREA=INPUT00,
            AREALEN=100,OPTCD=(ASY,ANY,CS),EXIT=AUTHEXIT,
            BRANCH=YES
```

5 Control now goes to the RPL exit routine named AUTHEXIT. Note that this exit routine runs under the control of an SRB and receives different parameter list than an RPL exit routine running under a TCB. On entry:

- Register 1 contains the address of the RPL.
- Register 13 *does not* contain a save area address since none is provided (this is also true of an RPL exit routine under a TCB).
- Register 14 contains the address of the OS/VS2 dispatcher.
- Register 15 contains the entry point address of the exit routine.

The CHECK macro instruction frees the RPL for re-use and causes entry to a LERAD or SYNAD exit routine if required. The CHECK macro instruction is the only VTAM macro instruction other than SEND, RECEIVE and RESETSR that can be issued under control of an SRB. Any other VTAM macro instruction will fail. The CHECK is coded:

```
CHECK       RPL=AUTHRPL
```

6 This exit routine runs under the control of an SRB since it is an exit routine entered from a macro instruction using authorized path. The SEND macro instruction therefore uses authorized path. In this sample, the SEND looks like this:

```
SEND        RPL=AUTHRPL,OPTCD=(SYN,CA),CONTROL=DATA,
            STYPE=REQ,RTYPE=DFSYN,RECLEN=95,AREA=OUTPUT00,
            POST=SCHED,RESPOND=(NEX,NFME,NRRN)
```

This example specifies that no response is to be returned, which assumes that failure of the message to arrive will be detected by analyzing terminal operator input.

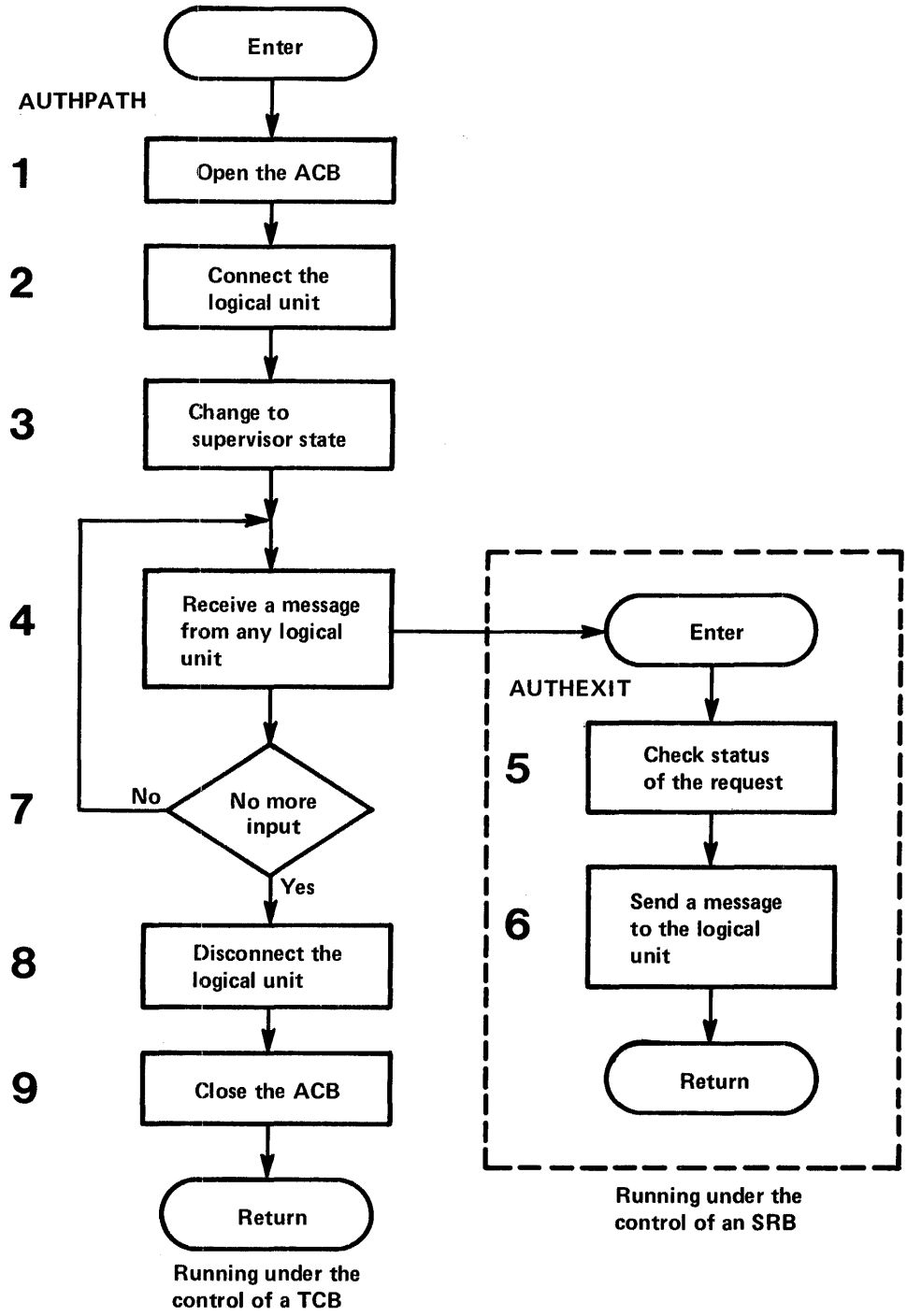


Figure 5-33. The Logical Requirements for Using Authorized Path (OS/VS2)

The OPTCD= operand in the sample specifies SYN to indicate synchronous processing.

When the exit routine completes its processing, control returns to the mainline code, AUTHPATH.

7 AUTHPATH determines whether or not all the I/O processing has completed. If more I/O must be performed, control goes through the RECEIVE/SEND loop described by steps 4-7. If all I/O processing has completed, control goes to step 8 to disconnect the logical unit.

8 The sample program disconnects the logical unit by using the CLSDST macro instruction. The CLSDST might be coded:

```
CLSDST          RPL=AUTHRPL,BRANCH=NO,OPTCD=SYN
```

The BRANCH=NO operand turns off the RPLBRANC flag that was turned on by the BRANCH=YES operand. This must be performed for the correct execution of the CLSDST macro instruction. Since there is no authorized path for this macro instruction, the flag cannot be on when CLSDST executes.

9 The CLOSE macro instruction closes the ACB.

CHAPTER 6. RELIABILITY, AVAILABILITY, SERVICEABILITY

This chapter describes what VTAM does to detect errors, to attempt to recover from them, and—when recovery is impossible—to record the conditions under which the error occurred. The chapter also describes VTAM debugging aids, dump programs, and test programs to correct programming errors and machine malfunctions.

VTAM's RAS Strategy

VTAM's reliability, availability, and serviceability (RAS) aids are intended to minimize the impact of programming errors and machine malfunctions on the telecommunication network. VTAM attempts to detect problems and handle them before they become serious. If an error is encountered, VTAM attempts recovery while recording the error for possible future maintenance. If recovery is not possible, VTAM automatically provides error records and dumps to help identify the problem and its cause. For additional debugging aid, the installation can request traces and testing services from VTAM. If possible, VTAM avoids terminating application programs or closing down the telecommunication system when an error occurs; instead, it tries to isolate the problem and either correct the error itself or provide information to enable the user to correct it. VTAM handles both hardware and software errors; software errors include those on the part of users (application programs and the network operator), of the operating system, and of VTAM itself.

Error-recovery and error-recording operations are invoked automatically. VTAM requires no action on the part of the installation to invoke them, although the error records could be used to perform maintenance on elements in the system that are causing frequent temporary, but recoverable, errors.

When an error is permanent, a message is usually sent to the network-operator's console. The message and its explanation define the condition, indicate the probable cause, and suggest a course of action. For the network operator, this action may involve circumventing the problem as well as collecting data for later debugging. To get adequate material for problem determination, the network operator does one or more of the following things:

- Saves all console logs that pertain to the error message. These logs probably reflect all of VTAM's actions from VTAM startup through the issuing of the message.
- Obtains printouts of any requested traces performed in connection with the error.
- Saves any dumps that may have resulted from the errors.
- Requests and saves status displays of the nodes involved in the error.
- Initiates the Teleprocessing Online Test Executive Program (TOLTEP) for testing devices and lines involved in the error.

As a temporary solution to a problem, the network operator may be able to circumvent it or avoid it by disabling part or all of the telecommunication system. In any case, to assist in later correction of the problem, the installation should keep a complete description of the network as it existed at the time of the error.

Correcting the problem involves identifying the cause and then making the necessary correction. The following steps may help identify the problem:

1. Studying the message log: The sequence of messages leading up to the error as well as the error message itself may identify the problem.

2. Examining error records: Error records pertaining to VTAM or its telecommunication system may identify the problem.
3. Examining dumps and traces: Dumps and traces can be used to identify the area in which the error occurred or to find the problem itself.
4. Recreating the problem: VTAM's network-operator commands and RAS facilities can help recreate the problem and to collect pertinent data.

VTAM's RAS Facilities

VTAM's RAS facilities can be grouped into those related to serviceability and those related to reliability and availability of the system. The following sections detail VTAM's RAS facilities.

Serviceability Aids

VTAM's serviceability aids assist in determining the cause of problems in the telecommunication system. Serviceability aids include:

- Error Recording

  Hardware Errors: VTAM's hardware-error recording augments the error recording of the operating system. Both permanent and temporary errors are recorded. In addition, messages are sent to the network operator when any permanent error occurs.

  Software Errors (OS/VS); VTAM uses SYS1.LOGREC to retain records of software errors that result in the abnormal termination of VTAM.

- Traces

  DOS/VS: The problem determination and serviceability aids (PDAIDS) can be used to monitor VTAM's SVCs, I/O operations, and internal storage management.

  OS/VS: The generalized trace facility (GTF) can be used to monitor VTAM's SVCs, I/O operations, and task management and to trace events in application programs using VTAM.

  VTAM: Internal VTAM traces can be used to monitor I/O activity, and to record contents of VTAM buffers, and in OS/VS, to trace VTAM storage management.

- Dumps

  OS/VS: System dump programs are tailored by VTAM to provide formatted dumps of some VTAM control blocks and in some cases, to provide additional diagnostic information.

  NCP: VTAM has tailored the NCP dump utility program. This utility program can be invoked using VTAM's MODIFY command, or, optionally, it can be invoked as part of VTAM's error recovery procedures for communications controllers. The dump provided is of the NCP in the communications controller.

- TOLTEP

  The Teleprocessing Online Test Executive Program (TOLTEP) provides online testing for telecommunication lines and certain devices in a VTAM system.

These aids are discussed in detail below.

Error Recording

This section describes VTAM's facilities for recording errors encountered in the telecommunication system. Hardware error recording and software error recording are discussed separately.

Recording Hardware Errors: Hardware error recording is a function of the error recovery procedures (ERPs). (See "Reliability and Availability Support," later in this chapter, for a

description of ERP processing.) The data on hardware errors collected by VTAM is placed in the error-record data set of the operating system. The information in this data set can be formatted and printed by each system's Environmental Recording, Editing and Printing (EREP) program.

VTAM's hardware error recording is an extension of the OS/VS Outboard Recorder and the DOS/VS Recovery Management Support Recorder.

In addition to recording errors, VTAM maintains two counters for each locally attached device in the network. One counter keeps track of the number of SIO commands issued for the device; the other keeps track of the number of temporary errors for the device. Counters are also kept in the device statistic tables that the operating system maintains for each locally attached device. These counters indicate the number of each type of error encountered for each device. The tallies in the counters are included in any records written to the error-record data set.

Recording occurs when any of the following conditions is encountered:

Permanent hardware error: This is an error for which recovery could not be made, either because the error is undefined or because attempts by ERPs to correct the problem were unsuccessful.

Counter overflow: A record is written whenever any of the counters is about to overflow.

End-of-day: A record is written whenever VTAM deactivates a device.

Records for permanent errors include the name and address of the failing device, the time and date of the failure, the contents of the counters at the time of the failure, the failing channel command word, the channel status word, sense information, and device flags.

Records for counter overflow and end-of-day conditions include the name and address of the associated device, the time and date that the event occurred, and the contents of the counter.

Recording Software Errors: In OS/VS systems, if VTAM cannot recover from a software error and must terminate processing, VTAM collects data on the error. This data is formatted and written to the system data set SYS1.LOGREC. Records on this data set can be formatted and printed by the OS/VS EREP program.

A VTAM software error record includes a description of the error and audit-trail information. A VTAM audit trail is a record of the modules entered during the processing in which the error occurred. (An audit of module activity is maintained for those areas of VTAM responsible for processing telecommunication requests.) The audit information in the software error-record identifies the module executing when the error was detected, as well as the modules that were entered prior to the error.

Traces

VTAM under both DOS/VS and OS/VS has:

- Operating-system traces
- VTAM traces (including NCP-provided traces)

The operating-system traces provide the highest level of problem determination. By using the appropriate system trace, a problem may be isolated to a particular program or operating-system component. If the problem is isolated to a component such as VTAM or to an application program using VTAM, the VTAM traces can then be used to help locate and identify the area in which the error is occurring.

Operating-system traces and VTAM traces are each discussed in more detail below.

Operating-System Traces: The system traces can be used to monitor the SVC and the I/O activities of VTAM. The data collected by these traces can be used as a chronology of VTAM activity and reflects the conditions in VTAM during errors. System traces for VTAM are started and stopped by using the facilities of the operating system.

DOS/VS trace records for VTAM are printed with the PDLIST program. OS/VS1 trace records for VTAM are printed with the operating system's HMDPRDMP service aid. Refer to the publication *OS/VS1 Service Aids,* GC28-0665, for informationon HMDPRDMP. OS/VS2 trace records for VTAM are printed using the operating system's AMDPRDMP service aid. Refer to the publication *OS/VS2 System Programming Library: Service Aids,* GC28-0633, for information on AMDPRDMP.

VTAM Traces: In addition to the operating-system traces, the following types of VTAM traces are provided:

I/O traces: To trace I/O activity within VTAM.

Buffer traces: To record the contents of VTAM buffers as data enters and leaves VTAM.

NCP line traces: To record the NCP trace records of specified line.

Storage management traces: To record the use of VTAM storage.

VTAM traces are initiated and terminated by VTAM start options or the MODIFY command. See "VTAM Commands," in Chapter 4, for information on starting and stopping VTAM traces.

The trace records generated by VTAM traces are written on a particular data set in each operating system. For DOS/VS, the file name of this data set is TRFILE. For DOS/VS, VTAM writes the VTAM trace records for all traces except the storage management traces. The system PDAIDs need only be active when the storage management traces are used. For OS/VS, VTAM passes the trace data to GTF, which records the records in SYS1.TRACE. Because VTAM traces use GTF to record the trace records, GTF must be active before VTAM data can be recorded. If GTF is not active, VTAM does not provide a record of the event to be traced.

To print VTAM trace records, VTAM provides a utility program for DOS/VS and uses a system utility program for OS/VS. The VTAM trace-print utility program for DOS/VS selectively edits and prints the data collected by the various VTAM traces. Using the MODIFY command, the network operator starts the print utility program and specifies the nodes for which trace records are to be printed. In OS/VS1, VTAM traces are printed by using the operating sytem's HMDPRDMP service aid. In OS/VS2, VTAM traces are printed by using the operating system's AMDPRDMP service aid.

Refer to the publication *DOS/VS Serviceability Aids and Debugging Procedures,* GC33-5380, for more information on the PDAIDs and the PDLIST Program. Refer to the publication *OS/VS1 Service Aids,* GC28-0665, for more information on GTF and the HMDPRDMP service aid. Refer to the publication *OS/VS2 System Programming Library: Service Aids,* GC28-0633, for more information on GTF and the AMDPRDMP service aid.

Dumps

VTAM provides routines to augment both the operating system dump facilities and the NCP dump facilities. The text below describes what these routines do.

Operating System Dumps: VTAM provides routines to format portions of OS/VS dumps. When the operating system is dumping VTAM or an application program that is using VTAM, it invokes these routines to locate and format key VTAM control blocks. For

each control block that is formatted, its name and address is printed, followed by the name and contents of the pertinent fields. A hexadecimal dump of the control block is printed following the formatted printout. If VTAM finds an invalid field (either in the control block or in the chain of pointers to the control block), the condition is noted in the dump.

This formatting is performed automatically for dumps resulting from abnormal termination (ABEND) of VTAM itself, or from the abnormal termination of an application program that is using VTAM, or from a SNAP macro instruction in an application program. For dumps produced by the PRDMP service aid, however, VTAM formatting is optional; the option must be specified as part of the input to the service aid.

Dumps of VTAM include audit trail information and unique identifiers for each VTAM module and control block. The audit trail information is like that recorded for software information.

NCP Dumps: VTAM uses the NCP dump program to dump the contents of communications controllers. An NCP dump is taken automatically if the NCP fails and an automatic dump was specified as part of the generation of that NCP. If an NCP fails and an automatic dump was not specified, the network operator is notified of the failure and is given the option of requesting a dump of the NCP. Other than during error recovery, the network operator can also request a dump of an NCP by using the MODIFY command, as described in "Starting and Stopping VTAM Facilities," in Chapter 4.

Teleprocessing Online Test Executive Program (TOLTEP)

TOLTEP is a VTAM component that controls the selection, loading, and execution of Online Tests (OLTs) within the telecommunication network. These OLTs are specific device tests designed to be used to diagnose hardware problems and to verify the reliability of a device in the telecommunication network.

TOLTEP allows multiple OLTs to be run concurrently with processing application programs in the VTAM telecommunication system. TOLTEP supports testing of devices attached on start-stop or BSC lines, devices in the 3270 family, SDLC lines, and the 3767 and 3770 family. In OS/VS, TOLTEP can also be used to dynamically modify its configuration data sets (CDS files).

TOLTEP is included automatically in the system when VTAM is generated. It requires that the OLT option in the NCP be specified during NCP generation. TOLTEP is automatically initialized and made ready to accept test requests when VTAM is started. If TOLTEP is abnormally terminated prior to VTAM termination, VTAM automatically attempts to restart it.

To run TOLTEP, a terminal must be available for use as a control terminal. This terminal could be the terminal to be tested, although this is usually unwise because it may be impossible to enter test data or receive test results at the terminal. The control terminal must be able to enter and receive alphameric characters. An alternate printer can be designated to receive hard-copy output from the test. (The alternate pointer is required if the control terminal is also the terminal being tested.)

TOLTEP can be invoked either from the network operator's console or from a network terminal. As explained in "Starting and Stopping VTAM Facilities," in Chapter 4, TOLTEP can be invoked from the console by an option of the MODIFY command or by the VARY command to log a specific terminal in the network onto TOLTEP.

If a terminal is being monitored by the network solicitor (and is, therefore, not connected to an application program), a terminal-initiated logon can be used to log the terminal on to TOLTEP.

If a terminal to be used by TOLTEP is currently connected to an application program, the connection must be broken before the terminal can be made available to TOLTEP. VTAM allows an application program to be notified if a connected terminal is to be used by TOLTEP; although the application must issue a CLSDST macro instruction to release the terminal. If a connected terminal specified in a test request is not released by the application program, TOLTEP waits until the terminal is available before proceeding with the test. Thus, the installation should assure that application programs are designed to release terminals when needed by TOLTEP.

When TOLTEP is invoked, it requests information from the operator at the invoking terminal (or the terminal specified in a network-operator logon for TOLTEP). Requested information includes the name of the control terminal (unless a logon was used to invoke TOLTEP, in which case the terminal specified in the logon is used as the control terminal), the name of the alternate printer (optional), the name or names of the nodes to be tested, and the tests to be executed. If TOLTEP was invoked by other than the MODIFY command, the network operator is notified of the terminals and other nodes specified in the request to TOLTEP and must grant permission for the testing to proceed. Any additional nodes identified by the control terminal for testing must also be approved by the network operator. (The installation-coded authorization exit routine must permit TOLTEP to connect and disconnect from terminals involved in tests.)

Once authorization has been given by the network operator, TOLTEP connects to the appropriate terminals and begins testing the specified nodes according to instructions from the control terminal and the appropriate OLTs.

For information about how to run TOLTEP, see *DOS/VS and OS/VS TOLTEP for VTAM.*

Reliability and Availability Support

The purpose of VTAM's reliability and availability support is to maintain the operation of the telecommunication network. This support attempts to prevent problems, and if that is not possible, to minimize the impact of the problems. (To enhance reliability and availability, most of the VTAM modules are reusable or reenterable.) VTAM's reliability and availability support includes:

Error Detection and Feedback: Before attempting to act upon any request, VTAM analyzes it for any erroneous information. If an error is detected, VTAM returns the request along with an indication of the error.

NCP Initial Test: VTAM allows a communications controller to be tested before an NCP is loaded. This test can be used to identify problems in a controller before it is made an active part of the system. This testing is optional for locally attached communications controllers; it is required for remotely attached communications controllers.

Storage Management: VTAM controls the allocation of much of the storage required for its operation. Using this control, VTAM permits the queuing of requests and attempts to avoid storage interlocking conditions. (A storage interlocking condition is one in which so much of VTAM storage has been devoted to the initiation of request processing that not enough is available to complete that processing.)

Hardware Error Recovery: Using the facilities of the operating system and of the NCP, VTAM attempts to recover from some errors. If recovery is not possible, a permanent

error is recorded and VTAM attempts to reallocate system resources to reduce the impact of the failure. If recovery is possible, processing continues, but a count is maintained of the initial failure.

Software Error Recovery: VTAM tries to recover from some errors. If recovery is not possible, VTAM first attempts to isolate the problem and continue processing. If VTAM cannot continue processing, it attempts an orderly closedown of the telecommunication system so as not to affect the non-telecommunication jobs in the host CPU.

NCP Slowdown: VTAM recognizes NCP slowdown and assists the NCP to return to normal operations.

Configuration Restart: If an error in the telecommunication network causes an NCP to terminate, VTAM attempts to restart the affected NCP.

These operations are discussed in more detail below.

Error Detection and Feedback

All requests from the network operator and from application programs are initially tested for errors. If an error is detected, the request and an indication of the error are returned to the requester. VTAM does not process a request it determines to be in error. If, during the processing of the request, an invalid condition is encountered, VTAM stops processing the request and returns an error indication to the requester. No further processing is done for the request.

If the requester is the network operator, VTAM first checks the syntax of the command. If there is an error in syntax, the command is rejected with a message to the network operator; the message describes the error and specifies where it occurred. If the syntax is valid, the requested function is validated. (For example, a request to activate a minor node whose major node is inactive is an invalid function request.) If the function requested is invalid, the command is returned to the operator along with an explanation of the error.

If both the syntax and the function are validated but VTAM encounters an error while processing an operator command, the network operator is notified of the unsuccessful operation and of the reason for the failure.

In addition to notifying the network operator of error conditions directly attributable to operator commands, VTAM transmits messages to the console describing any unusual or error conditions that affect the operation of the telecommunication system; for example, permanent hardware errors and the abnormal termination by VTAM of an application program.

If an error condition is found in either an application program's request or in the processing of such a request, VTAM attempts to notify the application program of the error. Notification is made by scheduling a user-specified exit routine or by a return code. VTAM has numerous return codes to help isolate the reason for the error. VTAM avoids abnormally terminating application programs possible. Using the exit routines and the return codes, the application program can attempt to correct the error, bypass the error, or terminate processing.

NCP Initial Test

The NCP initial test is a facility of the communications controller. It is exercised automatically for remotely attached communications controllers; it is an option for locally attached communications controllers. If testing is specified in the generation of an NCP for a locally attached communications controller, VTAM automatically invokes the

test program prior to loading that NCP. If testing is not specified, initial testing of the NCP is not performed for a locally attached communications controller.

Storage Management

VTAM has a number of storage pools as noted in "VTAM Buffering," in Chapter 3. The installation can specify the size of each pool and a threshold value for each.

These storage pools are used to obtain buffers to transfer data between the host CPU and the network and to obtain storage for VTAM control blocks needed to service each telecommunication request. Because these pools are in VTAM storage and are managed by VTAM, each application program need not be significantly concerned with providing storage in its own partition or address space for VTAM. (The application program need provide storage only for the work areas and VTAM control blocks, such as the ACB, EXLST, NIB, and RPL, that it uses directly.)

VTAM consolidates much of the storage requirements for buffers and control blocks for all of its application programs and enables the storage pools to be shared among these applications.

VTAM manages its storage pools by determining when storage should be requested, the amount of storage to request, and whether sufficient space is available to meet the request. VTAM also enables requests for buffers to be queued if sufficient space is not available to meet demands. By queuing storage requests, VTAM reduces the need to abnormally terminate an application program when insufficient storage is available at a particular time for its operation.

The pool used to satisfy a storage request within VTAM depends upon the type of request. For example, the pool used to obtain storage for data buffers is not the same as the one used for control blocks, and the pool used to obtain storage for a control block built during the initial stages of processing a request is not the pool used to obtain storage for a control block built during the completion of that request processing. By discriminating among storage requests, VTAM attempts to avoid storage interlocking conditions. If one pool should become temporarily exhausted, the VTAM procedures that need storage from that pool may be forced to wait until it is replenished, but procedures that need storage from other pools can continue processing.

In OS/VS, VTAM also allows the installation to specify a maximum threshold value for each pool. When a pool is operating above its threshold, only priority requests (determined by VTAM) for storage from that pool are satisfied. Use of the threshold option can assist VTAM to maintain the availability of the telecommunication system by allowing critical telecommunication functions to continue even if the noncritical functions must wait temporarily. When storage is not available in a pool, VTAM queues storage requests for that pool; then, when storage is available, those requests are serviced.

Establishing buffer limits for terminal and application-program connections is another option that affects VTAM's storage management. Using this option, the installation can prohibit low-priority jobs from monopolizing VTAM's storage pools.

In addition to managing storage by controlling its allocation, VTAM attempts to protect its storage from unauthorized access. VTAM components, control blocks (except for application programs' ACBs, NIBs, and RPLs), and buffers reside in storage protected from nonprivileged users by an operating-system key.

Error Recovery

VTAM attempts to recover from both hardware and software errors. Both kinds of recovery are discussed below.

Hardware ERPs: When an I/O error interruption occurs, the appropriate error recovery procedures (ERPs) are invoked (within the operating system or VTAM for errors on locally attached devices and within the NCP for errors on remotely attached devices). The ERPs attempt to determine the type of error and to recover from it.

When a permanent error is encounterd for a locally attached device, the ERP notifies the network operator of the problem and creates an error record as described in "Serviceability Aids," earlier in this chapter. When a temporary error is encountered, no message is sent to the console although a count is maintained of the temporary errors for that device.

Each NCP provides ERPs for the devices attached to it. When an NCP has completed error-recovery processing, it transmits an error record to VTAM for recording on the operating system's error-record data set. For a permanent error, a message describing the problem is sent to the network-operator's console.

Processing Software Errors: When VTAM encounters a software error, it attempts to determine whether the error resulted from action on the part of the user, the operating system, or VTAM itself. If it is a user error, it is handled as explained in "Error Detection and Feedback," earlier in this chapter.

If VTAM cannot determine whether the problem is a user error, the access method attempts to recover from the error. If recovery is not possible, VTAM attempts to isolate the cause of the problem and to bypass it. If this partial recovery is not possible, VTAM abnormally terminates, attempting to close down the telecommunication system without impacting the rest of the operating system. Note that if a problem can only be alleviated by abnormally terminating the application program associated with the problem, VTAM terminates the application. Terminating the application program is an attempt to reduce the impact of an error on the total telecommunication system; terminating one application program still permits other applications to continue to use the telecommunication system.

NCP Slowdown

If telecommunication activities overload an NCP and exhaust its buffers, the NCP automatically enters slowdown mode. In slowdown mode, the NCP reduces its acceptance of input data (from both the network and the host CPU) and attempts to increase the rate of output. VTAM recognizes an NCP slowdown, and, to facilitate NCP recovery, VTAM stops scheduling input and output requests for that NCP, although VTAM does continue to read from the NCP. During this slowdown processing, VTAM continues to accept requests from application programs, but it queues those requests that are directed to nodes in the network controlled by the NCP in slowdown mode. When the NCP has recovered and is no longer in slowdown mode, the queued requests are processed and sent to the NCP.

Configuration Restart

VTAM provides the capability to restart an NCP in a communications controller after a failure occurs in that controller requiring the reloading of the control program. An NCP can be restarted only if the original failure was in the communications controller. If the failure is in the host CPU, configuration restart cannot be used.

When an error occurs in a communications controller, VTAM's ERPs determine whether the channel (for a local attachment) or the line (for a remote attachment) failed. If neither failed, VTAM initiates the NCP dump program (if automatic dumping was specified in the generation of the NCP or if requested by the network operator). If the network operator specifies the NCP is to be reloaded, VTAM attempts to reload and restart it. NCP initial testing is performed if the communications controller is remotely attached or if the communications controller is locally attached and the NCP was

generated with the test option requested. After the NCP is restarted, VTAM attempts to reinstate the network status at the time of the failure. That is, nodes active at the time of failure are reactivated (if possible), any remotely attached communications controllers active at the time of the failure are restarted if possible, any line-scheduling parameters modified by the network operator are reestablished, and (if the NCP included PEP) lines are reinstated to the mode they were in when the failure occurred.

If an NCP is successfully restarted, VTAM attempts to restore the connections between application programs and terminals:

- For application programs that were connected to terminals in basic mode, the connections are maintained by VTAM for host terminals that were successfully reactivated. Outstanding I/O requests for these terminals are purged, and the application programs must resubmit their requests.
- For application programs that were connected to terminals in record mode, the application programs must issue CLSDST and OPNDST macro instructions (in that order) for the terminals.
- For application programs connected to terminals that could not be reactivated, the connections are lost; the applications should issue a CLSDST macro instruction for those terminals but can continue processing with unaffected connections.

CHAPTER 7. VTAM PLANNING CONSIDERATIONS AND REQUIREMENTS

The first part of the chapter states the machine and program requirements (including NCP requirements) for using VTAM. The second part of the chapter discusses ways an installation can plan for and take advantage of features and facilities in VTAM.

Machine Requirements

This section discusses the machine requirements for a telecommunications system with VTAM. It specifies which units can be used with VTAM including the central processing units (CPUs) and their required features, communications controller requirements, and terminals and terminal features.

CPU Support

VTAM is a System/370 access method and runs in a virtual storage environment on one of the following CPUs with the relocation feature:

- Under DOS/VS: System/370 Models 115, 125, 135, 145, 155II, 158, and 158 Submodel 2 (available for World Trade countries only).
- Under OS/VS1: System/370 Models 135, 145, 155II, 158, 158 Submodel 2 (available for World Trade countries only), 165II, and 168.
- Under OS/VS2: System/370 Models 145, 155II, 158, 158 Submodel 2 (available for World Trade countries only), 158MP, 165II, 168, and 168MP.

The host CPU must be equipped with the Compare and Swap and the Compare Double and Swap instructions. These instructions are available as follows:

- Optional on the System/370 Model 135 via the Conditional Swapping Feature 1051.
- Optional on the System/370 Model 145 via the Advanced Control Program Support Feature 1001 or via the Conditional Swapping Feature 1051.
- Standard on the System/370 Models 115, 125, 155II, 158, 158 Submodel 2 (available for World Trade countries only), 158MP, 165II, 168, and 168MP. 168, and 168MP.

Local 3270s

VTAM provides communication with IBM 3270 Information Display Systems that are locally attached to the host CPU. The maximum number of 3270s that can be used and any required features are determined by the 3270 and the system, not by VTAM. Appendix A lists the devices and features for locally attached 3270s that can be used with VTAM.

Local 3790s

VTAM provides communication with IBM 3790 Communication Systems that are locally attached to the host CPU. Appendix A lists devices that may be used in a 3790 system.

Requirements for Communications Controllers

For remotely attached devices, VTAM requires a locally attached IBM 3704 or 3705 Communications Controller.

VTAM uses only the network control program/VS (NCP) for communications controllers. VTAM uses only the network control mode of the NCP, and it uses this mode with or without the partioned emulation programming (PEP) extension. VTAM Level 2 requires NCP/VS Version 4 Modification Level 0 or subsequent levels.

VTAM can use both the locally and remotely attached communications controllers. A remote communications controller must run in network control mode only and must be attached to a local communications controller on an SDLC line controlled by the network control mode of the NCP in that local controller.

An NCP used by VTAM requires at least 48K bytes of storage in the communications controller; therefore VTAM can use the following communications controller models:

- IBM 3704 Communications Controller–Models A3 and A4.
- IBM 3705 Communications Controller–Models A2, B2, B3, B4, C2, C3, C4, C5, C6, D2, D3, D4, D5, D6, D7, and D8.

VTAM does not support the following models because they have less than 48K bytes of storage:

- IBM 3704 Communications Controller–Models A1 and A2.
- IBM 3705 Communications Controller–Models A1, B1, C1, and D1

Except for storage capacity, the features required on a communications controller do not depend upon VTAM. Required features depend upon the intended application of the communications controller (including local or remote attachment) and the type of control program to be used (NCP with or without PEP).

Channel Adapter Support: Support of channel adapters depends on the communications controller and the NCP.

- The locally attached 3704 is equipped with the Type 1 Channel Adapter.
- The locally attached 3705 can be equipped with either the Type 1, Type 2, or Type 3 Channel Adapters.

3705 Two-Channel Support: Using NCP facilities and a Type 3 Channel Adapter, VTAM supports the attachment of two channels to a 3705 in an OS/VS2 tightly-coupled multiprocessing environment. This attachment is not available for a loosely-coupled multiprocessing environment for two independent CPUs.

VTAM does not support the manual Two-Channel Switch (8002) for the 3705. The installation is responsible for support of this feature in a VTAM system.

Backup (Alternate) SDLC Link Between Local and Remote Communications Controllers: VTAM supports use of an alternate SDLC link to back up the main SDLC link between a local and a remote communications controller. This link can be switched or nonswitched. VTAM notifies the network operator of the main link's failure; the network operator can then activate the backup link. *NCP Generation* describes how to specify a backup link. *VTAM Network Operating Procedures* describes the procedures that the network operator must perform.

Other Features not Supported: VTAM does not support speed selection for lines equipped with Dual Speed Modems. It also does not support Switched Network Backup.

Remotely Attached Terminals

VTAM can control remotely attached terminals only if they are attached through a communications controller in network control mode. SNA start-stop, and BSC terminals can be attached either to a local communications controller or to a remote communications controller.

Appendix A lists all devices and features that can be used as remote terminals by VTAM.

Because VTAM uses the NCP to control and communicate with remotely attached terminals, most requirements, restrictions, or limitations pertaining to device features are those of the NCP, not of VTAM. (See the *NCP Generation* publication for details on device requirements, restrictions, and limitations for NCP.)

Storage Requirements

Requirements for CPU storage, disk storage, and communications controller storage for VTAM users differ at each installation.

Requirements for CPU storage and for disk storage for VTAM data sets can be calculated by using one of these publications, depending on which operating system is used:

- *DOS/VS System Generation,* GC33-5377.
- *OS/VS1 Storage Estimates,* GC24-5094.
- *OS/VS2 System Programming Library: Storage Estimates,* GC28-0604.

Requirements for disk storage for NCP data sets can be calculated by using the *NCP Generation* publication, GC30-3008.

Requirements for TOLTEP data set storage are included as part of VTAM data set storage in the publications mentioned above, since TOLTEP is included in VTAM. Disk storage information for the online tests themselves is provided by the IBM program service representative.

Requirements for communications controller storage can be calculated by using *IBM 3704 and 3705 Communications Controllers Network Control Program Storage and Performance Estimates (for OS/TCAM, OS/VS TCAM, and OS/VS and DOS/VS VTAM Users),* GC30-3006.

Operating System Requirements

This section discusses VTAM's requirements for each operating system (DOS/VS, OS/VS1, and OS/VS2) in which VTAM can be included.

Upward Compatibility

To facilitate teleprocessing growth, VTAM's application program macro language is upward compatible from DOS/VS to OS/VS1 to OS/VS2. In addition, procedures for defining the telecommunication network and network operator commands are similar for all three systems.

Although VTAM does provide upward compatibility for telecommunications, it uses the facilities of the operating system under which it is executing. VTAM's requirements for each of these systems are detailed in the remainder of this chapter.

Requirements for DOS/VS

Generating VTAM in a DOS/VS system requires specification of VTAM as a teleprocessing access method in the SUPVR macro instruction for DOS/VS system generation. VTAM also requires the inclusion of some DOS/VS facilities that would be optional if VTAM were not in the system. Of these facilities required by VTAM, multitasking support must be specified by the installation during system generation. The other options are generated automatically if VTAM is specified; they include:

- Support for the use of the STXIT macro instructions (all options) by problem programs
- Storage-management support for the GETVIS and FREEVIS macro instructions
- Use of the PFIX and PFREE macro instructions for fixing and freeing pages
- Inclusion of the relocating loader
- Support for the time-of-day clock

All locally attached devices in the VTAM network must be identified during the generation of the operating system or added at IPL (initial program loading) time. These devices are locally attached 3270s, locally attached 3790s, and the locally attached communications controllers used by VTAM. Telecommunication lines and remotely attached devices need not be defined at system generation if they are to be used only through VTAM. If they are to be used directly by other access methods, they must be defined according to the requirements of those access methods; remotely attached devices so defined are still available to VTAM through the NCP. Remotely attached devices are defined to VTAM as explained in "VTAM Definition," in Chapter 3.

A DOS/VS system with VTAM must have at least two partitions: one for VTAM and one for VTAM application programs. Additional partitions may be needed for other VTAM application programs or for programs unrelated to VTAM. The partition containing VTAM must have a priority higher than any other partition that contains a VTAM application program. In addition to DOS/VS tasks required for application programs, three DOS/VS tasks are required for VTAM. A fourth task is required if the network solicitor is used.

VTAM is started under DOS/VS by entering an EXEC command or job control statement for the partition in which VTAM is to run. In addition to the EXEC statement, an ASSIGN command or statement is required if a local communications controller is to be included (that is, if remote terminals are to be included) in the VTAM network. The assignment must make logical unit SYS000 unassigned for the duration of the VTAM job step. Any other commands or job control statements required to start VTAM depend upon factors such as system generation and telecommunication options to be used. See "VTAM Data Sets," later in this chapter, for library and file requirements for VTAM under DOS/VS.

Requirements for OS/VS

To generate VTAM in an OS/VS system requires that VTAM be specified as a teleprocessing access method in the DATAMGT macro instruction for OS/VS system generation.

All locally attached devices in the VTAM network must be identified at system generation. These devices are the locally attached 3270s, locally attached 3790s, and the locally attached communications controllers used by VTAM. Telecommunication lines and remotely attached devices need not be defined at system generation if they are to be used only through VTAM. If they are to be used directly by other access methods, they need to be defined according to the requirements of those access methods; remotely attached devices so defined are still available to VTAM through the NCP. Data set requirements for OS/VS are discussed in "VTAM Data Sets," later in this chapter. Remotely attached devices are defined to VTAM as explained in "VTAM Definition," in Chapter 3.

For OS/VS1, VTAM runs as a system task; at least one problem program (VTAM application program) partition is required. The size of the problem program partition can be determined by using guidelines described in *OS/VS1 Storage Estimates.*

A cataloged procedure must be created for VTAM. This procedure must contain DD statements for SYS1.VTAMLST and SYS1.VTAMLIB. In addition, the procedure must contain statements for any NCP data sets that are to be used by VTAM. It must also contain a DD statement for SYS1.VTAMOBJ and DD statements for the TOLTEP data sets, if TOLTEP is to be invoked. Locally attached 3270s and 3790s, communications controllers, and remotely attached terminals and devices that are part of the VTAM telecommunication network do *not* require DD statements.

Multiple console support (MCS) can be used with VTAM. Any MCS console that is to be used by the VTAM network operator must be assigned a command authority of two and a routing code of eight.

It is not recommended that VTAM be run with the Dynamic Support System (DSS). The effects of DSS on VTAM includes DSS time delays that may result in NCPs in the VTAM network automatically closing down the network. Refer to the publication *OS/VS Dynamic Support System*, GC28-0640, for details on DSS.

If a VTAM application program is to use the OS/VS2 authorized path facility, the program must be authorized. See *OS/VS2 System Programming Library: Supervisor*, GC28-0628, for how to specify an authorized program. (The use of authorized path in a VTAM application program is described in "Using Authorized Path in OS/VS2" in Chapter 5.)

Network Control Program Requirements

This section describes the requirements placed upon an NCP by VTAM. It describes the relationship between NCP generation and VTAM definition. It also discusses VTAM's support of switched networks and VTAM's impact on the NCP with PEP.

Introduction to the Network Control Program

VTAM uses the network control mode of the network control program/VS (NCP) with the IBM 3704 and 3705 Communications Controllers to control and communicate with remotely attached devices. The NCP is generated with NCP generation macro instructions. These same instructions are then used as definition statements to define the remote fixed part of the network to VTAM. See "VTAM Definition," in Chapter 3, for more information on using the same deck of statements both to generate an NCP and to define the NCP and its network to VTAM. The variable part of the remote network, the switched major nodes are not described to the NCP; they need only be described to VTAM.

The remainder of this chapter contains details on VTAM's use of the NCP. Refer to the publication *Introduction to the IBM 3704 and 3705 Communications Controller*, GA27-3051, for a description of the NCP and its facilities. Refer to the *NCP Generation* publication for details on NCP generation macro instructions and NCP generation procedures.

NCP Functions Required by VTAM

Several types of NCPs may be needed in a VTAM telecommunication system:

- An NCP for a local communications controller to which no remote communications controller is attached
- An NCP for a local communications controller to which a remote communications controller is attached
- An NCP for a remote communications controller
- An NCP with the partitioned emulation programming (PEP) extension

Whatever the NCP or NCPs used, some functions that are optional for the NCP are required by VTAM. These functions are part of the NCP's dynamic control facilities that are created during NCP generation, and they are specified in the NCP's SYSCNTRL statement. The dynamic control facilities that must be specified for start-stop or BSC lines or certain types of operator control include facilities to:

- Change mode settings for a device
- Disconnect a dial connection

- Request a device to yield control of a line
- Reset a device if data transfer has not taken place
- Reset a device if data transfer has taken place
- Reset a device immediately

Defining the NCP to VTAM

To define an NCP and its remote network to VTAM, the following information must be specified in the definition (also the generation) statements:

- The capabilities of the NCP
- The interface between the NCP and VTAM
- The network configuration

NCP generation macro instructions and the PCCU definition statement supply this information to VTAM. Two additional VTAM-only definition statements, VIDLIST and VTERM, apply only to BSC and start-stop terminals. They are described in Chapter 8.

The PCCU statement, used only by VTAM, identifies the communications controller into which the NCP would normally be loaded when activated. One PCCU statement is required for each NCP defined to VTAM.

This statement also defines the functions VTAM is to provide for the NCP. These functions can include:

- Dumping the communications controller automatically following an unrecoverable error in that controller. If a dump is to be taken, a pointer to the name of the data set to contain that dump is specified in the PCCU statement.
- Reloading and restarting the NCP automatically following an unrecoverable error in the communications controller.
- Invoking the NCP initial test routine automatically prior to loading the NCP. (Initial testing is optional only for NCPs for locally attached communications controllers; initial testing is performed automatically whenever a remotely attached communications controller is loaded.)

If the NCP is for a remotely attached communications controller, this statement also provides a link to the NCP for the locally attached communications controller. A PCCU statement for an NCP for a remote controller, specifies the name of a PU or INNODE statement in another NCP deck. This deck defines the NCP for the local communications controller, and the INNODE statement defines the remote controller.

In addition to using the PCCU definition statement, VTAM requires information contained in the NCP generation macro instructions (also called VTAM definition statements). Most information required by VTAM is also required by the NCP, although some additional data is required by VTAM alone.

Defining the Nonswitched Network to VTAM: Using the GROUP, LINE, CLUSTER, PU, and LU statements that define the nonswitched network to the NCP, the installation defines the nonswitched network to VTAM. The GROUP statement defines a group of SDLC lines; the LINE, CLUSTER, PU, and LU statements define the configuration associated with each group. The CLUSTER statement applies only to a 3601 Finance Controller. Either a CLUSTER or a PU statement can be used to define a 3601. The physical units for other SNA terminals are always described with a PU statement.

Defining the Fixed Part of the Switched Network to VTAM: The GROUP, LINE, PU, and LUPOOL statements that are used to describe the fixed part of the switched network (that is, the switched lines) to the NCP, are also used to describe the fixed part of the

switched network to VTAM. The GROUP statement indicates the start of a switched line group and specifies the line control (SDLC) and that the lines are switched. The LINE statement specifies whether dialing is manual or automatic and other features. The PU statement defines the limitations of each line with respect to the type of physical unit that can be supported. The LUPOOL statement defines the total number of logical units that the line group can support. Note that the variable part of the switched network is described to VTAM by one or more sets of switched subarea statements; those subareas that are made active define to VTAM the variable part of the switched network.

Defining Logical Units: Using LU statements, the installation can specify to VTAM the following information for logical units:

- Automatic logon requirements.
- The initial status of the logical unit when the NCP is activated by VTAM.
- The buffer limit for the logical unit. See "Defining VTAM Buffering," in Chapter 3, for a description of how buffer limits are established.
- Pacing requirements for each logical unit.
- The name of a logon mode table that describes the logical unit's sets of session parameters.

Logical units are initially activated by VTAM only if they are defined as initially active and their physical units are initially active. Physical units, in turn, are initially activated only if they are defined as initially active and, for subsystem physical units, if they have been manually loaded prior to activation by VTAM.

Pacing

VTAM enables the installation to control the rate of data flow through the network path joining a VTAM application program and a logical unit. This control is called pacing. Pacing is intended to protect a node that is receiving data from being overloaded by too much input. Using the specifications supplied in an LU statement by the installation, both VTAM and the NCP can limit the number of messages transmitted to a particular logical unit before a response is returned that the messages have been received. For each logical unit, the installation can specify pacing requirements for the transmission of messages from VTAM to the NCP and for the transmission of messages from the NCP to the logical unit.

Support for SNA Switched Networks

This section describes VTAM support for dial-in and dial-out operations in a switched network using SDLC lines. (For information on switched network support for start-stop and BSC terminals, see Chapter 8). This section discusses:

- Defining a switched major node to VTAM
- Generating a group of switched lines in the NCP for dial-in and dial-out operations
- ID verification—what it is and when you define it
- Performing dial-in operations
- Disconnecting physical units and logical units with or without automatic call termination

Defining a Switched Major Node

A switched major node is a logical grouping of physical units and logical units that are serviced by SDLC switched line groups. Each switched major node must be defined to VTAM by these statements:

- The VBUILD statement defines the start and limitations of each major node.
- The PU statement describes a particular physical unit in the switched major node and its disconnection procedure (where it has automatic call termination).

- The PATH statement (for dial-out operation only) describes such information as a telephone number to be used, a group name (for a group of switched lines generated in the NCP), and a redial count for autodial devices.
- The LU statement describes characteristics of each logical unit supported by the physical unit.

Each physical unit and its associated logical units defined to VTAM in a switched major node can perform dial-in operations, but only physical units with a PATH statement can perform dial-out operations. (A PATH statement is also required for physical units that may be dialed out from the host and dialed in to the host.)

Generating Switched Line Groups in the NCP

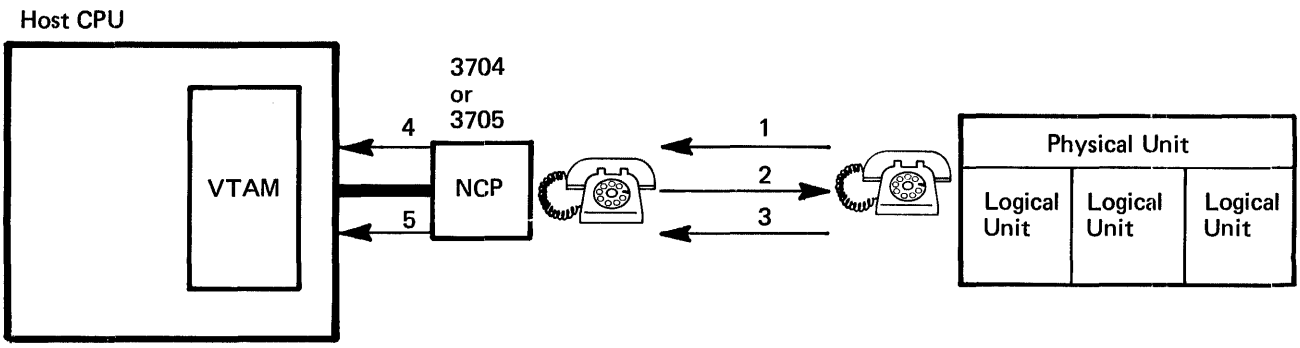
Switched line groups are defined as the last part of the NCP generation. Each switched line group is used to service physical units and their associated logical units in switched major node dial-in and dial-out operations. The following macro instructions must be used to define a switched line group:

- The GROUP macro instruction defines the start of a group of switched lines, the line control (for example, SDLC), and whether the lines have dial support.
- The LINE macro instruction describes whether a line has automatic or manual dial support and the line's function (dial-in, dial-out, or dial-in/out capability).
- The PU macro instruction describes the limitations of each line with respect to what type of physical unit can be supported on the line.
- The LUPOOL macro instruction states the total number of logical units that can be supported by the group of switched lines being defined.

Each group of switched lines is known to VTAM. During dial-out operations, VTAM determines for each line in the group the maximum number of logical units supported on the line, and whether the line is an in line, an out line, or an in/out line.

ID Verification for Switched Major Nodes

During dial operations, the IDs of the physical unit and the host are checked. If each ID is valid, normal connection procedures will occur to establish connection between a logical and an application program. (See Figure 7-1.)



1. A telephone is used to dial a dial port of a communications controller sending a signal to the NCP (if the dial port does not answer, other dial ports are called.)
2. The NCP sends an XID (exchange IDs) to the physical unit.
3. The physical unit responds with its identifier to be checked.
4. The NCP sends the physical unit's identifier to VTAM to be checked.
5. VTAM checks the identifier and returns information pertaining to the physical unit. The connection procedure from this point on is the same as that for a physical unit on a nonswitched line.

Figure 7-1. Dial-In Operation

In a dial-in operation, when a dial port is found to handle the dial-in request, the NCP sends an XID (exchange IDs) command to the physical unit that requests connection. The physical unit responds with its ID to be checked. The physical unit's ID is then sent to VTAM which validates the ID. Finding that the physical unit's ID is valid, VTAM sends an ACTPU (activate physical unit) containing the host ID to the physical unit.

The dial-out operation verifies the ID in a similar manner. After a dial line is established from a dial port to the physical unit, the same series of events for checking IDs occurs.

Dial-In Operation

A dial-in operation can be performed by any physical unit defined in a switched major node (see Figure 7-1). To initiate a dial-in operation:

- The dial port of the 3704 or 3705 that is being called must be active.
- The switched major node containing the physical unit and logical unit must be active.
- The VTAM application program to which connection is requested must be active with a LOGON exit routine or a pending OPNDST with OPTCD=ACCEPT to handle the dial-in request from the logical unit.

The dial-in operation can be accomplished by manually dialing a dial port of the communications controller or by using a subsystem program of a controller (for example, a 3791 controller can be programmed to dial ports) to dial a port of a local or remote communications controller. As soon as a port is found to handle the dial-in, host and physical unit IDs are exchanged. The rest of the connection procedure is the same as for physical units and logical units on nonswitched lines.

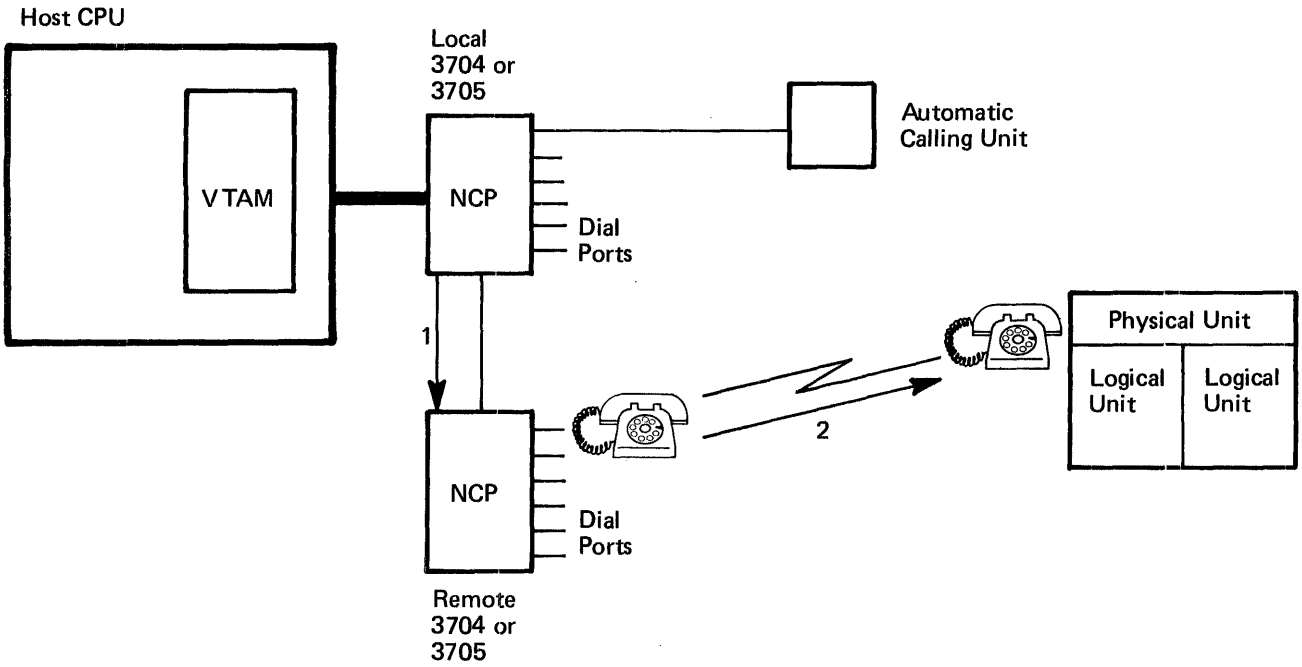
As with logical units on nonswitched lines, following the dial-in operation, the logical unit on a switched line can become connected to a VTAM application as the result of: (1) a request from the logical unit, or (2) a previous network operator-initiated request, or (3) a VTAM-initiated (automatic logon) request. In the case of a network operator- or VTAM-initiated logon request for a logical unit that must dial in, VTAM holds the request until the logical unit dials in. An application program-initiated request for connection to a dial-in logical unit (using SIMLOGON or an OPNDST with OPTCD=ACCEPT in a logon exit routine of an OPNDST with OPTCD=ACQUIRE) is unsuccessful unless the logical unit dialed in before the request issued.

Dial-Out Operation

A dial-out operation is performed by VTAM on behalf of an application program to connect it with a specific logical unit (see Figure 7-2). The conditions for performing the dial-out operation are similar to the dial-in operation with the exception that the application program must issue an OPNDST with OPTCD=ACQUIRE. Dial-out operation can be performed only on those physical units and logical units that have PATH statements associated with them in the VTAM definition.

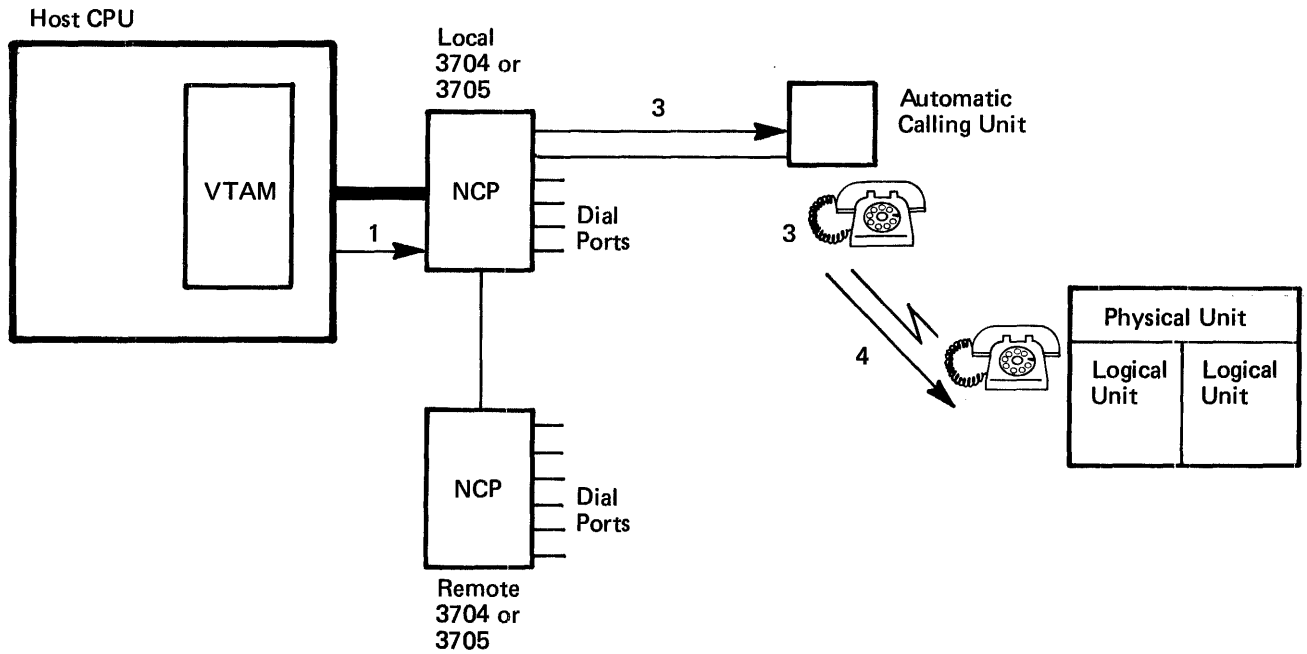
To accomplish a dial-out, VTAM uses the path table (built from the PATH statement defined during the VTAM definition) and the switched line groups. When a request is made to perform a dial-out to a particular physical unit, VTAM:

- Picks a usable line in the switched line group and specifies the line, a telephone number, and a redial count (for auto dial only) to the NCP to do the dial-out operation.
- Finds another usable line in the path (until connection is made or all the lines are used in the path) if the previous line cannot be used.
- Uses another path if the previous path fails (see Figure 7-3).
- Fails the OPNDST with OPTCD=ACQUIRE if all paths are nonusable. The VTAM application program can determine the reason for the failure and try again later.



1. VTAM sends a message (containing a telephone number, a line to use, and, if an automatic calling unit is attached, a redial count) to a communications controller to dial-out to a particular physical unit.
2. The communications controller automatically or the operator manually dials the telephone number of the physical unit and starts to verify the identification of the physical unit.

Figure 7-2. A Path For a Dial-Out Operation



1. VTAM sends a message to a communications controller to perform a dial-out to a specific physical unit.
2. A signal is sent to an attached automatic calling unit (containing a telephone number).
3. The automatic calling unit dials the telephone number of the physical unit.
4. A signal is sent to the physical unit that will start the connection procedure.

Figure 7-3. An Alternate Path For a Dial-Out Operation With An Automatic Calling Unit

Paths can be controlled by operator commands. By coding the path identifier (PID) or a group identifier (GID), a path for a particular physical unit or a group of paths for a major node can be activated or deactivated. Group identifiers (GIDs) are defined by the installation to classify groups of paths in a switched major node (for example, to separate classes of telephone numbers such as local numbers, WATS lines, long distance, and direct dial). The path identifier (PID) is used to identify each unique path to a particular physical unit.

Another consideration in a dial-out operation is whether the dialing operation is done manually or by an automatic calling unit. When manually dialing, a message is sent to the console of the network operator telling him to dial a telephone number, using a specific line. When an automatic calling unit is attached to a communications controller, the unit will use the line given and dial the telephone number a specified number of times.

Disconnecting Physical Units and Logical Units in a Switched Major Node

All physical units and logical units in a switched major node have automatic call termination unless the installation specifies otherwise. When all sessions between logical units and application programs are terminated, the physical unit is disconnected.

Disconnection procedures can be changed for each physical unit in the switched major node using the DISCNT operand on the PU statement during VTAM definition. If DISCNT=YES is specified, automatic call termination is used as described above. If DISCNT=NO is specified, the physical unit remains connected to VTAM unless the disconnection request for the last logical unit requests that the physical unit be disconnected. See "Holding the Physical Unit Connection," in Chapter 3, for a description of specifying disconnection of the physical unit on a disconnection request.

Partitioned Emulation Programming (PEP) Considerations

VTAM can use the NCP with the partitioned emulation programming (PEP) extension. When PEP is in use in the NCP, VTAM does not provide services for the network serviced by emulation mode.

VTAM does provide three facilities that affect the emulation mode of an NCP with PEP: dumping the communications controller, reloading the controller, and changing line assignments.

If the network operator invokes the NCP dump utility program (using VTAM's MODIFY command), the dump disrupts the emulation function as well as the network control function when an NCP with PEP resides in the communications controller. After a dump is taken the NCP has to be reloaded before it can be used. (An NCP with PEP is loaded automatically by VTAM when the access method activates an NCP major node defining a network control function that is part of an NCP with the PEP extension.)

VTAM also provides support for the restart capability of the NCP. If an error is encountered in a communications controller and this error causes VTAM to reload the controller, the reloading disrupts emulation because the entire NCP is reloaded, including the emulation function.

VTAM also affects the emulation function in an NCP with PEP by changing line assignments. Using PEP, lines can be assigned to network control mode, emulation mode, or to both. If possible, VTAM assigns a line to both modes. VTAM's support for such lines is as follows:

- If the line is not active for VTAM, it is automatically assigned to emulation mode.
- Line assignments are changed in response to activation and deactivation requests from the network operator for VTAM. Lines are assigned to network control mode when they are activated by VTAM, and they are reassigned to emulation mode when they are deactivated by VTAM.

VTAM Data Sets

VTAM uses a number of data sets (files in DOS/VS) to support a telecommunication system. These data sets contain such items as VTAM modules, VTAM definition statements, NCP modules, and RAS records. Data set requirements vary depending upon the operating system in which VTAM is being used.

Data Sets for VTAM under OS/VS

In an OS/VS system, VTAM uses NCP data sets, operating system data sets, and data sets devoted entirely to VTAM itself. Figure 7-4 is a table of all the data sets used by VTAM in an OS/VS system. This table is divided into columns; each defined as follows:

- Name: Specifies the name assigned to the data set.
- VTAM Contents: Indicates what information in the data set is used by VTAM.
- When Created: Specifies the latest time that the data set can be created.
- JCL: Specifies whether a DD statement must be provided for the data set in the VTAM start procedure.
- Comments: Provides additional information relevant to the data set. If the data set is specified as "required," it is required by VTAM. If it is specified as "optional," it is not required by VTAM.

  Note: *If an optional data set is not included in a VTAM system, the contents of that data set, and therefore the associated function, are not available to VTAM.*

  Refer to the publications *OS/VS1 Planning and Use Guide* or *OS/VS2 Planning Guide for Release 2* for more information on the operating system data sets listed in Figure 7-1. Refer to the publication *Introduction to the IBM 3704 and 3705 Communications Controllers* for additional information on NCP data sets listed in Figure 7-1.

Data Requirements for VTAM under DOS/VS

A file in DOS/VS is comparable to a data set in OS/VS. VTAM has file requirements in a DOS/VS system similar to the data set requirements in OS/VS. Figure 7-5 is a table that shows the file requirements for VTAM under DOS/VS. This table is divided into columns that are defined as follows:

- File Name: Specifies the name of the file. Except as indicated in Figure 7-2, this name must be specified as shown.
- File ID: Specifies the identification of the file on the volume.
- Contents: Indicates what VTAM uses in this file.
- Comments: Provides additional information relevant to the file. If the file is specified as "required" it is required by VTAM.

Refer to the publication *Introduction to DOS/VS*, for additional information on the DOS/VS logical units and files. Refer to the publication *Introduction to the IBM 3704 and 3705 Communications Controllers* for additional information for NCP data requirements.

VTAM also has data requirements pertaining to DOS/VS libraries. Whether these libraries are system or private libraries depends upon system generation options selected by the installation. Library requirements for VTAM under DOS/VS are described in Figure 7-6.

Sharing Telecommunication Resources

Sharing resources among several users can reduce the amount of time that a user must wait for system resources and permit more efficient use of these resources.

Name	VTAM Contents	When Created	JCL	Comments
		VTAM Data Sets		
SYS1.VTAMLIB	Load modules for VTAM; see note 1	System generation	Yes	Only the VTAM load modules are required
SYS1.VTAMLST	VTAM definition statements and start options	Prior to starting VTAM	Yes	Required. See note 2
SYS1.VTAMOBJ	VTAM network configuration table	When VTAM is started	Yes	Required. See note 3
		Operating System Data Sets		
SYS1.DUMP	Records for SVC dumps	IPL	No	Optional
SYS1.LINKLIB	VTAM initialization module, used when VTAM is started	System generation	No	Required
	NCP loader utility program	Prior to starting VTAM	Yes	Required
	NCP dump bootstrap program	Prior to starting VTAM	Yes	Required
	Local communications controller pre-IPL testing modules	NCP generation	Yes	Required if the controller is to be tested automatically prior to the loading of the NCP by VTAM
SYS1.LOGREC	VTAM error records	System generation	No	Required
SYS1.LPALIB	VTAM load modules, user-written authorization and accounting exit routines, and logon mode and interpret tables to be loaded into the shared link-pack area	System generation	No	Required. An OS/VS2 data set only
SYS1.MACLIB	VTAM macro definitions	System generation	No	Required
SYS1.NUCLEUS	VTAM resident SVCs and abnormal termination modules	System generation	No	Required
SYS1.PARMLIB	VTAM space parameter for IPL	System generation	No	Required. An OS/VS1 data set only
SYS1.PROCLIB	VTAM cataloged procedure	System generation	No	Required
SYS1.SVCLIB	VTAM non-resident SVCs and ERPs for locally attached devices	System generation	No	Required. An OS/VS1 data set only
SYS1.TRACE	GTF trace records for VTAM	System generation	No	Optional

Notes:

1. SYS1.VTAMLIB is referred to as the *VTAM load module library.* For OS/VS1,SYS1.VTAMLIB contains VTAM load modules for the VTAM virtual partition; for OS/VS2, it contains VTAM load modules for the VTAM private address space. In OS/VS1, this library can contain the interpret table, or tables, containing logon descriptions and any installation-coded logon-routines specified in these tables; logon mode tables; and authorization and accounting exit-routines. In OS/VS1 and OS/VS2, this library can contain the network solicitor.
2. SYS1.VTAMLST is referred to as the *VTAM definition library.* Definition statements for each major node are filed in this library. Each major node is a separate member of SYS1.VTAMLST.

   Start options are filed in members under the name of either ATCSTR*xx* or ATCCON*xx* (where *xx* are two-digit numbers specified by the installation). ATCSTR*xx* members contain lists of major nodes to be activated automatically when VTAM is started.

   See "VTAM Definition", in Chapter 3, for details on VTAM definition statements and VTAM start options.
3. When a major node is activated, VTAM builds a table describing the node from the information supplied by definition statements. When a major node is deactivated, its table is deleted by VTAM. These tables are used to maintain the current status of all minor nodes in the telecommunication system.

   To reduce the time needed to construct these tables, VTAM stores a copy of each table on SYS1.VTAMOBJ the first time each major node is activated. This copy is then used whenever the major node is again activated.

   Note: A major node can be modified merely by modifying its definition statements and refiling them under the same member name on SYS1.VTAMLST. If a member is refiled, the copy of the corresponding table on SYS1.VTAMOBJ must be deleted (using an operating-systems utility program that can delete a memberof a BPAM data set). If the copy is deleted, the next time the major node is activated, VTAM builds a new table (based on the modified definition) and stores a new copy on SYS1.VTAMOBJ.

Figure 7-4. (Part 1 of 2). Table of Data Sets Used by VTAM Under OS/VS

Name	VTAM Contents	When Created	JCL	Comments
		NCP Data Sets		
NCP dump	Dump records for the NCP	When VTAM is started	Yes	Required if VTAM is requested to provide a dump of an NCP. One such data set must be provided for each NCP that is to be dumped simultaneously.
NCP load library	NCP load modules	NCP generation	Yes	One such data set must be created for each NCP used by VTAM. See note 4
		TOLTEP Data Sets		
TOLTEP CDS	Configuration data sets	Prior to invoking TOLTEP	Yes	Required if TOLTEP is to be executed
TOLTEP OLTs	Online terminal tests for TOLTEP	Prior to invoking TOLTEP	Yes	Required if TOLTEP is to be executed

Notes:

4. These data sets are referred to as the *NCP load module libraries.*

Figure 7-4 (Part 2 of 2). Table of Data Sets Used by VTAM Under OS/VS

File Name	File Identification	Content	Comments
DIAGFILE	INITST	Communications controller pre-IPL testing modules	Required if the controller is to be tested automatically prior to the loading of the NCP by VTAM
NCP load file	Note 1	NCP load modules	One such file must be created for each NCP used by VTAM. See note 2
NCPDUMP	Note 3	Dump records for the NCP	Required if VTAM is requested to provide a dump of an NCP. One such file must be provided for each NCP that is to be dumped simultaneously
TRFILE	TRFILE	VTAM trace records	Required. Must be assigned to SYS001

Notes:

1. File names and file identifications are created during NCP generation.
2. These files are referred to as the *NCP load libraries* in this publication.
3. File identifications are created during NCP generation.

Figure 7-5. Table of Files Used by VTAM Under DOS/VS

DOS/VS Library	Content	Comments
Core image library	VTAM modules; NCP tables and block handler routines; configuration data sets and online terminal tests for TOLTEP	Required. See Note 1
Procedure library	VTAM start procedure	Optional
Source statement library	VTAM definition statements and start options	Required. See note 2

Notes:

1. In this publication, *VTAM load module library* refers to the VTAM contents of this library. This library also contains the interpret table, or tables, containing logon descriptions and any installation-coded logon-routines specified in these tables; authorization and accounting exit routines; and the network solicitor.
2. In this publication *VTAM definition library* refers to the VTAM contents of this library. Definition statements for each major node are filed in this library. Each major node is a separate book of the source statement library.

Figure 7-6. VTAM's Use of Libraries Under DOS/VS

VTAM permits resources in its telecommunication system to be shared. The degree of sharing depends upon such factors as the resources themselves and the installation's management of these resources.

Resources That Can Be Shared

Using VTAM facilities, application programs can share terminal components, terminals, cluster control units, communications controllers, NCP facilities, telecommunication lines, and data channels. Of these elements, terminals (or components) and application programs can be thought of as end nodes, while the remainder of the elements are part of the path connecting the end nodes.

Sharing of end nodes is different from the sharing of path elements. End nodes are shared for each connection; path elements can be shared simultaneously. That is, an active terminal (an end node) can be connected to or queued for logon to only one application program at a time. If an active terminal is not connected or queued for logon to an application program, it is available for connection to any application program.

On the other hand, VTAM does not explicitly connect path elements to end nodes. Instead, VTAM employs path elements on behalf of end nodes only as long as needed to complete a specific I/O request. For example, more than one terminal can be attached to the same multipoint line, and each terminal can be connected to, or queued for logon to, a different application program. Thus, the line (path element) is used simultaneously by several applications.

Sharing and Managing Resources

While VTAM allows many resources to be shared, the installation can restrict some sharing for such reasons as performance or security. For example, a high-priority terminal might be attached to a nonswitched, point-to-point, line to improve access to it, or connection to a security-sensitive application program might be limited to only certain terminals.

An installation can affect resource sharing when:

- Defining the network
- Generating the NCP

- Executing application programs

Resource management for each of these areas is discussed below.

Managing Resources through VTAM Definition

The telecommunication network that VTAM manipulates is the one presented to it through network definition. VTAM provides a number of facilities in network definition that enable an installation to affect the sharing of telecommunication resources. These facilities include:

- Logon (automatic and terminal-initiated)
- Application program authorization
- Authorization exit routine
- Initial status of nodes

The effects of each facility on sharing are discussed below. Details on these facilities are provided in Chapter 3.

Sharing Through Logon: Using the terminal-initiated logon capability of VTAM, an installation can make all input/output terminals available to all application programs. A network can be defined in which application programs can connect with input and output terminals only in response to a terminal-initiated logon. (Input-only and output-only terminals are available for connection by other means, such as automatic logon or acquisition).

In practice, some of the terminals might not be available to all application programs. For reasons such as security, some of the terminals might be available only to a restricted set of application programs.

Restricting the availability of a terminal can be accomplished by:

- Limiting the number of valid logon messages for a terminal that is to use VTAM's terminal-initiated logon capability
- Making the terminal available for connection only through the SIMLOGON macro instruction or the ACQUIRE option of the OPNDST macro instruction, and then controlling which application programs can issue those macro instructions
- Prohibiting some connections through the installation-coded authorization exit routine

Application Program Authorization: Some facilities in VTAM require prior authorization before they can be used by an application program. These facilities can reduce the availability of resources and should therefore be used judiciously:

- Passing connections: Passing a terminal to another application program queues the terminal to that application program. As long as the terminal is on the queue, it is not available for communication, either to that application program or to any other application program. Care should be taken to ensure that terminals are on the connection queue no longer than necessary.
- Block processing for start-stop and BSC terminals: As compared to message or transmission processing, block processing can result in a greater number of I/O interruptions for the application program. More interruptions, in turn, can result in more delays for path elements such as telecommunication lines and data channels. These delays then decrease the availability of these resources to other application programs.

Authorization Exit Routine: Although VTAM allows any terminal to be connected to any application program, an installation can code an authorization exit routine restricting

specific connections. VTAM passes control to this exit routine during the processing of connection and disconnection requests.

Initial Status of Nodes: Using VTAM definition statements, an installation specifies the initial status of each terminal and cluster minor node in the VTAM system. These minor nodes are defined as initially active or initially inactive. If a minor node is defined as initially inactive, the network operator must explicitly activate it (with VTAM's VARY command) before it can be used by VTAM. On the other hand, if a minor node is defined as initially active, network-operator intervention is not required, and the node may be made available sooner.

Whatever the initial status specified for a minor node, the minor node cannot be activated until its major node is active. Like minor nodes, major nodes can be activated explicitly by the network operator, or they can be activated automatically (when VTAM is started) without network operator intervention.

Managing Resources through NCP Generation

Because VTAM uses the NCP to communicate with remotely attached devices, options generated in the NCP may affect VTAM's sharing capabilities. For example, if an NCP session limit specified for a multipoint line is not adequate to service all devices on that line simultaneously, some devices may be unavailable for communications.

Refer to the *NCP Generation* manual for more information on NCP options.

Managing Resources through Application Programs

Once a VTAM system has been generated and defined, the availability of resources can be influenced by the application programs' use of VTAM. Although Chapter 5 discusses the functions that can be performed in an application program using VTAM, this section summarizes some of the ways by which an application program can affect the degree of sharing.

As noted earlier, an application program can monopolize telecommunication resources that normally would be shared:

- As long as a terminal is connected to or queued for connection to an application program, it is unavailable to other application programs.
- As long as an application program and a terminal are connected, the number of NCP sessions available for a multipoint line (in the case of remotely attached terminals) is reduced by one.

Connection and NCP session, as they pertain to sharing resources, are discussed below.

Connection and Terminal Sharing: To achieve maximum use of terminals, the installation can instruct application programmers to ensure that terminals are connected, or queued for logon, only as long as necessary. Using VTAM's OPNDST and CLSDST macro instruction, an application program can connect with a terminal only when an actual input or output operation is needed. This feature is particularly useful for application programs with very long delays between message transmissions. If a program handling such applications could tolerate noncontinuous connections, terminals could be released during the delays for use by other application programs.

NCP Session and Line Sharing for Start-Stop and BSC Terminals: While an application program and a terminal are connected in basic mode, it is possible for them to monopolize the telecommunication line. The first I/O request from an application program for a terminal connected in basic mode initiates an NCP session for the terminal. This session is continued until the application disconnects the terminal. Thus, for most application programs, the duration of an NCP session nearly equals the duration of a connection.

The relationship between connection and the NCP session is particularly significant if the line is multipoint and the session limit (as specified in NCP generation) is less than the number of terminals on that line or if the system is DOS/VS and the line is point-to-point with attached components.

In the case of multipoint lines, it is possible for the number of simultaneous connections to equal the specified NCP-session limit. If they are equal and the session limit is less than the number of attached terminals, communication with a terminal not currently in session is prohibited until a current session on the line is ended by the disconnection of another terminal. If the session limit for a multipoint line equals the number of terminals on that line, contention for the NCP session is reduced.

In the case of components on a point-to-point line (under DOS/VS), only one session is permitted at a time for that line.

VTAM Telecommunication Security

This section discusses how to establish and maintain telecommunication security in a VTAM system. It identifies and describes the facilities in VTAM that help an installation to control access to privileged or sensitive data and resources.

Introducing Telecommunication Security

Using VTAM facilities, an installation can identify and control sensitive resources. Essentially, an installation can identify and control the use of specified terminals and application program.

Identification of resources to be controlled is performed during VTAM definition; control of the access to these resources can be performed at various points in the telecommunication network and during various stages of telecommunication processing. The installation specifies when this control is to be exercised.

Depending upon the desired type of control, an installation can specify control of sensitive resources in one or more of the following:

- The communications controller network control program (NCP)
- VTAM
- An installation-coded authorization exit routine
- Application programs

A VTAM installation can control sensitive resources by:

- Verifying terminal identifications
- Verifying logon requests
- Controlling the connections between selected application programs and terminals
- Controlling which application programs can use VTAM
- Restricting an application program's use of selected VTAM facilities
- Protecting sensitive data being transmitted by VTAM

The remainder of this section discusses each control facility in detail.

SNA Terminal Identification Verification on Switched Lines

The IDs of SNA terminals on switched lines are built from the PU statement. VTAM constructs a 48-bit station ID for a physical unit from the PUTYPE, IDBLK, and IDNUM operands. The SNA terminal ID is of the form:

- Bits 0-7: The physical unit type (PUTYPE)
- Bits 8-15: Reserved (X'00')
- Bits 16-27: A 12-bit binary block number assigned by IBM uniquely to the specific device (IDBLK)
- Bits 28-47: A 20-bit binary identification number assigned to the particular station being defined (IDNUM)

After a physical unit either logs on, dials in, or answers a dial-out request from an application program, an XID command (exchange IDs) is sent to the physical unit. The physical unit responds with its ID which is sent to VTAM for verification. If the ID is valid, VTAM sends its ID to the physical unit by the ACTPU command (activate physical unit) and gives the NCP (if the physical unit is a remote terminal) information to support a session (such as physical unit type and segment size used by the physical unit). If the ID is not valid, VTAM will disconnect the physical unit. When the identifications have been exchanged, normal connection procedures continue.

Host ID Verification for SNA Physical Units

VTAM allows a host identification number to be specified when VTAM is started, either in a start procedure or by the network operator. This host identification number or SSCPID can be used by a physical unit when it is initiating connection to the host to determine that it is in touch with an authorized host CPU. Use of the host identification number depends on each IBM terminal or terminal system product; see the programming publications for each product to determine use of the host identification number.

BSC and TWX Identification Verification on Switched Lines

Using the IDLIST and VIDLIST definition statements, an installation can identify binary synchronous communications (BSC) or TWX terminals attached to switched lines. Both definition statements are placed in the generation (and definition) deck for the communications controller network control program (NCP) for virtual systems. The IDLIST definition statement indicates those identifications to be verified by the NCP. The VIDLIST statement indicates those identifications to be verified by VTAM in the host computer. (See "Network Control Program Requirements," earlier in this chapter, for more information on these definition statements.)

In VTAM under OS/VS, an installation can distribute verification authority between an NCP in a communications controller and VTAM in the host computer for dial-in operations. Thus, to conserve NCP resources, an installation might specify in an IDLIST statement only those terminal identifications that are used heavily, and let VTAM verify less frequently used terminals. In VTAM under DOS/VS, verification for dial-in operations must be done by VTAM in the host computer and verification for dial-out operations must be done by the NCP.

Using IDLIST definition statements, an installtion can specify the following information for the NCP:

- The identification character string for each BSC or TWX terminal on a switched line to be verified by the NCP.
- The symbolic name to be assigned to the terminal entering a verified identification character string. A unique name can be specified for each verifiable character string.
- The action to be taken if the NCP encounters a character string that is not described in an IDLIST definition statement. The NCP can either pass the unverified identification to VTAM for further verification processing or stop communicating with the terminal.

Using The VIDLIST definition statement, an installation can specify the following information for VTAM:

- The identification character string for each BSC or TWX terminal on a switched line to be verified by VTAM.
- The symbolic name to be assigned to the terminal entering a verified identification character string. A unique name can be specified for each verifiable character string.

If VTAM is unable to verify a terminal identification, it disconnects the terminal or passes it to the application program designated to handle unidentified terminals. VTAM only disconnects the terminal if it cannot pass the terminal to an application program. An application program is designated to handle unidentified terminals by supplying special operands on a TERMINAL statement in the NCP definition deck. These special operands indicate:

- That the TERMINAL statement contains terminal descriptions to be applied to unidentified terminals. (This statement contains the CTERM=YES parameter and the UTERM specification.)
- That whenever such a terminal description is applied to a terminal, that terminal should be automatically logged on to a specified application program.

Any further verification of the terminal is then the responsibility of that application program. See "Network Control Program Requirements," earlier in this chapter, for more information on the TERMINAL statement, including discussions on CTERM=YES and on UTERM.

To accomplish identification verification for dial-in operations in a VTAM system under DOS/VS, the IDLIST definition statement should not be specified, but the VIDLIST definition statement should be coded. Thus, all identification character strings are passed to VTAM for verification. For dial-out operations, the IDLST statement (rather than the VIDLIST) is coded since verification is performed by the NCP.

To accomplish identification verification in a VTAM system under OS/VS, three options are available:

- Use the method described for VTAM under DOS/VS.
- Specify IDLIST definition statements to describe all identifications to be verified by the NCP. Specify in an IDLIST definition statement that all unverified identifications are to be passed to VTAM. Code VIDLIST definition statements to handle the verification of all identifications not verified by the NCP.
- Code only IDLIST definition statements, verifying all identifications in the communications controller, and pass no unverified identifications to VTAM.

Verifying all identifications in the host CPU relieves the NCP of verification requirements and reduces NCP space requirements. On the other hand, verification in the host increases VTAM's requirements for virtual storage and for CPU time. This option is probably best suited for systems having only a small number of terminals to be verified and expecting only infrequent verification activity. It is also suited for systems whose NCP space and performance requirements are significantly more important than those of the CPU and VTAM.

Dividing the verification responsibility between the NCP and VTAM offers flexibility. The exact division can be weighed against space requirements, processing-time considerations, and verification usage.

Verifying identifivations only in the NCP reduces VTAM and CPU requirements, and can free the access method of some processing requirements. This option might be selected for those systems expecting heavy identification verification.

Controlling Connections

Before a terminal and an application program can communicate with each other in a VTAM system, they must be connected. VTAM provides the following facilities to assist an installation in controlling connections:

- An exit to allow an installation-coded routine to authorize connection dynamically requests during program execution.
- A mechanism to allow an installation to control which applications can acquire a terminal.
- The capability to tailor logon requests to installation requirements and specifications. This facility includes helping the installation to control which terminals and operators can log onto which application programs.

Authorization Exit Routine

VTAM allows an installation to code its own connection-authorization routine. The authorization exit routine is executed each time a logon or OPNDST macro instruction requests connection or a logoff or CLSDST macro instruction requests disconnection.

Upon entry, this exit routine is provided with names of the nodes and the type of request. The exit routine can then determine whether the request is valid.

Upon returning to VTAM, the authorization exit routine indicates to VTAM whether the request should be permitted. If the request is valid, it is completed by VTAM; otherwise, it is rejected. See "Coding and Including Installation Exit Routines," in Chapter 3, for more information on this exit routine.

Acquiring and Passing Connections

An application program can request that a terminal be connected or queued for logon by using one of the following methods:

- OPNDST macro instruction with the ACQUIRE option
- SIMLOGON macro instruction
- CLSDST macro instruction with the PASS option

In VTAM, the installation controls these options; that is, only application programs authorized by the installation during VTAM definition can issue these macro instructions. Unauthorized application programs can only be connected in response to logon requests with an OPNDST (with the ACCEPT option) macro instruction.

An installation authorizes an application program in the APPL definition statement.

APPL authorization allows an installation to control which application programs can initiate connection requests. The following topic, "Controlling Logon Requests," explains how an installation can control connection requests that originate outside an application program.

Note: *These options can also be controlled through the authorization exit routine; because they are all forms of connection, logon, or disconnection, these options schedule the authorization exit routine. See "Authorization Schedule Routine," above.*

Controlling Logon Requests

VTAM definition procedures and the authorization exit allow an installation to control which terminals can log onto which application programs.

To permit terminal-initated logons in VTAM, an installation specifies the following during VTAM definition:

- For the terminal: Specifies in the GROUP, LINE, CLUSTER, VTERM, TERMINAL, or LU definition statement (for remotely attached terminals) and in the LOCAL definition statement (for locally attached terminals), that a terminal is to be monitored by VTAM for logon requests and indicates the interpret table containing the logon description to be used when monitoring logon requests from that terminal.
- For the application program: Specifies in the APPL definition statement, the name of the application program.
- For the logon message: Uses the INTAB, ENDINTAB, and LOGCHAR macro instructions to define interpret tables. Each interpret table contains descriptions of the valid logon messages that can be issued from each terminal and indicates which program is to receive the connection request for each message. (See "VTAM Definition," earlier in this chapter, for details on these macro instructions.)

Using the logon information supplied by the installation at VTAM definition, VTAM's logon facilities determine which logon requests should be routed to which application programs. In addition to controlling this determination, the user has two other opportunities to inspect and control the pending connection request. These opportunities are in the installation-coded authorization exit routine and in the application program to which the logon request is directed.

The installation-coded authorization exit routine is notified of all terminal-initiated logon requests. Upon entry, this exit routine can make a further determination as to whether the connection is valid. If invalid, the connection request is rejected by VTAM; if valid, the request is passed to the application program. (See "VTAM Definition," in Chapter 3, for a description of the authorization exit routine.)

Finally, having passed VTAM's logon facilities and the authorization exit routine, the logon request must still be accepted by the application program before a connection is established. The application program can inspect the logon request and either accept the request or reject it. VTAM provides a number of tools for verifying logon requests in the application program. Using these tools effectively requires planning and preparation on the part of the installation at VTAM definition. It requires that specific procedures be established for the verification of logon requests in the application program, and it requires that application programs that can accept logon requests contain logon exit routines adhering to these installation-established procedures. (Though an application program can accept a logon request without using a logon exit routine, the program would have access to the logon message only through the exit routine.)

For an application program to provide logon verification effectively, the installation should:

- Establish the format and content of permissible logon messages for each application program. Permissible messages might be required to contain terminal-user identifications and passwords.
- Use the INITAB, ENDINTAB, and LOGCHAR macro instructions to define these messages to VTAM. (The LOGCHAR macro instructions are also used to specify which application program is to receive notification for each logon request.)
- Use the VTERM, TERMINAL, LU, or LOCAL and the APPL definition statements to identify valid terminals and application programs to VTAM.
- Use the GROUP, LINE, CLUSTER, VTERM, TERMINAL, LU, or LOCAL definition statements also to specify which interpret table is to be used for each terminal.

- Modify, if necessary, and activate VTAM's network solicitor (for start-stop and BSC terminals).

Once the installation has prepared the telecommunication system for terminal-initiated logons, the application program can provide a final authorization check. To assist in this final check, VTAM provides:

- A logon-exit in the application program: This exit is triggered whenever a logon request is to be passed to the application program.
- A means of obtaining the logon message: VTAM's INQUIRE macro instruction can be used to obtain the logon message. If installation-defined procedures specify that the message contain required information (such as a user identification and a password), this information can be inspected for accuracy and validated.
- A means of determining device type: VTAM's INQUIRE macro instruction can also be used to determine the type of device from which the message was received. If the application program is not equipped to handle such devices, the request can be rejected.
- A means of identifying the specific terminal: Input to the logon exit routine includes the symbolic name of the terminal. The name is established by the installation at VTAM definition. Through installation-defined procedures, valid terminal names can be made known to each application program. Criteria for accepting or rejecting logon requests from specific terminals can then be established by the installation.

Thus, for a terminal to log on to an application program using VTAM's logon facilities, the following conditions must be met:

- The terminal, the logon message, and the application program must be defined by the installation at VTAM definition.
- An interpret table containing valid logon messages must be created for each terminal, unless the standard logon message is to be used.
- The installation-coded authorization exit routine (if supplied) must authorize the logon request.
- The application program must accept the logon request.

VTAM also allows an installation to control other types of logons. As noted above, the automatic logon is specified at VTAM definition. Although this specification can be altered by a network-operator logon command, all connection requests can be monitored by the installation-coded authorization exit routine. Also, the application program can accept or reject connection requests. Thus, even if an unauthorized connection is requested by the network operator, the request must still pass the authorization checking of the installation exit routine and of the application program.

Application program logon requests are also controlled by the installation. (The application-program logon results from issuing a SIMLOGON or CLSDST, with the PASS option, macro instruction.) This logon request must be initiated by an application program, and it can only be initiated by authorized programs.

Authorization to issue the SIMLOGON or the CLSDST (with the PASS option) macro instructions is provided at VTAM definition via the application program's APPL definition statement. In addition to this authorization, connections resulting from application-program logon requests must pass the installation-coded authorization exit routine check (if supplied) and must be accepted by the application program to be completed.

Security Considerations for OS/VS Logon: If an OS/VS logon can be issued from a local 3270, BSC, or start-stop terminal, that terminal has access to any application program in the VTAM system.

If caution is not exercised when authorizing OS/VS logons, the security of the telecommunication system may be jeopardized. As noted in Chapter 3, an installation can prescribe data to be added to the OS/VS logon message. The application program can then inspect this data to decide whether to accept or reject the logon. In addition, an OS/VS logon must also be validated by the installation-coded authorization exit routine (if any). This routine could be designed to control security-sensitive connections. Therefore, an installation, by requiring password data in the logon message and by coding an authorization exit routine, can prohibit unauthorized use of the OS/VS logon.

Caution should also be exercised when using the network-operator logon facility to log a local 3270, BSC, or start-stop terminal onto the network solicitor. If no interpret table is specified for such a terminal, a terminal operator might then have access to any application program in the VTAM system.

Symbolic Names

As a further aid in controlling connections, VTAM nodes can only be addressed in the active system by symbolic names. The symbolic names are assigned at VTAM definition. Terminal names are provided by the TERMINAL, VTERM, LU, or LOCAL definition statements; NCP group names are provided by the GROUP definition statement; line names are provided by the LINE definition statement; physical unit names are provided by the PU definition statement; cluster controller names are provided by the CLUSTER definition statement; component names are provided by the COMP definition statement; and application program names are provided by the APPL definition statement.

These names must be used by the network operator in addressing any nodes in a VTAM network. The application program name must be used by the application program (in the OPEN macro instruction, see "Controlling Access to VTAM," below) to access VTAM. The names assigned to terminals and components must be used also by the application program to initially access terminal nodes.

Thus, all nodes are symbolically named. Name assignments are controlled by the installation through VTAM definition facilities. These names must be used to address and access nodes within the VTAM network.

See "The Major and Minor Node Structure," in Chapter 3, for more information on assigning symbolic names to nodes.

Controlling Access to VTAM

An application program must open a VTAM access method control block (ACB) before the program can use VTAM resources and facilities. To be opened, this control block must supply an application-program identification for verification by VTAM. The identification is the name of an APPL definition statement specified and filed in a member of the VTAM definition library during VTAM definition by the installation.

The installation may also specify in the APPL statement that any attempt to open an ACB using the name of the statement as an application-program identification must include a password. The password specified in the ACB must agree with that specified in the APPL definition statement if the ACB is to be permitted to open.

In addition to controlling the names of authorized application programs and the specification of passwords, an installation can control an application program's initial use of VTAM. A program's initial use of VTAM (that is, issuing an OPEN macro instruction

for a VTAM ACB) can be restricted by controlling the activation of major nodes. Thus, even if the identification and the password match an APPL specification, the open request is denied if the major node containing the APPL specification has not been activated or if the specification is already in use by another opened ACB that used the same identification and password.

Controlling the Use of VTAM Facilities

Although an application program has been recognized by VTAM (a VTAM ACB has been successfully opened), it may still not have access to all of VTAM's facilities. Some of VTAM's facilities are considered sensitive resources. These facilities, if used without discretion, could affect the integrity or security of the telecommunication system. The installation uses VTAM definition facilities to identify and control these facilities which include:

- The passing of a terminal connection to another application program (CLSDST macro instruction with the PASS option)
- The initiation of a terminal connection using the OPNDST macro instruction with the ACQUIRE option or the SIMLOGON macro instruction
- The use of block processing instead of message or transmission processing (for terminals connected in basic mode only)
- In OS/VS2, the use of authorized path

Passing and acquiring enable an application program to initiate a connection request for a terminal. To help ensure that only authorized connections are made, VTAM allows an installation to control connection requests initiated by application programs.

The authorization specification in the APPL definition statement provides control for block processing and application-program-initiated connection. An application program can use only those facilities specifically authorized in the APPL definition statement associated with its ACB.

An application program can use block processing by specifying the BLOCK option of the PROC operand of the NIB macro instruction. See "Data Blocks" in Chapter 8, for a description of block processing.

The use of authorized path in OS/VS2 requires authorization of a VTAM application program. *OS/VS2 System Programming Library: Supervisor* describes how to specify an authorized program.

Protecting Sensitive Data

When data is transmitted between an application program and a terminal, it passes through VTAM and through NCP buffers. These buffers are obtained from and returned to common buffer pools for each transmission request.

To protect sensitive data, the application program can request that VTAM and NCP buffers be cleared before being returned to the buffer pools. This request is made by specifying the PROC=CONFTXT option of the node initialization block (NIB). (See "The VTAM Language," in Chapter 5, for information on the NIB.)

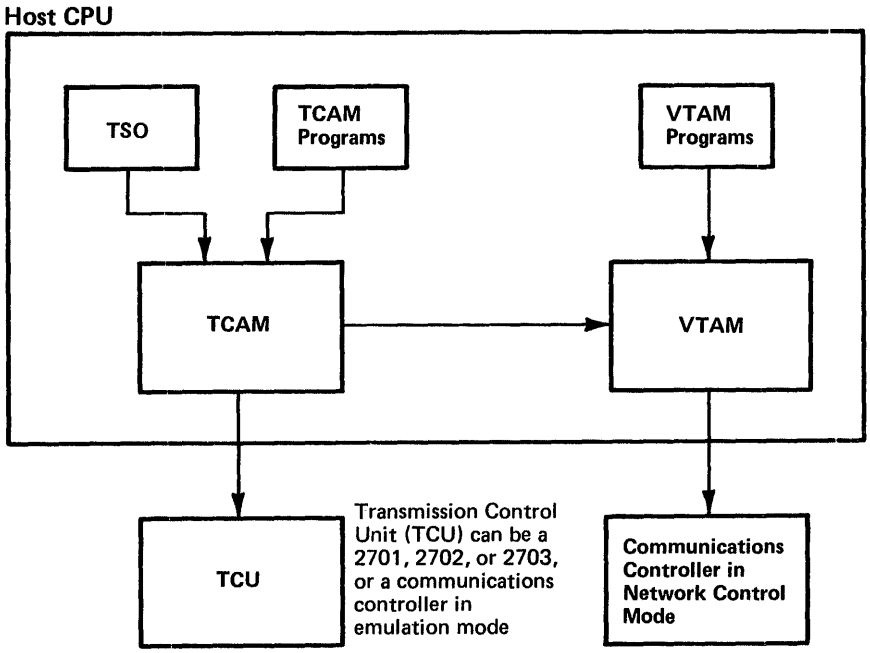
Buffer traces of confidential data produces only the name of the application program, the name of the terminal, and the direction of the data flow. The actual data is not included in the trace records, as in the case of a buffer trace of non-confidential data. The PROC=CONFTXT option is also used by VTAM's trace facility to determine confidential data.

Other Telecommunication Access Methods

VTAM can coexist with QTAM and BTAM under DOS/VS and with BTAM and TCAM under OS/VS. QTAM programs, BTAM programs, and TCAM programs that do not use the communications controller in network control mode can be executed concurrently as long as they have separate telecommunication networks. Additionally, when VTAM and TCAM are both in the operating system, TCAM programs that use terminals attached to a communications controller in network control mode are supported through VTAM. Figure 7-7 shows an OS/VS telecommunication system with TCAM and VTAM being executed concurrently.

TCAM Programs Under VTAM

TCAM application programs and the message control program (MCP) can share the resources of a VTAM telecommunication network with application programs written for VTAM. When sharing a network with TCAM, VTAM processes requests for all remote terminals attached to a communications controller in network control mode and, optionally, requests for locally attahced 3270s. TCAM supports terminals attached to other transmission control units, including those terminals attached to a communications controller in emulation mode (with or without PEP), and locally attached devices. The TCAM user can choose between having individual local 3270 devices supported directly or through VTAM.



VTAM and TCAM use VTAM to communicate with terminals in the VTAM network. TCAM programs, including TSO, can also communicate with terminals attached to a 2701, 2702, or 2703 transmission control unit, or through a communications controller in emulation mode. The same communications controller can be shared by TCAM programs using emulation mode and VTAM programs using network control mode.

Figure 7-7. Communications Controllers and Transmission Control Units in a Telecommunication Network

The principal advantages of this shared capability are:

- The ability to share network resources between VTAM and TCAM programs
- The ability to allow terminals to log onto TCAM
- The availability of the queued-control capability of TCAM in a shared system

Existing TCAM application programs may not require changes, recompilation, or reassembly; their interface with the TCAM MCP remains the same. However, application programs that use TCAM operator control commands should be evaluated to ensure that they will operate as expected in the new environment.

The following TCAM macro instructions are altered for VTAM operation: CODE, INTRO, MSGFORM, MSGGEN, STARTMH, and TERMINAL. An MCP that uses any of these macro instructions must be reassembled.

TCAM in a shared VTAM and TCAM environment depends upon VTAM to exercise physical control over stations and lines attached to a communications controller in network control mode. Differences exist between OS/VS TCAM Release 5, which directly supports the communications controller in network control mode, and the version of TCAM that operates in a shared environment with VTAM. These differences consist of:

- TCAM functions not available in the shared VTAM and TCAM environment
- TCAM operator control functions that have an altered meaning in a shared VTAM and TCAM environment
- TCAM operator control functions that are replaced with similar VTAM functions available only from a system console
- TCAM operator awareness messages that are replaced with VTAM messages routed to a system console rather than to an operator control station or application program

TCAM Functions Not Available in a Shared Environment

Certain functions supported by OS/VS TCAM Release 5 for the network control program are not available in the shared network. Some of these are partially replaced with VTAM functions, while others are not. The functions that are not available are listed below.

- Because VTAM restart after host failure requires a re-IPL and cold start of the NCP, the following items, which are checkpointed by TCAM Release 5 for warm start of the NCP after host failure, are not checkpointed in the VTAM/TCAM shared network. TCAM's checkpoint/restart of its MCP is maintained as with TCAM Release 5.
  - Line and terminal status
  - Service seeking pause
  - Session limit
  - Negative response limit
  - Block-handler sets
  - Transmission limit
  - Modifications to dial digits, polling characters, and addressing characters made as a result of ICHNG and TCHNG macros issued in TCAM application programs

  For communications controller failure without host failure, VTAM provides warm start of the communications controller including items 1-6 listed above. The exception is item 7; that is, changes made by TCAM's ICHNG and TCHNG macros and the Modify BH Set operator command are not warm-started. These changes could be reestablished after the communications controller restart by a user-written TCAM application program which would maintain a record of changes and make the changes based on operator notification that the NCP has been restarted.

- The TCAM operator command to change the dial mode of a switched line between manual and automatic dial ("Change Dial Mode") for lines connected to a communications controller is not supported in the VTAM/TCAM shared system. VTAM allows the user to specify the dial mode of a switched line during network generation.
- The TCAM operator command to set the NCP time and date ("Set 3705 Time and Date") is no longer supported. VTAM provides this function when the NCP is loaded.
- The TCAM operator command to display 32 contiguous bytes of communications controller storage ("Display 3705 Storage") is no longer supported.
- The TCAM Release 5 capability to designate a communications controller as a backup and then switch dynamically to this backup in the event of controller failure, using a warm start, is not provided for the VTAM/TCAM shared network. The operator commands involved ("Activate 3705 Backup," "Switch 3705 Backup," "Switch 3705s") are no longer supported. The VTAM user can manually switch between two communications controllers with appropriate physical switching equipment and use of VTAM's VARY command, but restart in the backup controller is cold.
- The TCAM Release 5 capability to dynamically switch a communications controller through a second Type 2 Channel Adapter to a backup CPU (with the "Switch 3705 Channel Adapter" operator command) is not supported in the VTAM/TCAM shared network. The VTAM user can start a new system in the backup CPU, and re-IPL the communications controller through the second Type 2 Channel Adapter. Operator-initiated switching through a second channel to the same CPU by a toggle switch on the communications controller is still available in the shared network; for more information on this feature see the OS/VS TCAM Programmer's Guide.
- The TCAM Release 5 operator commands for switching between specific and general polling for the 3270 Information Display System ("Activate General Poll," "Deactivate General Poll") are not supported for 3270 systems under the control of VTAM. VTAM always uses general polling for these stations.
- The CUTOFF and MSGLIMIT message handler macro instructions are not applicable for stations managed by VTAM. The CUTOFF operand of the NCP's LINE macro instruction can be used.
- The "Read Full Buffer" support available in TCAM Release 5 for locally attached 3270 Information Display Systems is not available for locally attached 3270 stations managed by VTAM. Users who require this support, described in the OS/VS TCAM Programmer's Guide, should use the IOS local support option available with TCAM.
- The input data from a remote 3270 managed by VTAM does not contain the control unit or station addresses. The input format is the same for the local 3270 and remote 3270. (The output format is unchanged.)

Altered TCAM Operator Control Functions

Certain TCAM Release 5 operator command functions are modified in the VTAM/TCAM shared network environment. In the shared network, VTAM exercises physical control over the network, and certain TCAM functions which previously permitted dynamic physical reconfiguration are now limited to TCAM logical reconfiguration.

The operator commands involved are:

- Activate Station to Receive and Transmit
- Activate Station to Transmit
- Deactivate Station for Receive and Transmit
- Deactivate Station for Receive
- Start Line Transmission
- Stop Line Transmission

- Suspend Transmission
- Release Intercepted Station

The altered TCAM application program macro instructions are HOLD and MRELEASE.

For lines and stations associated with TCAM through VTAM, these commands and macro instructions are effective only for message traffic that is being handled by TCAM. If a line or terminal in the shared network is used only by TCAM, the operator command or macro instruction has the same effect as in TCAM Release 5. If, however, the resource is shared, data that is not handled by TCAM can still reach a station for which data flow has been inhibited by a TCAM function.

VTAM itself provides facilities for activating and deactivating terminals and lines from a system console. These commands can be used to prevent all data flow to or from a station.

The TCAM Release 5 operator commands that display which lines or stations are currently active or inactive are still available in a shared network but they display only the stations and lines activated or deactivated by TCAM for TCAM data. These commands do not display the status of lines and stations activated or deactivated by VTAM commands.

These commands are as follows:

- Display Active Stations
- Display Station Status and Message Numbers
- Display Intercepted Stations
- Display Inactive Line Entries
- Display Inactive Open Lines
- Display Line Status and Message Error Record

TCAM Operator Control Functions Replaced by VTAM Operator Control Functions

Certain TCAM Release 5 operator control functions are replaced in the VTAM/TCAM shared environment by VTAM functions which are available from a system console, and not from TCAM operator control stations or TCAM application programs. The replaced TCAM operator commands are as follows:

- Display 3705 Status
- Activate a 3705
- Deactivate 3705 Line and Terminal
- Dump 3705 Storage
- IPL a 3705
- PEP Switch Line Mode
- Start/Stop BTU Trace
- Change NCP Load Module
- Change Session Limit
- Change 3705 Transmission Limit
- Activate/Deactivate Line Trace
- Change Polling Delay Duration

TCAM Operator Messages Replaced with VTAM Operator Messages

In the VTAM/TCAM shared network, VTAM awareness messages replace the TCAM operator awareness messages present in OS/VS TCAM Release 5. TCAM may route the Release 5 operator awareness messages to operator control stations and TCAM application programs, but the VTAM awareness messages that replace them are directed to a system console. (An installation-designated console can be used in conjunction with MCS and an installation-supplied WTOR exit routine. See Note 2.)

Awareness messages deal with:

- Channel operations
- NCP status
- Line and terminal errors

Replies to TCAM operator commands continue to be routed to the TCAM operator control stations or TCAM application programs that enter the commands. Additionally, the TCAM ERRORMSG message handler macro instruction can be used to route information to TCAM application programs.

Note 1: *The following TCAM operator commands and macro instructions still exercise some degree of physical control over the shared network, and their effect on the VTAM portion of the network should be considered before they are issued:*

- Change 3705 Line Speed.
- Switch 3705 Devices (dial backup for non-SDLC leased lines). If using this TCAM feature, do not use VTAM operator commands to activate or deactivate the lines to these devices.
- TCHNG
- ICHNG

Note 2: *In the above discussion of TCAM operator commands, the term "a system console" denotes either the operating system console or a Multiple Console Supported (MCS) terminal. For details of Multiple Console Support, see the* OS/VS VTAM System Programmer's Guide.

DOS/VS Coexistence

In a DOS/VS system, QTAM, BTAM, and VTAM can operate concurrently. QTAM and BTAM programs do not interact with VTAM. QTAM programs use QTAM, and BTAM programs use BTAM to communicate with:

- Terminals attached to transmission control units
- Terminals attached to communications controllers by line in emulation mode
- Terminals attached locally (BTAM only)

VTAM application programs use VTAM to communicate with:

- Terminals attached to communications controllers by lines in network control mode
- 3270 terminals attached locally
- 3790 terminals attached locally

Lines attached to communications controllers using the NCP with PEP can be used in either network control or emulation mode with an appropriate access method.

Figure 7-8 illustrates concurrent use of QTAM, BTAM, and VTAM in DOS/VS.

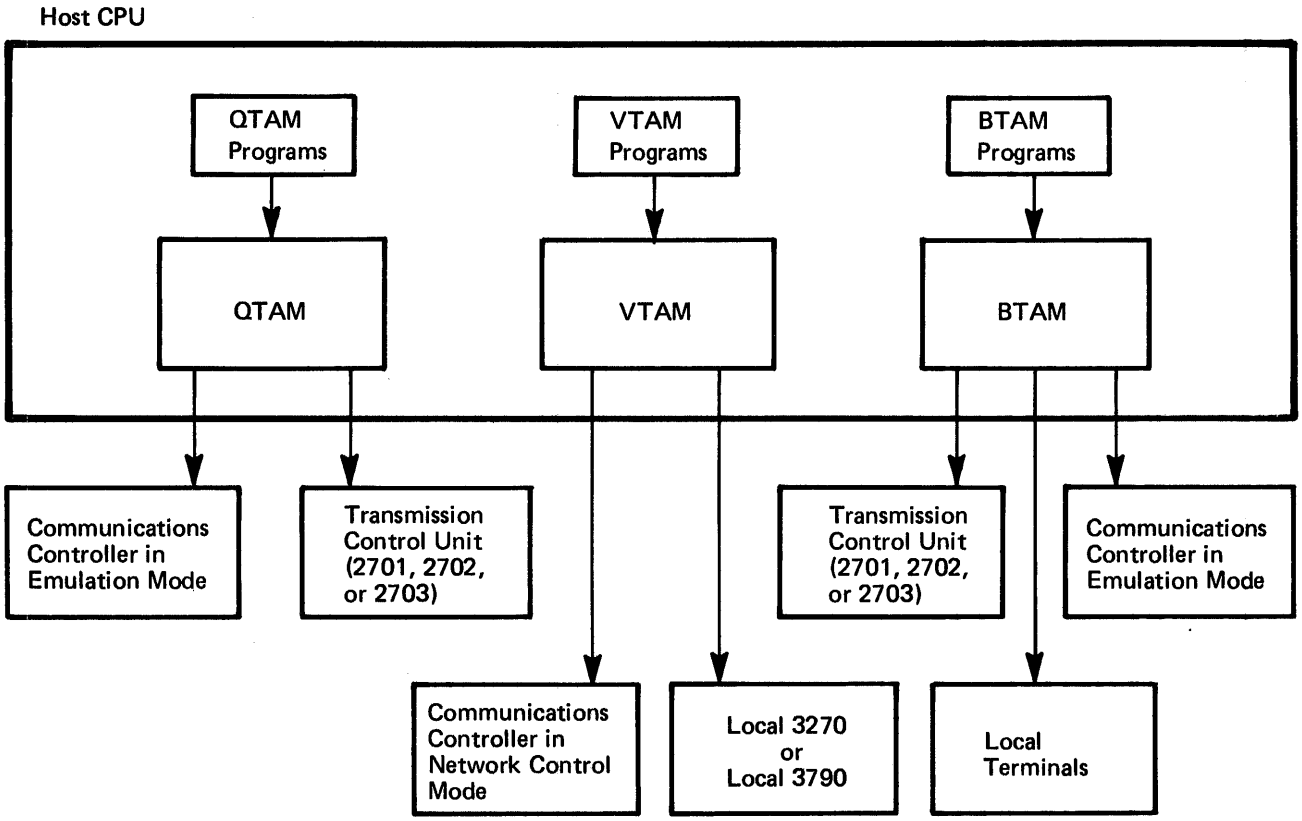
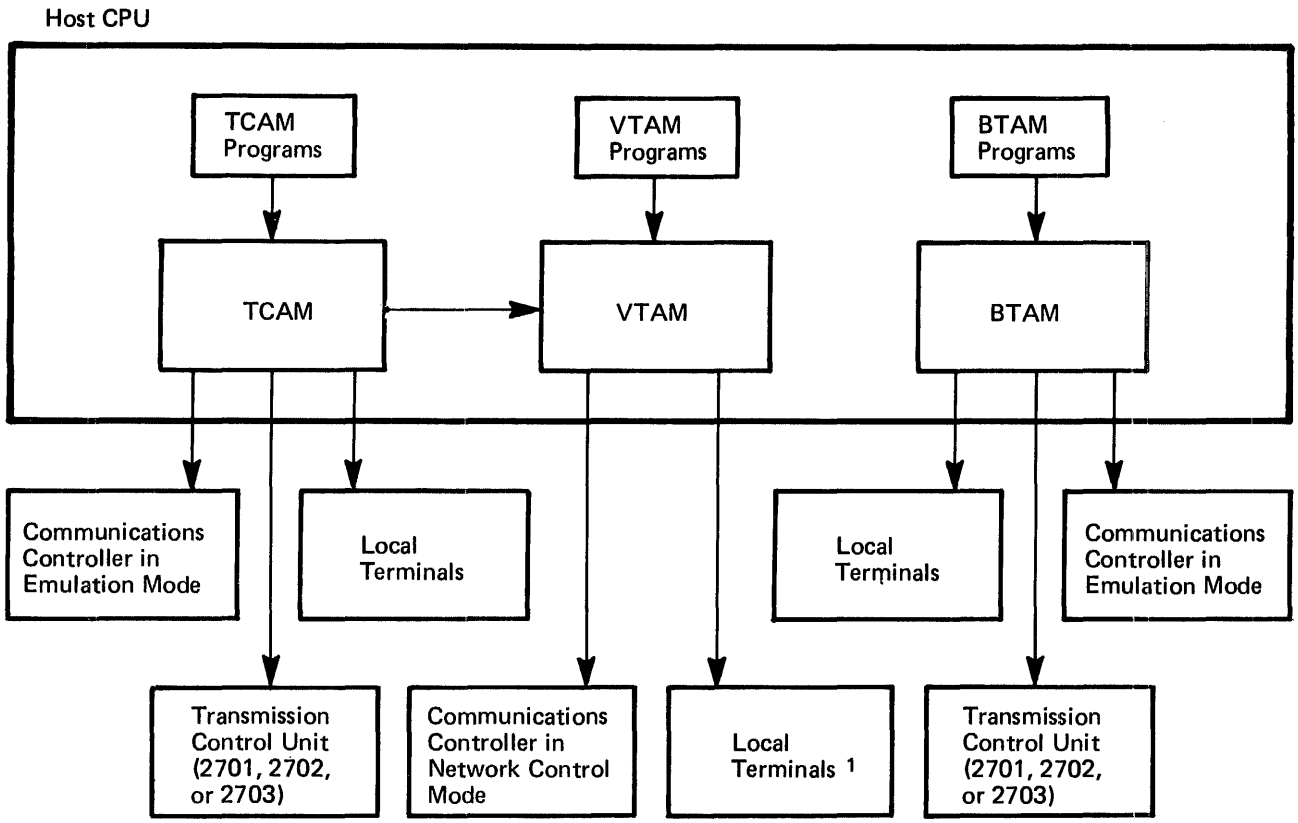


Figure 7-8. DOS/VS Coexistence



[1] VTAM programs can communicate locally with 3270 and 3790 terminal systems. TCAM programs using VTAM can communicate locally with 3270 terminal systems.

Figure 7-9. OS/VS Coexistence

With concurrent execution of the access method, a single application program can use both BTAM and VTAM to communicate with separate networks, provided that all requirements of both access methods are met.

OS/VS Coexistence

In an OS/VS system, BTAM, TCAM, and VTAM can operate concurrently.

BTAM programs use BTAM and TCAM programs use TCAM to communicate with:

- Terminals attached to transmission control units
- Terminals attached to communications controllers by lines in emulation mode
- Terminals locally attached (except 3790 terminals)

TCAM programs use VTAM to communicate with:

- Terminals attached to communications controllers by line in network control mode
- 3270 terminals locally attached

  Note: *TCAM programs can communicate with locally attached 3270s either directly or through VTAM.*

VTAM application programs use VTAM to communicate with:

- Terminals attached to communications controllers by line in network control mode
- 3270 terminals locally attached
- 3790 terminals locally attached

Lines attached to communications controllers containing the NCP with PEP can be used in either network control or emulation mode with the appropriate access method.

Figure 7-9 illustrates the concurrent use of BTAM, TCAM, and VTAM in OS/VS.

With the concurrent execution of the access methods, a single application program can use both BTAM and VTAM to communicate with separate networks, provided that all of the requirements of both access methods are met. In addition, an application program can use both VTAM and TCAM; in this case, the networks can be separate or the same.

CHAPTER 8. SUPPORT FOR LOCAL 3270, BSC, AND START-STOP TERMINALS

In addition to its support for terminals in a Systems Network Architecture environment, VTAM supports the local 3270, BSC, and start-stop terminals listed in Appendix A. The information about VTAM in the previous chapters applies to both its support of SNA terminals and its support of local 3270, BSC, and start-stop terminals, except as noted. This chapter tells what specific facilities or requirements are not applicable to local 3270, BSC, and start-stop terminals and describes the special facilities and requirements for these terminals that are not described elsewhere in this book.

Before deciding whether to use VTAM for these terminals, an installation should consider using BTAM. The installation may wish to continue to use existing BTAM programs, or may wish to modify these programs to interface with new VTAM application programs, or may wish to use BTAM macro instructions in new programs because it is familiar with BTAM.

Using BTAM

Figure 8-1 shows that communication using BTAM can be combined with communication using VTAM in a number of ways:

- An application program that uses VTAM record mode macro instructions to communicate with logical units can also include BTAM macro instructions to communicate with local 3270, BSC, and start-stop terminals.
- An application program that uses VTAM record mode macro instructions to communicate with logical units can use VTAM basic mode macro instructions to communicate with some local 3270, BSC, and start-stop terminals and can also use BTAM macro instructions to communicate with other local 3270, BSC, and start-stop terminals. A program of this nature would be required to communicate with some local 3270, BSC, and start-stop terminals that were supported by VTAM and some that were not supported by VTAM.
- One application program can be used for VTAM communications and another can be used for BTAM communications.

An installation that wants to continue to use terminals not supported by VTAM would have to use one of these combinations. However, even if all local 3270, BSC, and start-stop terminals in a network are supported by VTAM, the installation might want to use BTAM to communicate with them.

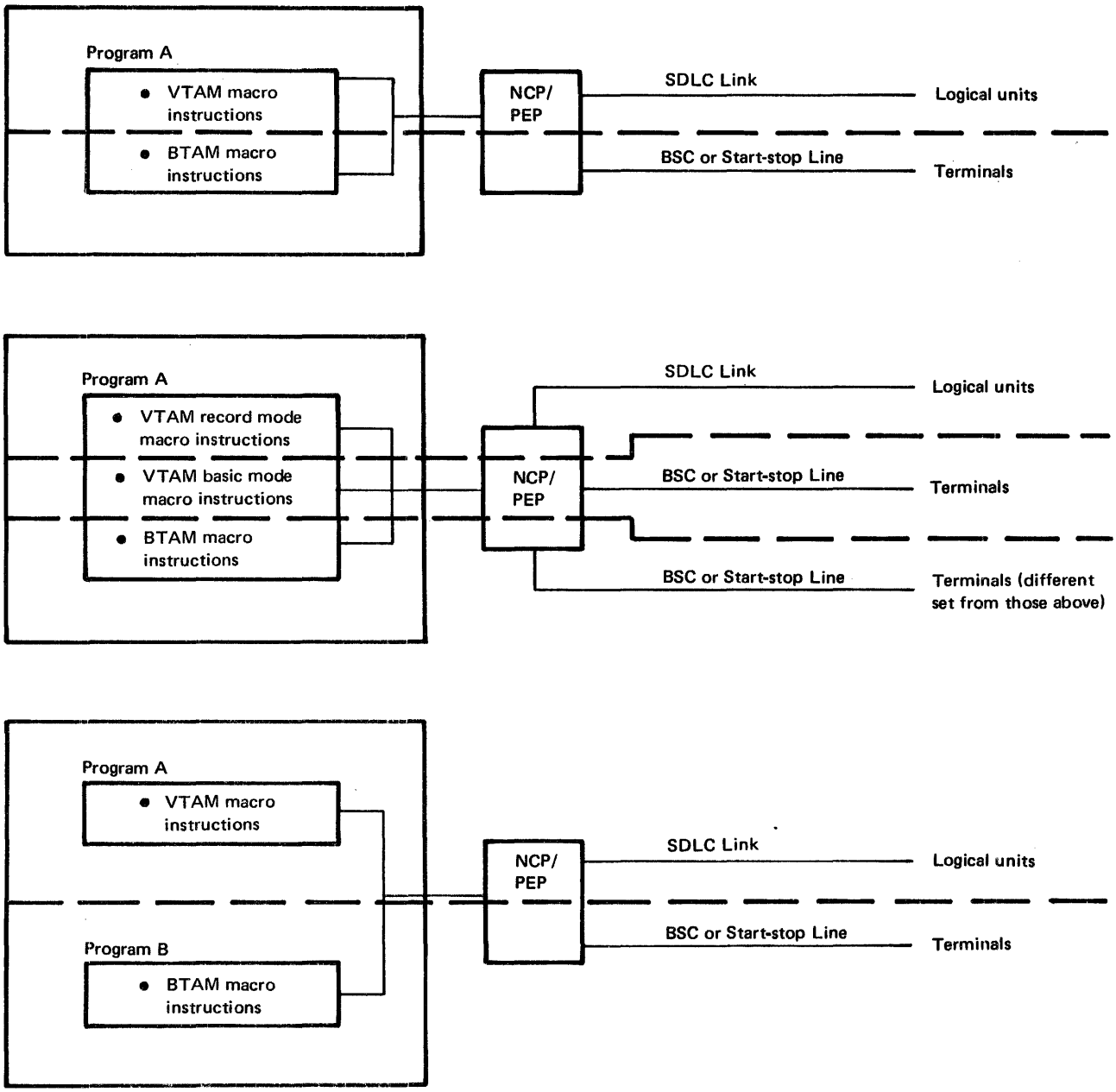
Using VTAM

As for SNA terminals, using VTAM for local 3270, BSC, and start-stop terminals requires:

- Creating a VTAM network (that includes these terminals)
- Operating the network
- Writing VTAM application programs

Except as described below, the information in Chapters 3, 4, and 5, on creating a network, operating the network, and writing VTAM application programs apply for local 3270, BSC, and start-stop terminals. Chapters 6 and 7 also apply, except where noted, to local 3270, BSC, and start-stop terminals. In addition, VTAM provides special support consisting of:

- A network solicitor that can be used to control logon requests



Notes: Local and remote 3270 terminals can be communicated with using VTAM record mode macro instructions, VTAM basic mode macro instructions, or BTAM macro instructions. Using VTAM record mode allows logical units and 3270s to be communicated with using the same set of macro instructions.

Figure 8-1. Using BTAM and VTAM to Communicate with Local 3270, BSC, and Start-Stop Terminals

- Interpret tables that can be defined to specify a name in a logon message and relate the name with a VTAM application program
- Support for start-stop and BSC terminals on switched lines
- A special set of communication macro instructions–SOLICIT, READ, WRITE, RESET, and others–used when writing a VTAM application program

Since the BSC and local 3270 can also use the communication macro instructions that are used for SNA terminals (with certain restrictions), this support is also discussed in this chapter.

Facilities Not Applicable to Local 3270, BSC, and Start-Stop Terminals

In Chapter 3, "Creating a Telecommunication System with VTAM," these topics do not apply to local 3270, BSC and start-stop terminals:

- "Defining Terminal-Initiated Connection" (which includes defining USS definition tables and logon mode tables). Instead, see "Defining Terminal-Initiated Connection" in this chapter.
- "Defining Local Subareas" and "Defining Switched Subareas." These topics apply only to SNA terminals.

In Chapter 4, the topics "Activating a Local Subarea" and "Activating a Switched Subarea" do not apply.

In Chapter 5, the topic dealing with establishing session parameters under the topic "Connection" does not apply. The communication macro instructions–SEND, RECEIVE, RESETSR, and SESSIONC–and the topic "Communication" do not apply to communication with local 3270, BSC and start-stop terminals (unless using record mode support to communicate with local and BSC 3270 terminals). Instead, see "Communicating with Local 3270, BSC, Start-Stop Terminals" in this chapter. Sample Program 1 does not apply; the 3270 support in Sample Program 2 does apply as an example of support for the 3270 terminals using the record mode macro instructions (such as SEND and RECEIVE). See Appendix A in *VTAM Macro Language Guide,* GC27-6994, for examples of program logic using the basic mode macro instructions. See Appendix I in *VTAM Macro Language Reference,* GC27-6995, for specific terminal considerations when writing a VTAM application program to communicate with one or more types of local 3270, BSC, and start-stop terminals.

Creating a VTAM System That Includes Local 3270, BSC, or Start-Stop Terminals

This process consists of generating a network control program (unless only local terminals are in the network) and defining the network configuration and characteristics to VTAM, as described in Chapter 3. In defining connection procedures, an installation may want to understand and define the use of a network solicitor. Optionally, the installation can also use interpret tables to have VTAM interpret a logon request from a local 3270, BSC, or start-stop terminal.

The installation uses the GROUP, LINE, CLUSTER, TERMINAL, COMP, and VTERM statements to define the network to VTAM. An installation can specify:

- Automatic logon and interpret table requirements.
- A description of the features for a BSC 3270 terminal.

- The initial status of a terminal or a cluster control unit when the NCP is activated by VTAM.
- The buffer limit for a teminal. (See "Defining VTAM Buffering" in Chapter 3 for a description of how buffer limits are established.)
- The name of each terminal. The name of the terminal is usually the name of the TERMINAL or COMP statement. If the TERMINAL statement defines a *logical connection terminal* (that is, if it contains the CTERM=YES operand), the name of the terminal must be specified in an additional operand, UTERM. The name specified by UTERM is used only by VTAM and applies to terminals on a switched line. See "Support for Start-Stop and BSC Switched Networks" in this chapter for an explanation of the use of the UTERM name. Refer to the *NCP Generation* publication for an explanation of the CTERM operand.
- The name of each group, line, and cluster control unit (if any). Names are specified in the GROUP, LINE, and CLUSTER statements, respectively.

The Network Solicitor

This section describes VTAM's logon monitor facility for local 3270, BSC, and start-stop terminals (referred to as the network solicitor) and describes how it can be modified.

The network solicitor monitors local 3270, BSC, and start-stop terminals for logons and passes the terminals with valid logons to the appropriate application program. Using the network solicitor, an installation can permit terminal-initiated logons for local 3270, BSC, and start-stop terminals.

How the Network Solicitor Works

Figure 8-2 shows how the network solicitor functions. The network solicitor monitors terminals assigned to it by the automatic logon specification. Terminals are monitored only when they are active but not connected, or queued for connection, to an application program.

When a terminal being monitored by the network solicitor enters a message, the network solicitor determines whether the message is a valid logon request. The message is validated in one of two ways:

- If an interpret table is specified for the terminal, a search is made in that table for an entry corresponding to the message.
- If no interpret table is specified and the operating system is OS/VS, the message is checked for standard format.

See "Defining Terminal-Initiated Logons for Local 3270, BSC, and Start-Stop Terminals" later in this chapter, for a description of defining valid logons.

If the logon is valid, the terminal is passed to the appropriate application program, if the application is active and accepting logons. The application program is the one specified for that logon message in the interpret table, or in the case of an OS/VS logon, it is the application program named in the message itself.

If the logon message is invalid and the application program is not active, or if the application program is not accepting logons, the terminal operator is notified that the logon has been rejected and is invited to enter another logon message.

Modifying the Network Solicitor

A version of the network solicitor is automatically included in VTAM during system generation. This network solicitor has the name NETSOL and can release terminals to requesting application programs (as explained under "Network Solicitor Release Request," below). It can also be started and stopped by VTAM's network operator facilities.

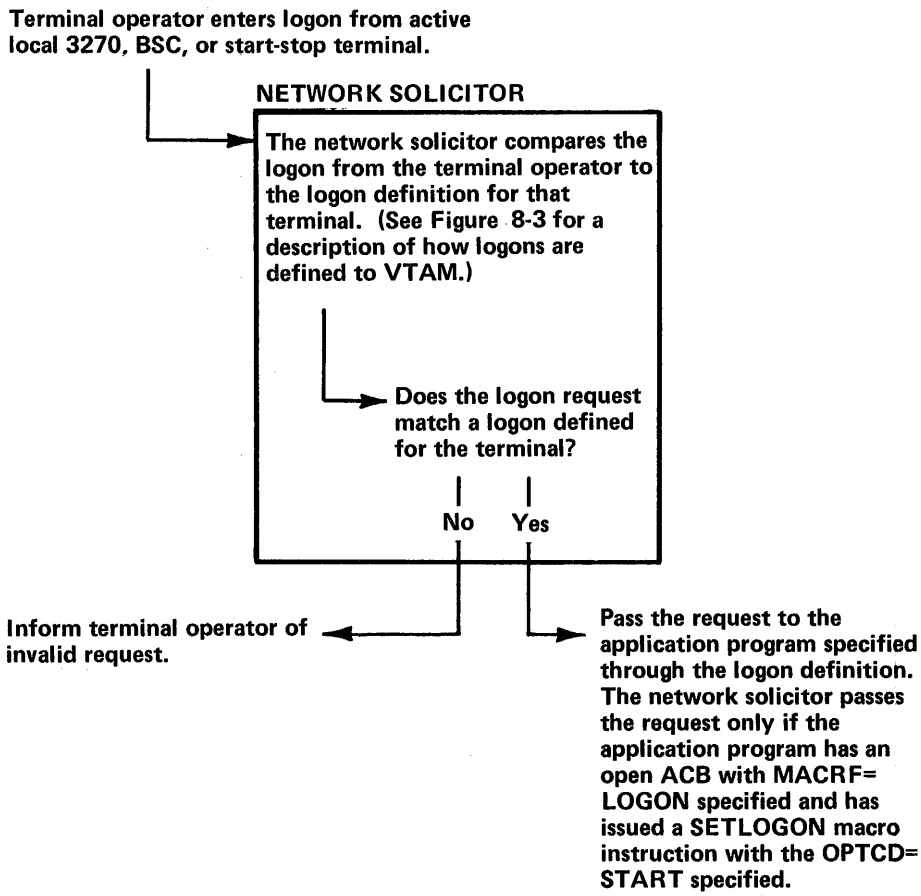


Figure 8-2. Processing a Terminal-Initiated Logon with the Network Solicitor

An installation can retain this network solicitor, modify it, or replace the network solicitor with its own logon monitor. If the VTAM network solicitor is to be used, the installation need only establish logon capabilities for local 3270, BSC, and start-stop terminals as described in "Defining Terminal-Initiated Logons for Local 3270, BSC, and Start-Stop Terminals" later in this chapter. The network solicitor can be modified through the use of VTAM's NETSOL macro instruction as explained below. Replacing VTAM's network solicitor with an installation-written logon monitor is discussed below under "Replacing the Network Solicitor".

The network solicitor is modified by coding, assembling, and link-editing VTAM's NETSOL macro instruction. If a NETSOL macro instruction is not used to replace the network solicitor, VTAM's network solicitor remains available for use. The modified network solicitor can replace, or be used in addition to, the default network solicitor. In DOS/VS, the modified network solicitor should be cataloged in the same library as the one it replaces. In OS/VS, if the modified network solicitor is to be a replacement, it must be link-edited with the VTAM modules in the VTAM load module library.

Using the NETSOL macro instruction, an installation can tailor the following facets of the network solicitor:

Network solicitor name. The installation can change the name of the network solicitor.

Messages. The network solicitor writes messages to the terminal if unusual conditions are encountered while processing a logon. The IBM-supplied messages can be replaced by ones specified by the installation.

Release request. The installation can specify whether the network solicitor should release terminals to application programs that are attempting to acquire them.

Password. The installation can specify a password to be included in the network solicitor's ACB.

More detailed information on the NETSOL specifications is provided below.

Network Solicitor's Name: The name of the VTAM-supplied network solicitor is NETSOL. If modified network solicitor is to replace the VTAM-supplied network solicitor, this name can be retained. (*Note:* Only the network solicitor that runs in VTAM's partition or private address space can use the name NETSOL.) If any other name is specified, the modified network solicitor is treated as an application program by VTAM. That is, the network solicitor must run in its own partition or private address space, the installation must supply an APPL definition statement for it, and it must be started and stopped like an application program. VTAM's start options and MODIFY command cannot be used to start or stop a network solicitor with a name other than NETSOL. See Chapter 4 for using the start options or the MODIFY commands with the network solicitor.

The name specified on the NETSOL macro instruction is the name in the automatic-logon specification for each terminal to be monitored by the network solicitor.

The load module name of the network solicitor is ISTNSC00; this load module name must be used when modifying the VTAM-supplied network solicitor.

Network Solicitor Messages: The network solicitor issues a message if any of the following conditions is encountered:

- The application program specified in the logon request is unavailable for logons. The application program is unavailable if it is inactive, closing down, or not accepting logon requests.
- The logon message is invalid; that is, it does not match any entry in the specified interpret table or (for OS/VS only) it is not in the OS/VS-defined format.
- No interpret table is specified (DOS/VS only) or no interpret table is available for the terminal. (Interpret tables are discussed later in this section.)
- The telecommunication system is closing down.
- An input error is encountered.
- The logon request is rejected by the authorization facilities of VTAM.
- The terminal is not supported by the network solicitor.

For each of these conditions, the installation can replace IBM-supplied messages with its own.

Network Solicitor Release Request: If an installation authorizes application programs to *acquire* terminals, the network solicitor should be able to release, upon request, terminals it is monitoring. If release request is specified in the NETSOL macro instruction, the network solicitor is generated with a RELREQ (release request) exit routine like the one in the VTAM-supplied network solicitor. Whenever this exit routine is scheduled, the network solicitor releases the requested terminal unless a logon request is being processed. (See Chapter 5 for details on the RELREQ exit and how it is invoked and on acquiring terminals.)

Password: If the installation wants the ACB for the modified network solicitor to contain a password, this password must be specified in the NETSOL macro instruction. If

a password is specified, the modified network solicitor is treated as an application program by VTAM; that is, it must run its own partition or private address space, the installation must supply an APPL definition statement for it, and it must be started and stopped like an application program. VTAM's start options and MODIFY command cannot be used to start or stop a network solicitor with a password.

Replacing the Network Solicitor

If an installation does not want to use the IBM-supplied network solicitor, but does want a general-purpose terminal-initiated logon facility for local 3270, BSC, or start-stop terminals, an installation can code an application program to perform the network-solicitor functions. Such an application program would need to monitor terminals for logons and pass valid logons to the appropriate application programs. The monitoring program would be treated like an application program by VTAM: an APPL definition statement would have to be filed for it, and it would have to be activated and deactivated as an application program. (See "Activating and Deactivating Nodes," in Chapter 4, for details on activating application programs.) As an application program, though, the installation-coded monitoring program would still have access to the interpret tables through the application-program INTRPRET macro instruction.

Specifying Interpret Tables

VTAM's INTAB, ENDINTAB, and LOGCHAR macro instructions are used to construct interpret tables (Figure 8-3). For the purpose of defining logon messages, the LOGCHAR macro instruction describes a single logon message. The INTAB and ENDINTAB macro instructions specify a group of logon messages, each message defined by a LOGCHAR macro instruction. This group is called an *interpret table.* Each interpret table must be assembled and filed separately (as a member in OS/VS or a book in DOS/VS) in the VTAM load module library. The name assigned to the member or book is the name assigned (by the INTAB macro instruction) to the interpret table.

Thus, the INTAB and the ENDINTAB macro instruction define a group of logon message definitions and provide a name for that group. The LOGCHAR macro instruction describes a specific message and can be used to indicate the following:

- Whether the logon is requested by a character string or by a 3270 program function key.
- Which program function key, if any, can make the request.
- What character string, if any, can make the request The characters specified in the LOGCHAR macro instruction are the only characters to be checked by VTAM at the beginning of the message. The actual message entered from the terminal can contain additional data to be used, for example, for password protection or accounting by the application program. This additional data must not be specified in the LOGCHAR macro instruction.
- The name of the application program to receive this logon request.

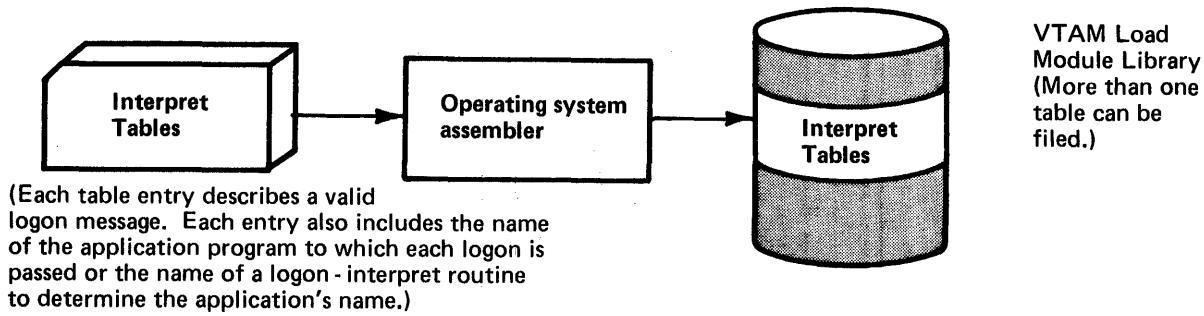


Figure 8-3. Filing Interpret Tables

For each message, the installation can specify in the LOGCHAR macro instruction either the name of an application program or the name of a routine (a logon-interpret routine) that is to determine the appropriate application program. All logon-interpret routines specified in the same interpret table must be link-edited with that interpret table.

VTAM's application-program INTRPRET macro instruction allows access to the contents of the interpret table. The network solicitor uses the INTRPRET macro instruction to validate logon requests. (The macro instruction can be used similarly by application programs.) The following description of the network solicitor's use of the INTRPRET macro instruction and of the interpret tables explains a possible use of VTAM's interpret function.

The network solicitor invokes INTRPRET while specifying a logon message received from a terminal and the name of that terminal. INTRPRET then determines if an interpret table is specified for that terminal. (If one is not specified, INTRPRET indicates to the network solicitor that a table is not specified.)

If a table is specified, INTRPRET checks for a match between the message passed to it and one defined by a LOGCHAR macro instruction for the table. If no match is found, INTRPRET returns to the network solicitor, with an indication that the logon message is not in the table.

If a match is found, INTRPRET determines whether an application program or a logon-interpret routine is specified in the LOGCHAR macro instruction. If an application program is specified, INTRPRET returns the name of the program to the network solicitor.

If a logon-interpret routine is specified, INTRPRET invokes the routine. This routine is installation coded and should validate the logon request. The logon-interpret routine should specify the name of the application program to receive the logon request; otherwise, it should specify that the logon request is invalid. If the output from the routine is valid, it is returned to the network solicitor.

Upon entry to a logon-interpret routine, the following information is available:

- The name of the terminal requesting the logon
- The logon message

Output from a logon-interpret routine should be:

- An indication of whether the logon is valid
- The name of the application program to receive the logon if it is valid

The message from the terminal is given as input to a logon-interpret routine and it can therefore contain more data than is specified in the associated LOGCHAR macro instruction. The routine can use this additional data to determine the application program name. In addition, this data might also contain information such as a password that is verified by the routine.

Although the interpret tables are intended primarily for validating terminal-initiated logon messages, they are also available to application programs through VTAM's INTRPRET macro instruction. See "The VTAM Language," in Chapter 5 for additional information on this macro instruction.

Defining Terminal-Initiated Logons for Local 3270, BSC, and Start-Stop Terminals

To enable a terminal operator to issue a logon request from a local 3270, BSC, or start-stop terminal, the steps are as follows:

1. Modify the network solicitor. (This step is optional.)
2. Define terminal operator logon procedures and messages to VTAM.
3. Activate the network solicitor.
4. Activate the terminal.
5. Enter a logon request to be processed by the network solicitor.

Steps 1 and 2 are done as part of VTAM definition. Steps 3 and 4 are completed by the network operator, although the degree of network-operator involvement depends upon the VTAM definition options selected. (See Chapter 4 for details on activating the network solicitor and terminals.) Step 5 is accomplished by the terminal operator.

To use VTAM's network solicitor, the installation must define:

1. Which terminals are to be handled by VTAM's network solicitor. (Output-only terminals cannot be handled and should not be defined.)
2. What is the format and content of each logon message and what is the name of each application program to be notified for each logon request.
3. Which logon messages can be used by each terminal.

An installation can use VTAM's automatic logon capability to accomplish item 1. Instead of specifying an application program name for automatic logon in a terminal's GROUP, LINE, CLUSTER, VTERM, TERMINAL, or LOCAL definition statement, the installation specifies VTAM's network solicitor. Whenever a terminal so designated is available, the network solicitor monitors it for a logon request. The installation can also use the network operator logon option to temporarily assign a terminal to the network solicitor. The network operator can assign a terminal to the network solicitor by logging the terminal on the network solicitor.

Interpret tables define valid logon messages to VTAM and indicate which application programs are to be notified of the connection request for each valid logon message (item 2). OS/VS also provides a logon message that does not use an interpet table. See "Specifying Interpret Tables," earlier in this chapter, for details on setting up interpret tables.

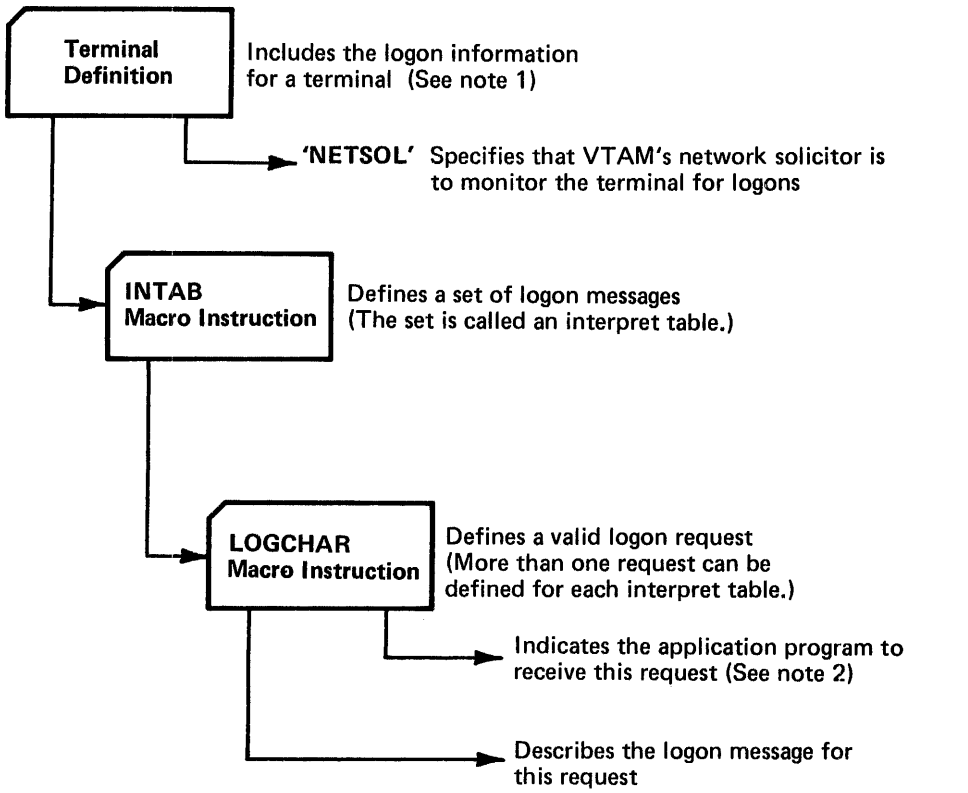
For item 3, the interpret table used to validate logon requests from this terminal is named in a terminal's GROUP, LINE, CLUSTER, VTERM, TERMINAL, or LOCAL definition statement. Figure 8-4 shows how control information for processing terminal-initiated logons is defined to VTAM for start-stop and BSC terminals.

Defining a Switched Network for Start-Stop and BSC Terminals

This section describes VTAM's support for call-in, call-out, and call-in/call-out terminals. (Switched-network support is provided only for start-stop and BSC terminals as specified in Appendix A.) Also included are discussions on network operator considerations that apply to controlling switched networks. This section is meant to augment the discussion on NCP support for start-stop and BSC switched networks provided in the *NCP Generation* publication.

Call-in Terminals

Call-in (dial-in) terminals have VTAM-definition requirements that affect some network operator and application program activities. Understanding these effects requires an understanding of VTAM's support of call-in terminals.



Notes:

1. The logon information can be specified in the GROUP, LINE, CLUSTER, VTERM, TERMINAL, or LOCAL statement.
2. This parameter can point to an actual application program or to an installation-coded exit-routine (the interpret-logon routine) that determines the application program to receive the request.

Figure 8-4. Providing Control Information for Processing Logon Requests from Local 3270, BSC, and Start-Stop Terminals

VTAM uses the concept of a port to support call-in terminals. As noted in the *NCP Generation* publication, each call-in line must have a TERMINAL statement with a CTERM=YES operand. These statements represent logical connections to the NCP, but they represent ports to VTAM.

For VTAM to accept a call-in request over a switched line, the port for that line (in addition to the other nodes in the path) must be active. Ports are activated automatically when an NCP is activated and loaded by VTAM. Thereafter, ports can be activated or deactivated using the network operator's VARY command.

The name of a port is the name of the TERMINAL statement with the CTERM=YES specification. This name is used by the network operator to address the port, but it is not used by an application program because application programs connect only to terminals.

Note that for a switched line, a port is a minor node: It is defined to VTAM and can be addressed. In addition, the port must be active (as must other path elements such as lines) for a terminal to be connected with an application program.

The TERMINAL statement that defines a port may also contain a definition of a terminal. Such a statement is used as a definition of a terminal only if the terminal calling in can not be identified and associated with another TERMINAL or VTERM statement. (This identification would be provided by either the NCP's MTA facility or by VTAM's and the NCP's BSC, and TWX, identification facilities.) That is, if a terminal calling in over a switched line cannot be identified, VTAM attempts to apply the terminal description in the port (TERMINAL) statement for that line to the terminal. Thus, information such as terminal type, automatic-logon specifications, and initial status is applied to the terminal calling in.

For the descritpion on the port statement to be applied to the terminal, a UTERM operand must be specified in that statement. The name in the UTERM specification is the name of the terminal (any unidentified terminal) calling in over the line serviced by the port. See 'Defining the NCP," earlier in this chapter, for a description of the UTERM specification on the TERMINAL statement.

An application program wishing to be connected to any unidentified terminal calling in over a specific line uses, as the name of the terminal, the UTERM name, not the name of the TERMINAL statement itself. Likewise, an automatic-logon specification in a port statement is applied to any unidentified terminal calling in over that line; the terminal is logged on to the program specified (either the network solicitor or an application program).

In summary, the following should be considered when planning to use VTAM's support for call-in terminals:

- For MTA lines, the VTERM statement can be used to provide identification and logon information for each type of terminal. If a VTERM statement is not used, an MTA terminal is defined by the port statement (if that statement has a UTERM specification).
- For BSC (and TWX) terminals, the IDLIST and the VIDLIST statements can be used to distribute identification responsibility between the NCP and VTAM.
- For port definition, the TERMINAL statement with the CTERM=YES parameter defines a port.
- For unidentified terminals, a UTERM name must be specified on a port statement if VTAM is to connect unidentified terminals calling in over the line serviced by the port.

Call-out Terminals

VTAM has no special requirements for supporting call-out terminals. Using VTAM-definition and NCP facilities, an installation can specify that a terminal is to be dialed automatically or manually by the network operator. The dial digits must be specified at NCP generation. For automatic dialing, the numbers are dialed by the NCP. For manual dialing, VTAM transmits a message (containing the dialing instructions) to the network operator.

Call-in/Call-out Terminals

Special planning is required for terminals that both call in and call out. Such terminals might be represented twice to the NCP and to VTAM.

If a terminal can be identified during a call-in operation through BSC (or TWX) identification facilities, it is defined for both call-in and call-out operations by the same TERMINAL statement.

If a call-in terminal is unidentified or is an MTA terminal, it is defined by either the UTERM name in a TERMINAL statement or by a VTERM statement (MTA terminals only). If the same terminal can be called, it must have another TERMINAL statement

defining its call-out characteristics. Thus, the terminal has two definitions and two names: one for calling in, the other for calling out.

Although the call-out definition applies to a specific terminal, the call-in definition can apply to any valid, but unidentified (including MTA) terminal calling in.

An activation or deactivation request received from the network operator and directed to one of the terminal definition does not affect the other definition. For example, a deactivation request specifying the call-out name does not affect the terminal's ability to call in. If such a request is issued, the terminal can still be used to call in even though a call-out operation would be prohibited by VTAM.

Similar considerations apply for application programs connecting with terminals capable of both being called and calling in. If an application program were to connect with a terminal (MTA or unidentified) that has called in, it connects using the call-in name. Then, if the application program disconnects the terminal and subsequently attempt to reconnect it by dialing out, the name of the terminal specified in call-out definition has to be used in the connection request.

Operating a VTAM System with Local 3270, BSC, and Start-Stop Terminals

The operation of a VTAM network described in Chapter 4 includes the facilities that apply to local 3270, BSC, and start-stop terminals. In addition, the network operator can start and stop the network solicitor.

The MODIFY command can be used to start the network solicitor. This network solicitor must be (1) the VTAM-defined network solicitor or (2) the network solicitor that was modified with the NETSOL macro instruction (with the name NETSOL).

Activating the network solicitor causes all appropriate available terminals to be automatically logged onto it. Appropriate available terminals are terminals that meet all the following requirements:

- They are active but not connected, nor queued for connection, to another program.
- The automatic-logon specification in their node definition indicates the network solicitor.

As long as the network solicitor remains active, it continues to accept appropriate available terminals. It attempts to solicit logon requests from these terminals and passes valid logon requests to the appropriate application programs.

Deactivating the network solicitor with the MODIFY command causes the network solicitor to complete handling all terminal-initiated logon requests in process and to disconnect all other terminals connected to it. No additional automatic logons are accepted by the network solicitor until it is reactivated.

Writing a VTAM Application Program

The concepts and facilities described in Chapter 5, "Writing a VTAM Application Program," except where noted, apply for programs that communicate with local 3270, BSC, and start-stop terminals. These concepts include:

- Connection
- Overlapping VTAM requests with other processing
- Application program exit routines. (However, the DFASY, SCIP, and RESP exit routines do not apply.)

- Error notification and handling
- Specific-mode and any-mode input requests
- Continue-any and continue-specific modes
- Handling overlength input data

Routines for Local 3270, BSC, and Start-Stop Terminals

As suggested in Sample Program 2, a program that communicates with both logical units and local 3270, BSC and start-stop terminals might be organized so that as much code as possible is shared. Separate communication routines might be written for individual types of local 3270, BSC, or start-stop terminals. In a program with logon acceptance, VTAM, as the result of an OPNDST macro instruction, identifies the type of terminal (logical unit or type of local 3270, BSC, or start-stop terminal) by setting a value in the DEVCHAR field of the NIB furnished by the application program. By testing the value in this field, the program can determine the appropriate macro instructions and related logic with which to communicate with the terminal.

Basic Mode Concepts

The macro instructions and facilities used by the application program to communicate with VTAM-supported start-stop and BSC terminals and local 3270 terminals are different from those used to communicate with logical units. The two sets of macro instructions and facilities are called the *basic mode* and the *record mode.*

The basic mode set of macro instructions must be used for the start-stop and BSC terminals, and can also be used for the local and BSC 3270. The record mode set is used for logical units and can be used for the local and BSC 3270.

The Basic Mode Macro Instructions

Here is a brief description of the basic mode communication macro instructions:

SOLICIT: Requests that VTAM solicit input from a specific local 3270, BSC, or start-stop terminal or from a group of terminals. Input is read into a VTAM buffer, not into the application program; a READ macro instruction is used to read the input into the application program's data area. The solicitation action caused by a SOLICIT macro instruction issued to a group of terminals continues until input has been received from every terminal in the group. Once input has been received from a terminal, the terminal must be resolicited unless, at connection, continuous solicitation of the terminal was specified.

READ: Requests that VTAM transfer data from a specific terminal or from any one of a group of terminals into an area in the application program. A request to read any one of a group requires a SOLICIT prior to the READ; a request to read a specific terminal solicits input if a SOLICIT was not previously issued for that terminal. If data is already in a VTAM buffer as the result of a previous solicit or read operation, data is transferred immediately.

WRITE: Requests that VTAM transfer data from an application program to a specific terminal. The application can also request that VTAM send certain control information to a terminal or write conversationally (write and then read from a terminal).

DO: Requests that VTAM perform the special I/O action associated with a specific LDO control block in the application program.

RESET: Requests that VTAM cancel all outstanding I/O requests to a terminal and if necessary, reset any error lock that may have been set for the terminal to prevent output.

The LDO Control Block

In addition to the control blocks described in Chapter 5, in "The Control Block Macro Instructions," VTAM application programs that communicate with certain start-stop and BSC terminals can define a logical device order (LDO) control block. This control block

defines a particular kind of I/O operations that is not ordinarily performed, such as writing a positive response with leading graphics to a System/3 or System/370 CPU. The operation is requested by issuing a DO macro instruction that specifies the LDO.

The CHANGE Macro Instruction

In addition to the macro instructions described in Chapter 5, in "Support Macro Instructions," another macro instruction, CHANGE, can be used only with terminals in basic mode. (It cannot, however, be used with the 3270 in either basic or record mode.) The CHANGE macro instruction requests that the information furnished to VTAM as part of a connection request be changed. The macro instruction assumes that the NIB, used to pass information when requesting connection, has previously been modified with a MODCB macro instruction. Using CHANGE quiesces I/O activity, disconnects the terminal, then reconnects it.

Communicating with Local 3270, BSC, and Start-Stop Terminals

Data Blocks

The unit of data exchanged in record mode operations is different from that exchanged in basic mode operations. In record mode, the units exchanged are *messages* (which include data) and responses. In basic mode, the unit of data is the *block.*

Blocks are delimited differently for different types of terminals. For start-stop terminals, a block ends with an EOB character; for BSC terminals, a block ends with an ETB or ETX character.

Although the application program can solicit more than a block from a terminal, a READ macro instruction can move only a block into the application program's input area (or less, if the input area is smaller than a block). An output operation (a WRITE macro instruction) always sends one block to the terminal.

Solicitation

When the application program solicits data from a terminal, VTAM initiates whatever actions (such as polling or line preparation) are required to obtain data from the terminal and put it into VTAM buffers.

READ requests issued in the specific mode cause solicitation if no previously solicited data is in VTAM's buffers, and then moves data into the application program's I/O storage area. In contrast, READ requests issued in the any-mode can only move solicited data from VTAM's buffers into the application program's input area. The user of READ requests in the any-mode must therefore explicitly request solicitation. Figure 8-5 illustrates both explicit and implicit soliciting of data.

Specific-mode and any-mode is also used when data is solicited. In the specific-mode, data is solicited only from a single terminal. In the any-mode, data is solicited from all connected terminals.

An application program might use these forms of solicitation in the following manner:

1. The application program initially solicits data from all of the terminals to which it has become connected.
2. The application program then issues a READ in the any-mode, which is completed when one of the terminals responds to the solicitation.
3. The application program communicates with the terminal using WRITE and READ macro instructions issued in the specific-mode. The READ macro instructions cause implicit solicitation.

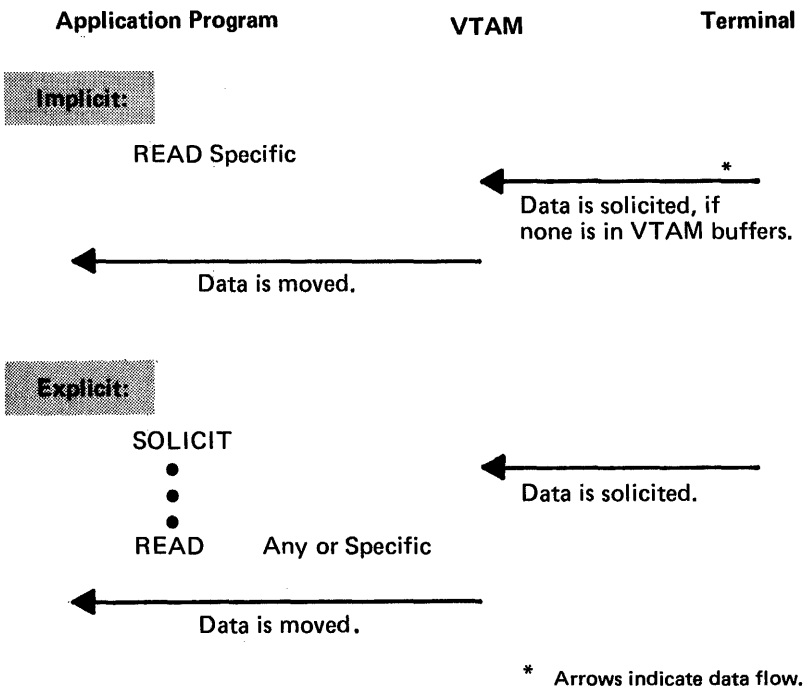


Figure 8-5. Implicit and Explicit Silicitation Using Basic Mode

4. When the transaction is completed, the application program issues a new SOLICIT macro instruction directed specifically at the terminal, so that a new READ issued in the any-mode will be satisfied when the next transaction begins.

When connection is established with a terminal in the basic mode, the application program indicates the amount of data that each solicit request (implicit or explicit) is to obtain data from that terminal. It is the application program's responsibility to determine when a new solicit request should be issued.

The application program can designate that for each solicit request, VTAM:

- Solicits only a *block* of data from the terminal. For start-stop terminals, a block ends with an EOB character; for BSC terminals, a block ends with an ETB or ETX character.
- Solicits a *message* from the terminal. Messages do not apply to start-stop terminals; for BSC terminals, a message ends with an ETX character. Messages consist of one or more blocks. Note that a message in basic mode is not the same as a message in record mode.
- Solicits a *transmission* from the terminal. For both start-stop and BSC terminals, a transmisssion ends with an EOT character. Transmissions are comprised of one or more messages (for start-stop terminals, one or more blocks).
- Solicits the terminal *continuously* until the application program cancels the solicitation.

Soliciting Blocks: When data is solicited a block at a time and an error occurs during transmission, the terminal needs to recover only a limited amount of data. However, since the application program must frequently reissue a solicit request (to acknowledge the previous block and obtain a new one), data throughput over the communications line is reduced. Block solicitation is appropriate when an unusually high number of line errors is

expected and when the length of retransmitted data must be kept to a minimum, even at the expense of slower response times and poorer line utilization. The installation must authorize the soliciation of blocks in the application's APPL definition statement.

Soliciting Messages and Transmissions: The lengths of messages and transmissions are not as closely dependent on the type of terminal as are block lengths. Messages and transmission lengths are usually established by the terminal's operator and the nature of the application. The lengths of messages and transmissions from a remote job entry station, for example, are determined by the number of cards in each job deck and the number of job decks available for transmission at one time.

Since messages and transmissions tend to be much longer than blocks, message and transmission solicitation requires recovery of more data when an I/O error is detected. However, with these forms of solicitation, data transmission can be more efficient, because the acknowledgements and resolicitations needed to obtain the blocks are performed by the communications controller, not the application program.

Message and transmission solicitation is appropriate for applications that require short response times but can tolerate lengthy transmissions when required.

The choice between message solicitation and transmission solicitation (which can be made only for BSC terminals) depends on the acceptability of delays between messages. With transmission solicitation, delays between messages are minimized although more data must be recovered if errors occur.

Continuous Solicitation: The advantages and disadvantages of continuous solicitation are the opposite of those of block solicitation. By soliciting continuously, the application program can obtain data with the minimum of programming. However, the application program must determine when solicitation should cease, and must explicitly tell VTAM when to do so. If the solicitation must be interrupted frequently, the efficiency is lost.

Continuous solicitation is appropriate for batch input applications, where transmissions are relatively frequent and delays between blocks, messages, and transmissions must be minimized.

Special I/O Operations

The application program can initiate the following I/O operations with one request:

- Copy a remotely attached 3277 Display Station's buffer into the buffer of any printer or display station attached to the same cluster control unit (COPYLBM or COPYLBT operation)
- Read the entire contents of any 3270 display station buffer (READBUF operation)
- Send a positive response with leading graphic characters to a System/3 or System/370 CPU and then read the terminal's next block of data (WRTPRLG and READ operations); or send a negative response with leading graphic characters to one of these terminals and then reread the block of data (WRTNRLG and READ operations)
- Write data beginning with a block of heading characters to a System/3 or System/370 CPU (WRTHDR and WRITE operations)
- Write data to a terminal from separate output data areas (gather-write) or read from a terminal into separate input data areas (scatter-read)

To use these facilities, the application program builds a set of *logical device orders* (LDOs). Each LDO indicates the specific type of I/O operation (such as COPYLBM or READBUF), the data area to be used, and an optional indicator that links the LDO to a following one. In both form and manner of use, LDOs resemble channel command word (CCW) programs. A set of LDOs is executed with a DO macro instruction.

By using LDOs, the application program can request I/O operations that are not available with the conventional macro instructions like READ and WRITE.

Special Processing Options

When connection is established with a terminal, the application program can designate rules that VTAM is to follow during subsequent communication with that terminal. The extent of solicitation described above–block, message, transmission, or continuous–is one example. Other options, most of which relate to NCP processing, can be selected by the application program (some options are not available for all types of terminals):

- VTAM can treat the receipt of leading graphic characters as either a normal condition or as an error condition. These options are called the LGIN and LGOUT options. (The names of these options, like those that follow are the names coded as part of the PROC operand of the terminal's NIB.)
- The application program can allow the communications controller to insert idle device-control characters into output data, or it can prevent the insertion of these characters (TMFLL option).

  If the communications controller is prepared to receive intermediate transmission blocks (ITBs) from a terminal, the application program can allow the communications controller to insert an error information byte (EIB) into each block, or it can prevent the insertion of EIBs (EIB option). The application program can use the EIBs to perform error recovery (retries) on a subblock basis, rather than on a block basis.
- The application program can override any text time-out limitation that the communications controller might otherwise use with the terminal (TIMEOUT option).
- The application program can prevent the communications controller from employing error recovery procedures if an error is detected during output to the terminal, during input from the terminal, or during either input or output (ERPIN and ERPOUT options).
- For some start-stop terminals, the application program can determine whether the communications controller is to monitor the terminal for attention interruptions and whether or not to notify the application program when the attention interruption is detected (MONITOR option). VTAM notifies the application program by scheduling its ATTN exit-routine. (These are attention interruptions detected when the application program is not communicating with the terminal; attention interruptions that occur *during* an I/O operation are always brought to the attention of the application program by an RPL return code.)
- The application program can insert its own line-control characters into output data, or it can allow VTAM to do so (ELC option).
- The application program can send all data to the terminal in transparent text mode (BINARY option).

All of these options can be specified for each terminal. Unless the application program issues a request to change the rules, they remain in effect as long as the terminal is connected.

Communicating with a BSC or Local 3270 Terminal in Record Mode

The BSC or local 3270 is not defined as a logical unit; however, the application program can communicate with it, in *record mode* as if it were an SNA 3270.

The restrictions listed in Chapter 5 that apply to the SNA 3270 also apply to the BSC or local 3270 in record mode. Because of the more limited capabilities of the BSC or local 3270, the following additional restrictions also apply:

- The bracket convention must be used. If the application has no use for brackets, the first I/O operation of a connection and disconnection can be considered to be one bracket. Both the application program and the 3270 can begin a bracket. The first

input from a 3270 that begins an NCP session is marked as the beginning of the bracket. All subsequent messages received from the 3270 during the NCP session indicate that the bracket is being continued. The 3270 cannot end a bracket; this can only be done by the application program.

- SNA unformatted system services are unavailable to the BSC or local 3270 (this includes the ability to specify session parameters). The network solicitor must be used. (See "The Network Solicitor" earlier in this chapter for more information.)

Note: *The application program has the option of communicating with the BSC or local 3270 in the same manner used to communicate with other BSC and start-stop terminals. See "Basic Mode Macro Instructions" earlier in this chapter.*

APPENDIX A. SUPPORTED TERMINALS

This appendix lists the terminals[1] and terminal features supported by VTAM. Supported terminals are grouped as:

- SNA terminals, in turn grouped as:
  - Local SNA terminals
  - Remote SNA terminals
- Local 3270, BSC, and start-stop terminals, in turn grouped as:
  - Local 3270 terminals
  - BSC terminals
  - Start-stop terminals

Note: *For terminals on BSC and start-stop lines, VTAM receives and transmits data only in extended binary coded-decimal interchange code (EBCDIC). The NCP translates EBCDIC messages from VTAM to the appropriate transmission codes for remotely attached BSC and start-stop terminals. The NCP also translates all messages in transmission codes other than EBCDIC to EBCDIC before sending them to VTAM. For terminals on SDLC lines, VTAM receives and transmits data in EBCDIC or any other code. The NCP does no translation. The VTAM application program must do any translating from another code to EBCDIC and from EBCDIC to another code.*

SNA Terminals

SNA terminals consist of local and remote terminals. Local terminals are attached directly to the CPU on a channel. Remote terminals are attached on SDLC lines to either a local or a remote communications controller. The communications controller must contain a network control program.

An *SNA terminal product* is an IBM terminal product for which a PU (physical unit) statement and at least one LU (logical unit) statement would be required when defining the network to VTAM.

Note: *Features of an SNA terminal product and of the devices that attach to it are a function of the particular terminal product, not of VTAM.*

Local SNA Terminals

VTAM supports the 3790 Communication System attached on a channel. The following devices are supported:

3791 Controller. The 3791 is required for attaching any of the devices listed below as part of the 3790 system.

3277 Display Station (Models 1 and 2).

3792 Auxiliary Control Unit (Model 1). Support is included for the optional attachment of a 3793 Keyboard-Printer and a 2741 Communication Terminal.

3793 Keyboard-Printer.

Remote SNA Terminals

VTAM supports the following SNA terminals attached on an SDLC line to a local or remote communications controller.

[1] Terminals which are equivalent to those explicitly supported may also function satisfactorily. The customer is responsible for establishing equivalency. IBM assumes no responsibility for the impact that any changes to the IBM-supplied products or programs may have on such terminals.

Note: *SNA physical units (cluster controllers) cannot be operated in the same network containing start-stop or BSC terminals if that network is controlled by a 3704. Mixed operation of this type requires a 3705.*

3270

3270 Information Display System on nonswitched lines. The following devices in the 3270 system are supported in the VTAM network:

3271 Control Unit (Models 11 and 12). The 3271 is required for attaching the 3277, 3284 (Models 1 and 2), 3286, and 3288 as part of a 3270 system. Support is included for the following 3271 optional features:

1200–ASCII Code
1550–Copy
9761–EBCDIC Code

3277 Display Station (Models 1 and 2). Support is included for the following 3277 optional features:

6350–Selector Light Pen
9089–EBCDIC Character Set

3284 Printer (Models 1 and 2). Support is included for the following 3284 optional feature:

9089–EBCDIC Character Set

3286 Printer (Models 1 and 2). Support is included for the following 3286 optional feature:

9089–EBCDIC Character Set

3288 Printer (Model 2) supported as a 3286 Model 2 attached to a 3271 Model 12. Support is included for the following 3288 optional feature:

9089–EBCDIC Character Set

3275 Display Station (Models 11 and 12). The 3275 is required for attaching the 3284 Model 3. Support is included for the following 3275 optional features:

1200–ASCII Code
6350–Selector Light Pen
9089–EBCDIC Character Set
9761–EBCDIC Code

3284 Printer (Model 3). The 3284 Model 3 requires a 3275. Support is included for the following 3284 optional feature:

9089–EBCDIC Character Set

3600

3600 Finance Communication System on nonswitched lines. The following devices in the 3600 system are supported in the VTAM network:

3601 Finance Communication Controller (Model 1). The 3601 is required for attaching the 3604, 3610, 3611, 3612, and 3618 as part of the 3600 system.

3604 Keyboard Display (Models 1 and 2).

3610 Document Printer (Models 1, 2, and 3).

3611 Passbook Printer.

3612 Passbook and Document Printer (Models 1, 2, and 3).

3618 Administrative Line Printer (Model 1).

3614 Consumer Transaction Facility (Models 1 and 2). VTAM supports attachment of the 3614 to a 3704/3705 or a 3601. When attached to a 3601, a user-written 3601 application program is required.

Note: *The 3601 or 3614 cannot be operated with other devices in network control mode on a 3704. Mixed operation of this type requires a 3705.*

3650

3650 Retail Store System (U.S. and Canada only) on switched or nonswitched lines. The following devices in the 3650 system are supported in the VTAM network:

3651 Store Controller (Model 50). The 3651 is required for attaching the 3653, 3657, 3275, and 3659 as part of the 3650 system.

3653 Point of Sale Terminal.

3657 Ticket Unit.

3275 Display Station (Model 3). Support is included for the optional attachment of a 3284 Printer (Model 3).

3659 Remote Communications Unit (Model 1). The 3659 requires a 2400 bps nonswitched line.

3660

3660 Supermarket System on switched lines. The following devices in the 3660 system are supported in the VTAM network:

3651 Store Controller (Model 60). The 3651 is required for attaching the 3663 and 3669 as part of the 3660 system.

3663 Supermarket Terminal (Models 1 and 2). VTAM supports the optional attachment of the 3666 Checkout Scanner to a 3663.

3669 Store Communication Unit (Model 1).

3767

3767 Communication Terminal (Models 1 and 2) on switched or nonswitched lines. Support is included for the following 3767 optional feature:

SDLC adapter, unless one of the start-stop features is specified.

1201—ASCII Code

3770

3770 Data Communication System on switched or nonswitched lines. The following devices in the 3770 system are supported by VTAM:

3771 Communication Terminal (Models 1 and 2).

3773 Communication Terminal (Models 1 and 2).

3774 Communication Terminal (Model 1).

3775 Communication Terminal (Model 1).

3776 Communication Terminal (Model 1).

VTAM requires the following features for all the devices in the 3770 system:

1460—SDLC/BSC, Switch Control, or

1470—SDLC

Support is included for the following 3770 optional feature:

1201—ASCII Code

System/32

System/32 Batch Work Station on switched or nonswitched lines.

3790

3790 Communication System on switched or nonswitched lines. The following devices in the 3790 system are supported in the VTAM network:

3791 Controller (Models 1A, 1B, 2A, and 2B). The 3791 is required for attaching the 3793, 3277, and 3792 as part of the 3790 system.

3793 Keyboard-Printer

3277 Display Station (Models 1 and 2).

3792 Auxiliary Control Unit (Model 1). Support is included for the optional attachment of a 3793 Keyboard-Printer and a 2741 Communication Terminal.

Local 3270, BSC, and Start-Stop Terminals

Start-stop and BSC terminals can be attached to either a local or a remote communications controller in network control mode. Local 3270 terminals are attached by a channel to the CPU. Start-stop and BSC terminals (except for the 3270) are supported using basic mode macro instructions. The BSC 3270 and local 3270 terminals are supported using either the basic mode or record mode VTAM application program macro instructions.

Start-Stop Terminals

Start-stop terminals can be attached to either a local or a remote communications controller in network control mode. Application programs can communicate with start-stop terminals only through the basic mode of VTAM. The devices listed below are supported:

1050

1050 Data Communication System on either switched or nonswitched lines. The following devices in the 1050 system are supported by VTAM:

1051 Control Unit (Models 1 and 2). The 1051 is required for attaching any of the devices listed below as part of the 1050 system. Support is included for the following 1051 optional features:

1313–Automatic EOB
2953–Open Line Detection
4795–Line Correction
4796–Line Correction Release
5465–Open Line Detection
6100–Receive Interrupt
9698–Text Time-Out Suppression
9700–Transmit Interrupt

1052 Printer-Keyboard (Models 1 and 2). Support is included for the following 1052 optional features:

1313–Automatic EOB
9567, 9597–PTTC/BCD Code
9571, 9591–PTTC/EBCD Code

1053 Printer (Model 1). Support is included for the following 1053 optional features:

9567, 9597–PTTC/BCD Code
9571, 9591–PTTC/EBCD Code

1054 Paper Tape Reader (Model 1).

1055 Paper Tape Punch (Model 1).

1056 Card Reader (Models 1 and 3).

1057 Card Punch (Model 1).

1058 Printing Card Punch (Models 1 and 2).

1092 Programmed Keyboard.

1093 Programmed Keyboard.

2740 Model 1

2740 Communication Terminal (Model 1). Support is included for the following optional features for a 2740 on a switched line:

3255–Dial-up
8028–Transmit Control

Support is included for the following optional feature for a 2740 on a nonswitched line:

7479–Station Control

Support is included for the following optional features for a 2740 on either a switched or a nonswitched line:

6114–Record Checking
9567, 9597–PTTC/BCD Code
9571, 9591–PTTC/EBCD Code
Correspondence Code

Note: *VTAM does not support the 2760 Attachment feature (8301).*

2740 Model 2

2740 Communication Terminal (Model 2) on nonswitched lines. Support is included for the following 2740 optional features:

1495, 1496–Buffer Expansion
1499–Buffer Receive
6114–Record Checking
9571, 9591–PTTC/EBCD Code

2741

2741 Communication Terminal (Model 1) on either switched or nonswitched lines. Support is included for the following 2741 optional features:

3255–Dial-up
4708–Receive Interrupt
7900–Transmit Interrupt
9567, 9597–PTTC/BCD Code
9571, 9591–PTTC/EBCD Code
Correspondence Code

3767

3767 Communication Terminal (Models 1 and 2) supported as a 2740 Model 1, 2740 Model 2, or 2741.

As a 2740 Model 1, on switched or nonswitched lines, VTAM requires:

7111–2740 Model 1 Start-Stop

An optional feature is:

9560–Station Control

As a 2740 Model 2, on nonswitched lines, VTAM requires:

7112–2740 Model 2 Start-Stop

As a 2741, on switched or nonswitched lines, VTAM requires:

7113–2741 Start-Stop

Magnetic Card Typewriter

Communicating Magnetic Card SELECTRIC® Typewriter on switched lines. This device is supported as a 2741 terminal on a switched line with correspondence code. Thus, it is supported by VTAM as if it were a 2741 Communications Terminal equipped with a standard correspondence keyboard and element, the Dial-up feature, the Receive Interrupt feature, and the Transmit Interrupt feature.

System/7

System/7 Processor Station on switched or nonswitched lines. System/7 is supported as a 2740 Model 1 with Checking. VTAM requires:

1610–Asynchronous Communication Control

World Trade Telegraph

World Trade Telegraph Terminals on nonswitched lines. Support is included for the following World Trade Telegraph optional features:

Paper Tape Reader
World Trade TTY ZSC3 Figures Protected Code
World Trade TTY ITA2 International Telegraph Alphabet 2

AT&T 83B3

AT&T 83B3 Selective Calling Stations on nonswitched lines. Support is included for the following AT&T 83B3 optional feature:

Paper Tape Reader

WU 115A

WU Plan 115A Outstations on nonswitched lines. Support is included for the following WU 115A optional feature:

Paper Tape Reader

CPT-TWX

CPT-TWX Terminal (Models 33 and 35) on switched lines. Support is included for the following CPT-TWX optional features:

Data Interchange Code (8 level)
Even Parity
Forced Parity

Binary Synchronous Communications (BSC) Terminals

BSC terminals can be attached to either a local or a remote communications controller in network control mode. Application programs can communicate with BSC terminals, except for 3270s, only through the basic mode of VTAM. Either the basic mode or the record mode of VTAM can be used to communicate with 3270s. The devices listed below are supported.

2770

2770 Data Communication System on either switched or nonswitched lines. The following devices in the 2770 system are supported by VTAM:

2772 Multipurpose Control Unit. The 2772 is required to support any of the other devices listed below as part of the 2770 system. The following 2772 feature is required by VTAM:

5010—Multipoint Data Link Control

Support is included for the following 2772 optional features:

1340—Automatic Answering
1490—Buffer Expansion (256 bytes)
1491—Buffer Expansion Additional (512 bytes)
1910—Conversational Mode
3250—Display Format Control
3650—EBCDIC Transparency
3860—144 Character Print Line (1491 and 5558 on 2203 required)
4610—Identification
4690—Keyboard Correction
5890—Horizontal Format Control
6555—Space Compression/Expansion
7705—Synchronous Clock
7950—Trans/Rec Monitor Print
9140—Extended Re-Entry
9402—Line Termination-2-Wire
9761—EBCDIC Code
9762—ASCII Code
9936—Immediate WACK

50 Magnetic Data Inscriber. VTAM does not provide any editing of input from the 50.

545 Output Punch (Models 3 and 4).

1017 Paper Tape Reader (Models 1 and 2).

1018 Paper Tape Punch (Model 1).

1053 Printer (Model 1).

1255 Magnetic Character Reader. VTAM does not support 1255 Stacker Select.

2203 Printer (Models A1 and A2).

2213 Printer (Model 1 or 2).

2265 Display Station (Model 2).

2502 Card Reader (Models A1 and A2).

5496 Data Recorder.

2780

2780 Data Transmission Terminal on either switched or nonswitched lines. Support is included for the following 2780 optional features:

1340—Automatic Answering

1350—Automatic Turnaround

3401—Dual Communication Interface

5010–Multiple Record Transmission

5020–Multipoint Line Control

5820–120 Character Print Line

5821–144 Character Print Line

6400–Selective Character Set (USASCII)

7850–Terminal Identification

8030–EBCDIC Transparency

9150–Extended Retry Transmission

9761–ASCII Code

9762–EBCDIC Code

Note: *VTAM does not support the six-bit Transcode feature (9760).*

2980

2980 General Banking Terminal System on nonswitched lines. The 2980 system is supported by VTAM in the U.S. only. The following devices in the 2980 system are supported by VTAM:

2972 Station Control Unit (Model 8–RPQ 858160 and Model 11–RPQ 858231). Support is included for the following 2972 optional features:

RPQ 835503–Buffer Expansion
RPQ 858165, 858182–96-Character Buffer

2980 Teller Station (Model 1–RPQ 835504 and Model 4–RPQ 858147)

2980 Administrative Station (Model 2–RPQ 835505)

2971 Remote Control Unit (Model 3–RPQ 858144)

Note: *VTAM does not support the Batched Message Input feature for 2980 stations.*

3270

3270 Information Display System on nonswitched lines. The following devices in the 3270 system are supported by VTAM:

3271 Control Unit (Models 1 and 2). A 3271 is required for attaching a 3277, 3284 (Models 1 and 2), 3286, or 3288. Support is included for the following 3271 optional features:

1550–Copy
9761–EBCDIC Code

3277 Display Station (Models 1 and 2). Support is included for the following optional features:

6350–Selector Light Pen
9089–EBCDIC Character Set

3284 Printer (Models 1 and 2). Support is included for the following 3284 optional feature:

9089–EBCDIC Character Set

3286 Printer (Models 1 and 2). Support is included for the following 3286 optional feature:

9089–EBCDIC Character Set

3288 Printer (Model 2) supported as a 3286 Model 2 attached to a 3271 Model 2. Support is included for the following 3288 optional feature:

9089—EBCDIC Character Set

3275 Display Station (Models 1 and 2). A 3275 is required for attaching a 3284 (Model 3). Support is included for the following 3275 optional features:

6350—Selector Light Pen
9089—EBCDIC Character Set
9761—EBCDIC Code

3284 Printer (Model 3). Support is included for the following 3284 optional feature:

9089—EBCDIC Character Set

3735

3735 Programmable Buffered Terminal on either switched or nonswitched lines. Support is included for the following 3735 optional features:

5010—Multipoint Data Link Control

9761—EBCDIC Code

9762—ASCII Code

The following devices are also supported for the 3735:

5496 Data Recorder (Model 1)

3286 Printer (Model 3)

3740

3740 Data Entry System on either switched or nonswitched lines. The 3740 is supported as a System/3. Thus, only 3740 functions equivalent to System/3 functions are supported by VTAM. The following devices in the 3740 system are supported by VTAM:

3741 Data Station (Model 2) and 3741 Programmable Work Station (Model 4) on switched or nonswitched lines. The 3741 is required for attaching the 0129, 3713, 3715, and 3717. (The 3717 may be attached only to a 3741 Model 2.) Support is included for the following 3741 optional features:

1680—Expanded Communications
1685—Expanded Communications/Multipoint Data Link Control
5450—Operator Identification Card Reader
7850—Terminal Identification

0129 Card Data Recorder (Model 2).

3713 Printer (Model 1).

3715 Printer (Models 1 and 2).

3717 Printer (Model 1).

3747 Data Converter (Model 1) on switched or nonswitched lines. Support is included for the following 3747 optional feature:

1660—Communications Adapter

3750

3750 Switching System on nonswitched lines. The 3750 system is supported by VTAM for World Trade only. The following device is supported:

3751 Control Unit. Support is included for the following 3751 required features:

1450—Binary Synchronous Communication Adapter, Type A, or
1451—Binary Synchronous Communication Adapter, Type B

3770

3770 Data Communication System supported as a 2770 on switched or nonswitched lines. The following devices in the 3770 system are supported by VTAM:

3771 Communication Terminal (Models 1 and 2) supported as a 2772.

3773 Communication Terminal (Models 1 and 2) supported as a 2772.

3774 Communication Terminal (Model 1) supported as a 2772.

3775 Communication Terminal (Model 1) supported as a 2772.

3776 Communication Terminal (Model 1) supported as a 2772 or 3780.

VTAM requires the following features for all the devices in the 3770 system:

1460—SDLC/BSC, Switch Control, or

1461—BSC Point-to-Point, or

1462—BSC Multipoint

Support is included for the following 3770 optional feature:

1201—ASCII Code

3780

3780 Data Communications Terminal on either switched or nonswitched lines. If the 3780 does not include the card-punch component, the 3780 is supported as a 2770 without the Component Select feature. If the 3780 does include the card-punch component, the 3780 is supported as a 2770 with the Component Select feature. Thus only 3780 functions equivalent to 2770 functions are supported by VTAM. The following optional features are supported for the 3780:

3601—EBCDIC Transparency
5010—Multipoint Data Link Control
5701—Print Positions, Additional
9761—EBCDIC Code
9762—ASCII Code

Note: *Space Compression/Expansion is not supported for the 3780.*

5275

5275 Direct Numerical Control Station (Model 1) supported as a 3275 Model 1 or 3275 Model 2 with EBCDIC Code and EBCDIC Character Set on nonswitched lines. The 5275 is supported by VTAM in the U. S. only.

System/3

System/3 Central Processing Unit on either switched or nonswitched lines.

1315—Autocall (U.S. only)
2074—Binary Synchronous Communications Adapter
7477—Station Selection
7850—EBCDIC Transparency
9060—EBCDIC Code
9061—ASCII Code

System/7

System/7 Processor Station on either switched or nonswitched lines. System/7 is supported as a System/3 with the Binary Synchronous Communications Adapter (2074). IPL of System/7 or a nonswitched point-to-point line is not supported.

System/32

System/32 Batch Work Station on either switched or nonswitched lines. System/32 is supported as a System/3 with the Binary Synchronous Communications Adapter (2074).

System/370

System/370 (Models 115-168 MP) as a remote station on switched or nonswitched lines. The System/370 CPU must be attached at the remote location to a 2701, 2703, local communications controller in emulation or network control mode, or an Integrated Communications Adapter (ICA). Support is actually provided for the communications control unit as indicated in Figure B-2 in Appendix B. The following lists the supported optional features for each control unit:

2701 Data Adapter Unit

1302–Autocall (U.S. only)
1303–Autocall (U.S. only)
1314–Autocall (U.S. only)
1355–Dual Code (U.S. only)
3455–Dual Code (World Trade only)
3463-3465–Dual Communication Interface
8029–Transparency
9060–EBCDIC Code
9061–ASCII Code
9062–6-Bit Transcode

2702 Transmission Control Unit on local channel

1290–Autocall (U.S. only)
1319–Autopoll (U.S. only)

2703 Transmission Control Unit on local channel

1340–Autocall (U.S. only)
1341–Autocall (U.S. only)
7715–EBCDIC Code
7716–ASCII Code
7717–6-Bit Transcode
8055–2741 Break
9100–Transparency for ASCII

2715 Transmission Control Unit (Model 1 for 2790) on local channel

3801–Expanded Capability
4850–Local 2740 Adapter

2715 Transmission Control Unit (Model 2 for 2790) on local channel

3801-Expanded Capability
9401–Point-to-Point Non-Switched
9403–Multipoint Non-Switched

3704/3705 Communications Controller on local channel

EBCDIC Code, ASCII Code, Autopoll (U.S. only), and EBCDIC Transparency do not have special feature codes in the 3704/3705
LP/VS
NCP/VS
PEP

8002–Two-Channel Switch

ICA for the System/370 Model 125

4640 Integrated Communications Adapter

ICA for the System/370 Model 135

4640–Integrated Communications Adapter (EBCDIC standard feature)
9673–9680–Transparency
9681–9688–ASCII Code
9689-9696–6-Bit Transcode

Figure A-1 summarizes the support provided for each device in a VTAM network.

Local 3270 Terminals

Application programs can use either the basic mode or the record mode of VTAM to communicate with locally attached 3270 Information Display Systems. The following devices in the 3270 system are supported:

3272 Control Unit (Models 1 and 2). The 3272 is required for attaching any of the devices listed below as part of a local 3270 system.

3277 Display Station (Models 1 and 2). Support is included for the following optional features:

6350–Selector Light Pen
9089–EBCDIC Character Set

3284 Printer (Models 1 and 2). Support is included for the following optional feature:

9089–EBCDIC Character Set

3286 Printer (Models 1 and 2). Support is included for the following optional feature:

9089–EBCDIC Character Set

3288 Printer (Model 2), supported as a 3286 Model 2 attached to a 3272 Model 2. Support is included for the following optional feature:

9089–EBCDIC Character Set

Support is not included for the ASCII character set features for locally attached 3270s.

Controlling Device	Attachable Device	Type of Network		
		Point-to-Point Switched	Point-to-Point Nonswitched	Multipoint Nonswitched
Local				
3272-1, -2	3277-1, -2, 3284-1, -2, 3286-1, -2, 3288-2			
SNA [1]				
3271-11, -12	3277-1, -2, 3284-1, -2, 3286-1, -2, 3288-2		X	X
3275-11, -12	3284-3		X	X
3601-1	3604-1, -2, 3610-1, -2, -3, 3611, 3612-1, -2, -3, 3618-1		X	X
3614-1, -2			X	X
3651-50	3275-3 (with optional 3284-3), 3653, 3657, 3659-1	X	X	X
3651-60	3663-1, -2, 3669-1	X		
3767-1, -2		X	X	X
3771-1, -2		X	X	X
3773-1, -2		X	X	X
3774-1		X	X	X
3775-1		X	X	X
3776-1		X	X	X
3791-1A, 1B, 2A, 2B	3277-1, -2, 3792-1 (with optional 3793 and 2741), 3793	X	X	X
System/32		X	X	X
Start-Stop				
1051-1, -2	1052-1, -2, 1053-1, 1054-1, 1055-1, 1056-1, -3, 1057-1, 1058-1, -2, 1092, 1093	X		X
2740-1		X	X	X
2740-2			X	X
2741		X	X	
3767-1, -2				
as 2740-1		X	X	X
as 2740-2			X	X
as 2741		X	X	
Communicating Magnetic Card SELECTRIC ® Typewriter as 2741		X		
System/7 as 2790-1		X	X	X
WT Telegraph			X	
AT & T 83B3				X
WU Plan 115A				X
CPT-TWX (-33, -35)		X		

[1] SNA terminal products contain a physical unit and one or more logical units.

Figure A-1 (Part 1 of 2). Summary of Terminals and Terminal Systems That VTAM Supports

Controlling Device	Attachable Device	Type of Network		
		Point-to-Point Switched	Point- to-Point Nonswitched	Multipoint Nonswitched
BSC		1		2
2772	50, 545-3, -4, 1017-1, -2, 1018-1, 1053-1, 1255, 2203-A1, 2213-1, -2, 2265-2, 2502-A1, -A2, 5496	X	X	X
2780		X	X	X
2972-8, -11	2980-1, -4, -2, 2971-3			X
3271-1, -2	3277-1, -2, 3284-1, -2, 3286-1, -2, 3299-2			X
3275-1, -2	3284-3		X	X
3735-1		X		X
3741-2, -4 as System/3	0129-2, 3713-1, 3715-1, -2, 3717-1 (attached to 3741-2 only)	X	X	X
3747-1 as System/3		X	X	X
3751			X	X
3771-1, -2 as 2772		X	X	X
3773-1, -2 as 2772		X	X	X
3774-1 as 2772		X	X	X
3775-1 as 2772		X	X	X
3776-1 as 2772 or 3780		X	X	X
3780 as 2770		X	X	X
5275-1 as 3275-1, -2			X	X
System/3		X	X	X
System/7 as System/3		X	X	X
System/370-(115-168MP) as remote station		X	X	

1. Except for the 3270 and the 2972, a BSC device can share a telephone number with any other BSC device in a point-to-point switched network.
2. Except for the System/7, a BSC device can share a multipoint nonswitched line with any other BSC device.

Figure A-1 (Part 2 of 2). Summary of Terminals and Terminal Systems That VTAM Supports

APPENDIX B. REMOTE STATION VERSUS REMOTE CONTROLLER

Because VTAM supports the communications controller both remotely attached and as a remote station, a distinction should be made between the connections of local and remote communications controllers. In a VTAM network, there are two ways in which two communications controllers can be connected to each other. One way is to have a communications controller remotely attached as a satellite of a locally attached controller. The remote connection is depicted in Figure B-1. The other way is to connect two independently, locally attached communication controllers. The connection of locally attached communications controllers is depicted in Figure B-2.

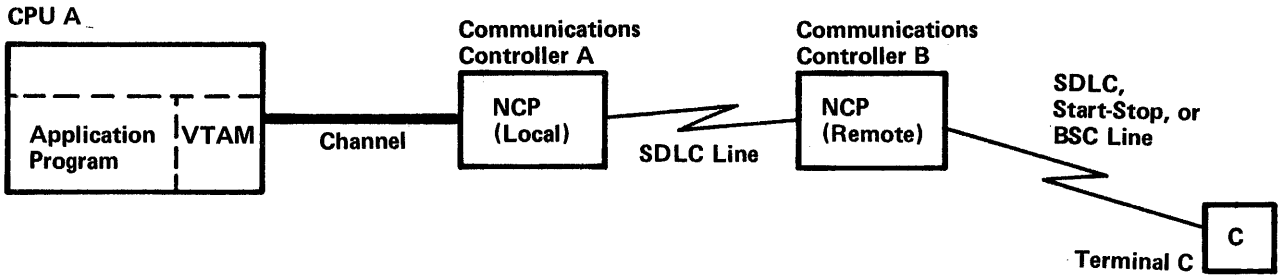


Figure B-1. A Remotely Attached Communications Controller

In Figure B-1, both VTAM in the host CPU and the NCP in the local communications controller control the remote communications controller. They recognize and use the remote unit as a communications controller, and they communicate with it, directing its attached devices. All control for the remote communications controller must emanate from the single host CPU and be passed through the local communications controller.

Figure B-2 also represents a connection between two communications controllers, but in this case, neither controller is viewed as a remote communications controller by the other. Each controller with its attached host CPU (containing independently functioning VTAMs) is viewed by the other as a single terminal. In fact, there are two telecommunications networks almost totally independent of each other. In this type of connection, the two communications controllers treat each other as remote stations, not as a remote communications controller with the processing capability of an NCP.

In the network configuration depicted in Figure B-1, VTAM in CPU A can directly address any terminal attached to communications controller B. Figure B-2 depicts a network in which the VTAM in CPU A *cannot* directly address a terminal attached to communications controller B.

Tracing a message through the two systems helps to clarify the differences between the two types of attachments. Suppose an application program in CPU A is to send a message to terminal C:

In the system depicted by Figure B-1:

To send the message, the application program must be executing in the CPU with VTAM. The application program must request connection, using VTAM facilities, to terminal C. After the connection is complete, the application program must request that VTAM transmit the message to the terminal.

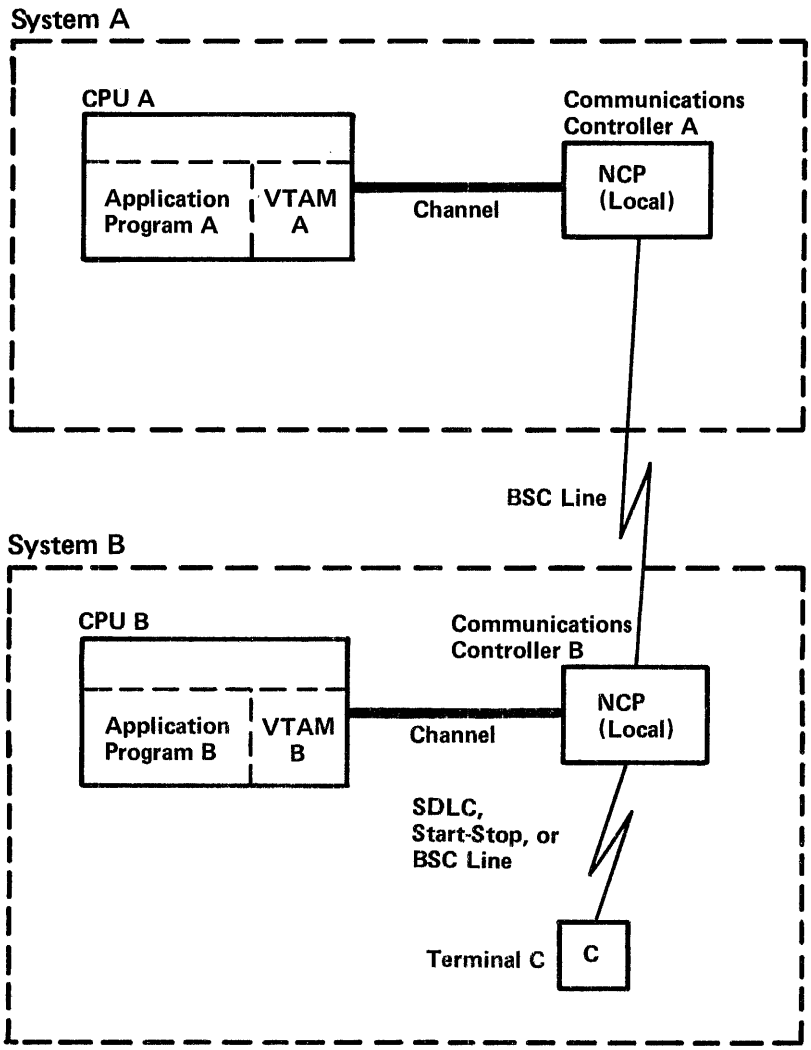
Upon receiving the request for data transmission, VTAM verifies that the terminal is in its network and transmits the message to the local communications controller. This controller (A in the figure), in turn, determines that the request is for a terminal

attached to the remote unit. Communications controller A then transmits the message to communications controller B. The remote controller routes the message to the terminal.

Note that the application program need not be aware of how the terminal is attached. For example, if the terminal is a 3270, it could be attached to CPU A, to communications controller A, or to communications controller B. Regardless of the attachment, the application program uses the same procedure to connect and communicate with the terminal.

In the system depicted by Figure B-2:

To send a message from an application program executing in the CPU in system A to a terminal (terminal C) in system B, the application program must be aware that terminal C is in another system. Also a companion application program must be executing in system B. The two application programs must be designed to work in coordination with each other.



Note: VTAM does not have to be the access method in both CPUs to use a controller as a remote station. For example, if another telecommunication access method were in CPU B, System B would still remain a remote station for VTAM in A

Figure B-2. Communications Controllers Attached as Part of Remote Stations

Application program A first requests VTAM for connection with the terminal that is system B. (Remember that each system is defined to the other as a single terminal.) The application program in CPU B also requests connection, but these requests are directed to VTAM B and are for terminal C and for the terminal that is system A.

To transmit the message, application program A requests that VTAM A transmit the message to the connected terminal (system B). VTAM A transmits the message to communications controller A which, in turn, writes the message to communications controller B. At this point, application program B must have issued a request to VTAM B for input from the terminal defined as system A. Upon receiving the message from controller A, communications controller B transmits it to VTAM B. VTAM B then sends the message to application program B. Using installation defined procedures, application program B determines that the message is destined for terminal C. So using VTAM B and communications controller B, application program B causes the message to be written to terminal C.

Thus when a communications controller is remotely attached (by a duplex line), its facilities and attached terminals are available directly to a single operating system. When two locally attached communications controllers are attached to each other, two operating systems are employed. Each operating system has direct access only to the facilities and attached terminals of only its communication controllers.

APPENDIX C. SUMMARY OF MESSAGE CONTROL INFORMATION

This appendix provides a summary of indicators and commands that can be exchanged between an application program using VTAM and a logical unit. The indicators and commands in this appendix are those that can be included as a message or part of a mesage.

A message can include data and/or indicators and commands. A response can include:

- The type of response, that is a definite response 1 or a definite response 2 and positive or negative.
- An explanation of the response if it is negative.
- Change-direction and bracket indicators. (These indicators can also be sent on messages and are described below.)

The remainder of this appendix is a table of the message commands and indicators. The table is divided into columns; each column is defined as follows:

Type of Indicator or Command: Specifies the name of the indicator or command. If it has an abbreviation, the abbreviation is in parentheses.

Function: Describes the use of the indicator or command.

VTAM Application Program can SEND/RECEIVE: Indicates whether the indicator or command can be transmitted or received by the application program.

Macro Used or RPL Field Set: Indicates the VTAM macro instruction or RPL field used in specifying the indicator or command.

Data-Flow Type: Indicates whether the indicator or command is sent in normal flow (DFSYN) or expedited flow (DFASY).

Next Action Expected: Indicates what action should occur following the exchange of indicator or command.

This table contains the following types of indicators and commands:

- Normal-flow
- Expedited-flow
- Change-direction
- Bracket
- SESSIONC

Type of Command or Indicator	Function	Application Program Can Send/Receive	Macro Used or RPL Field Set	Data-Flow Type	Next Action Expected
Normal-Flow Commands					
Bid	Bid to begin a bracket	Send only	SEND with CONTROL=BID	DFSYN	Receive response that says whether or not a bracket can be started
Cancel	Tell receiver to purge chain elements already received	Send	SEND with CONTROL=CANCEL	DFSYN	Response indicates cancellation
		Receive	CANCEL set in CONTROL field	DFSYN	Purge chain elements
Chase	Determine that the receiver has no more responses to send	Send	SEND with CONTROL=CHASE	DFSYN	When a response to the CHASE is received, all responses are accounted for
		Receive	CHASE set in CONTROL field	DFSYN	Send a positive or exception response
Logical Unit Status (LUS)	Inform receiver of an unexpected condition	Send	SEND with CONTROL=LUS	DFSYN	Varies with condition
		Receive	LUS set in CONTROL field	DFSYN	Varies with condition
Quiesce Complete (QC)	Tell the receiver that the sender is now quiesced and will not send until released	Send	SEND with CONTROL=QC	DFSYN	Await Release-Quiesce command
		Receive	QC set in CONTROL field	DFSYN	Send to quiesced receiver
Ready to Receive (RTR)	Tell the VTAM application program that a bracket has ended and a request to begin one can be sent	Receive only	RTR set in CONTROL field	DFSYN	Send a message that includes BRACKET=BB
Expedited-Flow Commands					
Quiesce-at-End-of-Chain (QEC)	Tell the receiver to quit sending now or, if chaining, at the end of this chain	Send	SEND with CONTROL=QEC	DFASY	Receiver should send a Quiesce-Complete command
		Receive	QEC set in the CONTROL field	DFASY	Finish sending, then send Quiesce-Complete command
Release Quiesce (RELQ)	Permit the receiver to send now	Send	SEND with CONTROL=RELQ	DFASY	Expect message from receiver
		Receive	RELQ set in CONTROL field	DFASY	Send a message, if desired

Type of Command or Indicator	Function	Application Program Can Send/Receive	Macro Used or RPL Field Set	Data-Flow Type	Next Action Expected
Expedited-Flow Commands (Cont.)					
Request-Shutdown (RSHUTD)	Request the VTAM application program to send a SHUTD or to disconnect the logical unit	Receive only	RSHUTD set in CONTROL field	DFASY	Send a SHUTD or disconnect
Shutdown-Complete (SHUTC)	Tel the receiver that preparation for shutdown is complete	Receive only	SHUTC set in CONTROL field	DFASY	Disconnect the logical unit
Shutdown (SHUTD)	Tell a logical unit to prepare to shut down	Send only	SEND with CONTROL=SHUTD	DFASY	The logical unit should send a SHUTC
Signal	Pass a 4-byte message with an agreed-upon meaning	Receive	SIGNAL set in in CONTROL field; values in SIGNAL and SIGDATA fields	DFASY	Installation-or terminal product-defined
		Send		DFASY	Installation-defined
Change-Direction-Indicators					
Change-Direction-Command (CMD)	Tell the receiver that it is now its turn to send	Send	SEND with CHNGDIR=CMD	DFSYN	Receive from the logical unit
		Receive	CMD in CHNGDIR field	DFSYN	Send to the logical unit
Change-Direction-Request (REQ)	Request the receiver to send a Chang-Direction-Command indicator	Send	SEND with CHNGDIR=REQ	DFASY or RESP	Check incoming messages or responses for CMD in CHNGDIR field
		Receive	REQ in CHNGDIR field	DFASY or RESP	Send CMD when able

The Change-Direction-Command indicator can be sent in a message that contains data, or with a bracket indicator. It can also be sent in a response which, can include a bracket indicator.

The Change-Direction-Request indicator can be sent in a message that contains a QEC, RELQ, or Signal command, or a bracket indicator. It can also be sent in a response which, can include a bracket indicator.

Type of Command or Indicator	Function	Application Program Can Send/Receive	Macro Used or RPL Field Set	Data-Flow Type	Next Action Expected

Bracket Indicators

The normal-flow commands, Bid and Ready-to-Receive, are used by the VTAM application program to determine whether it can send a Begin-Bracket indicator.

The bracket indicators can be sent in a message that contains data or a normal-flow command (except Bid or Ready-to-Receive). A Change-Direction indicator can also be sent in the same message or response.

Begin-Bracket (BB)	Begin a bracket	Send	SEND with BRACKET=BB	DFSYN	Receive response that indicates a bracket was begun
		Receive	BB set in BRACKET field	DFSYN	None
End-Bracket (EB)	End a bracket	Send	SEND with BRACKET=EB	DFSYN	Receive response that indicates the bracket was ended
		Receive	EB set in BRACKET field	DFSYN	Can mean "end-of-transaction"

SESSIONC Commands

These commands control and are sent separately from normal- and expediated-flow messages and their responses.

Bracket and change direction indicators cannot be sent with SESSIONC command messages or responses.

Clear	Stop all sending of normal-flow or expediated-flow messages and responses, clear pending ones from the network, and reset sequence numbers to 0	Send only	SESSIONC with CONTROL=CLEAR	Not Applicable	Reset sequence number to other than 0 (Optional) and restart message flow
Request-Recovery (RQR)	Tell the VTAM application program that a recovery procedure is required	Receive only	RQR set in CONTROL field in SCIP exit-routine	Not Applicable	Clear and reset sequence numbers to other than 0 (Optional) and restart message flow
Start-Data-Traffic (SDT)	Allow the sending of normal-flow or expediated-flow messages and responses	Send only	SESSIONC with CONTROL=SDT	Not Applicable	Normal message flow can resume
Set-and-Test Sequence-Number (STSN)	Exchange information with the logical unit so that sequence numbers can be determined and/or reset	Send only	SESSIONC with CONTROL=STSN and settings in IBSQAC and/or OBSQAC fields and IBSQVAL and/or OBSQVAL fields	Not Applicable	Receive response indicating logical unit reaction to STSN message

APPENDIX D. COMPATIBILITY CONSIDERATIONS FOR BSC AND LOCAL 3270 TERMINALS AND SNA 3270 TERMINALS

Since SNA unformatted system services are unavailable to BSC and locally attached 3270 terminals, problems may arise for installations that are currently using these terminals in record mode and that plan to replace them or use them together with SNA 3270 terminals.

The following guidelines may ease the transition from one type of terminal to the other. They should also permit both types to be used together without any apparent differences to the terminal operator or application program.

Changes Necessary when Replacing Terminals

NCP Definition: PU and LU macro definition statements for the SNA 3270 must replace the CLUSTER and TERMINAL macros for the BSC 3270.

Interpret Tables: SNA interpret tables may be required for SNA 3270 terminals. Only the application-program name field (from the APPLID in an SNA unformatted system service logon request) is interpreted for the SNA 3270. For BSC and local 3270 terminals, the entire logon character string is interpreted.

Logon Compatibility

If an installation permits a terminal operator to issue logons from a BSC or local 3270, the logon character string that is defined in its interpret tables or within an installation-defined network solicitor should conform to the syntax required by SNA unformatted system services. The logon sequence entered through a BSC 3270 will also work through an SNA 3270. For example:

LOGON APPLID(application)

LOGON application

LON

LOGON APPLID(application) LOGMODE(mode)

In addition, if the application program is to be unable to distinguish between a BSC or local 3270 and an SNA 3270, it should not examine logon data (using INQUIRE LOGONMSG or INQUIRE SESSPARM commands). This restriction is necessary since the logon data from a BSC or local 3270 is the entire logon character string while the logon data from an SNA 3270 is only the data portion of the logon message.

As an alternative to not examining the logon data, an installation can write its own network solicitor that will truncate the logon character string and CLSDST PASS only the data portion of the logon message to the application program. Using this modified network solicitor, the following types of logons could be entered from a BSC or local 3270 and handled by the application program as if they had been entered by an SNA 3270:

LOGON APPLID(application) DATA(user-data)

LOGON application, user-data

Using an Interpret Table for SNA Terminals

An installation can define an interpret table for SNA devices. This will convert logons entered through SNA devices into a form acceptable to non-SNA devices. This facility is not recommended because it precludes the use of the full range of SNA functions by the SNA terminal.

Application Program Considerations

Debugged application programs that have been written using record mode for BSC and local 3270 terminals can operate with SNA 3270 terminals, with the following exceptions:

- The program cannot examine logon data unless modified to handle the SNA format as discussed above.
- Since the SNA 3270 sends segmented request units, the LOSTERM exit may be scheduled with a return code of 28 when segmenting errors are detected. The application program should be modified to handle this condition in processing the LOSTERM exit.

Application programs that use both the BSC and local 3270 and the SNA 3270 must be according to the following rules:

- There should be logon compatibility as discussed above.
- Record mode must be used since SNA 3270 terminals cannot use basic mode.
- The restrictions listed in "Comunicating with a BSC or Local 3270 Information Display System in Record Mode" in Chapter 5 must be carefully adhered to. Since VTAM does not enforce these restrictions for the SNA 3270, failure to adhere to these restrictions will cause unpredictable results.
- SNA services (logon mode, variable session parameters, 3270 non-bracket operation, conditional logoff) can not be used; they are available to the SNA 3270 only.
- When an error is encountered on a SEND operation that has specified End Bracket, the bracket will be ended for local and SNA 3270 terminals, but will not be ended for BSC 3270 terminals. When retrying the SEND macro instruction, the Begin Bracket indicator can be specified. If a bracket error occurs because the bracket had not been ended (for the BSC 3270), the SEND can be reissued after the bracket indicator is changed to End Bracket.

GLOSSARY

Note: *Terms marked (SNA)–for example, SSCP-LU session. (SNA)–are defined in the Systems Network Architecture (SNA) publications. The definitions in this glossary explain what the terms mean in a VTAM context.*

A

ACB. *Access method control block.*

accept. To connect a terminal to an application program in response to a logon request from that terminal, a connection request from the network operator or from another application program, or as the result of an *automatic logon* request. In these cases, a terminal is accepted by issuing an OPNDST (OPTCD=ACCEPT) macro instruction.

access method control block (ACB). In VTAM, a control block that links a VTAM application program to VTAM.

accounting exit routine. An installation-coded routine invoked by VTAM to collect statistics for each connection and disconnection request.

acquire. To connect a terminal to a VTAM application program in the absence of a logon request from the terminal. The application program acquires a terminal by issuing an OPNDST (OPTCD=ACQUIRE) macro instruction or by issuing a SIMLOGON macro instruction followed by an OPNDST (OPTCD=ACQUIRE) macro instruction.

Activate Logical Unit command. (SNA) The command sent by VTAM to a logical unit that begins the *SSCP-LU session.*

Activate Physical Unit command. (SNA) The command sent by VTAM to a physical unit that begins the *SSCP-PU session.*

active. Pertaining to a major node that is known to VTAM and is available for use in the network or pertaining to a minor node that is connected to, or is available for connection to a VTAM application program. Contrast with *inactive.*

ACTLU command. (SNA) See *Activate Logical Unit command.*

ACTPU command. (SNA) See *Activate Physical Unit command.*

any-mode. (1) The form of READ or RECEIVE operation that obtains data from any single terminal. (2) The form of SOLICIT operation that solicits data from all eligible connected terminals. (3) The form of OPNDST operation that establishes connection with any single eligible terminal from which a logon request has been received.

application program. See *VTAM application program.*

application program identification. In VTAM, the symbolic name by which a VTAM application program is identified to VTAM and the rest of the telecommunications network. This name is pointed to by the ACB's APPLID field or, if the APPLID field is omitted, it is the job step name (OS/VS) or the name on the EXEC statement (DOS/VS).

application-program logon request. A logon request issued by a VTAM application program on behalf of a terminal. A VTAM application program must be authorized during VTAM definition to issue this type of logon.

application program major node. The major node whose constituent minor nodes are VTAM application programs. An application program major node is defined by one or more APPL statements.

APPLID routine. An installation-coded exit routine that assists in the translation of an argument for an interpret table. The routine may provide another level of authorization that has to be passed by a logon request.

asynchronous operation. A connection, communication, or other RPL-based operation, in which VTAM, after receiving the request for the operation from the program, returns control to the program so that it can continue its execution. When the operation is complete, VTAM interrupts the application program and either posts an ECB or schedules an RPL exit routine, as specified by the program. Contrast with *synchronous operation.*

asynchronous request. A request that causes control to be returned to a VTAM application program as soon as possible after the request has been accepted by VTAM. When the operation is completed, VTAM, as specified by the program either schedules an RPL exit-routine, or posts an ECB. A request is made asynchronous by setting the ASY option code in its RPL. Contrast with *synchronous request.*

authorization exit routine. An installation-coded routine that is invoked by VTAM for each connection and disconnection request to determine whether the request should be processed.

authorized path. In OS/VS2, a facility that enables an authorized VTAM application program to specify that a given SEND, RECEIVE, or RESETSR macro instruction be executed in a faster manner than usual.

automatic logon request. A request for connection to a specified VTAM application program that is generated by VTAM (rather than by the terminal itself) when the terminal becomes available for connection and the application program has opened its ACB and issued SETLOGON (OPTCD=START). Automatic logon requests are specified by the installation during VTAM definition and can be modified as the result of a network operator initiated logon request.

B

basic mode. The facilities (including the macro instructions needed to use them) that enable a VTAM application program to communicate with local 3270, start-stop, and BSC terminals. READ, WRITE, SOLICIT, RESET, DO, and LDO macro instructions are basic-mode macro instructions.

BB. See *Begin Bracket indicator.*

Begin Bracket indicator. A specification or indication in a VTAM application program RPL that a bracket be established.

Bid command. (SNA) A command used to determine whether a new bracket can be started. The receiver of the Bid command sends a positive response if a new bracket can be started or sends a negative response if a new bracket cannot be started. A Bid command is sent when a SEND macro instruction is issued with CONTROL=BID set in its RPL.

Bind command. (SNA) The command sent by VTAM to a logical unit that begins a *VTAM application program to LU session.*

block. In the basic-mode of VTAM, the smallest unit of data that may be transmitted between a VTAM application program and a terminal. The maximum size of a block is determined by the characteristics of the device that is sending or receiving the data. For start-stop devices, a block is a unit of data between an EOA or EOB character and an EOT or EOB character; for BSC devices, a block is a unit of data between an STX or SOH character and an ETB or ETX character. Contrast with *message* and *transmission.*

bracket. An exchange of one or more messages between a VTAM application program and a logical unit that accomplishes some task defined by the user as uninterruptable.

bracket protocol. A method of communication in which a new bracket is not started until the bracket in progress is completed.

BSC. Binary synchronous communication.

BSC cluster controller. A 3271, 3275, or 2972 control unit.

C

CA mode. See *continue-any mode.*

Cancel command. (SNA) A command that signifies to its receiver that the current chain being received should be discarded. A Cancel command is sent when a SEND macro instruction is issued with CONTROL=CANCEL set in its RPL.

change-direction protocol. (SNA) A method of communication in which the sender stops sending on its own initiative, signals this fact to the receiver, and prepares to receive.

Change Direction Request indicator. (SNA) An indicator that requests that a *Change Direction Command indicator* be returned.

Change Direction Command indicator. (SNA) An indicator that specifies that the sender has finished sending and is prepared to receive.

character-coded. (SNA) Pertaining to a logon or logoff command usually entered by a terminal operator from a keyboard and sent by a logical unit in character form. The character-coded command must be in the syntax defined in the installation's USS definition table. Contrast with *field-formatted.*

Chase command. (SNA) A command that signifies that all responses have been received. A Chase command is sent by issuing a SEND macro instruction with CONTROL=CHASE set in its RPL.

CID. Communications identifer.

Clear command. (SNA) A command sent from a VTAM application program to a logical unit indicating that no further messages and responses are to be exchanged and resetting sequence numbers to zero.

closedown. The process of deactivating VTAM and the telecommunication network. See also *quick closedown* and *orderly closedown.*

cluster controller. See *physical unit, SDLC cluster controller* and *BSC cluster controller.*

command. (SNA) In data communication, that part of a message that controls the exchange of other messages. For example, a Quiesce command tells the receiver to stop sending messages until notified to resume. When a message containing a command is sent by a VTAM application program, VTAM sets the RU Type indicator in the *request header* to indicate data flow control or session control (as appropriate) and puts a request code (for example, X'80' for a Quiesce at End of Chain command) in the *request unit.* In addition to commands, other control information can be sent in messages by specifying certain *indicators,* such as change-direction or bracket indicators. Contrast with *reply.*

communication control unit. A general term for a communication device that controls the transmission of data over lines in a telecommunication network. Communication control units include transmission control units and communications controllers.

communication line. Any physical link, such as a wire or a telephone circuit, that connects one or more remote terminals (or a remote computer) to a communication controller, or connects one communication controller to another.

communications controller. A type of communication control unit whose operations are controlled by a program stored and executed in the unit. Examples are the IBM 3704 and 3705 Communications Controllers.

communications identifier (CID). A VTAM-assigned network-oriented equivalent for a terminal's symbolic name. The installation assigns a symbolic name to each terminal in its network configuration. When a VTAM application program requests connection to the terminal—by placing the terminal's symbolic name into a NIB and issuing an OPNDST macro instruction—VTAM converts this 8-byte symbolic name into a 4-byte communications identifier. The application program uses this identifier for communication with the terminal.

configuration restart. In VTAM, the facility for reloading an NCP automatically following a failure in an NCP or in the communications controller containing that NCP and the facility for reestablishing contact with a physical unit after contact has been lost. Configuration restart includes the ability to restore the network to its status just prior to the failure.

connection. In VTAM, in response to a request (OPNDST) from a VTAM application program, the linking of VTAM control blocks in such a way that the program can communicate with a particular terminal. The connection process includes establishing and preparing the network path between the program and the terminal. Also see *queued for logon.*

continue-any (CA) mode. A state into which a terminal is placed that allows its input to satisfy an input request issued in the any-mode. While this state exists, input from the terminal can also satisfy input requests issued in the specific-mode. Continue-any mode is established by specifying OPTCD=CA for the RPL used by an RPL-based macro instruction. Contrast with *continue-specific mode.*

continue-specific (CS) mode. A state into which a terminal is placed that allows its input to satisfy only input requests issued in the specific-mode. Continue-specific mode is established by specifying OPTCD=CS for the RPL used by an RPL-based macro instruction. Contrast with *continue-any mode.*

conversational write operation. An operation wherein data is first sent to a local 3270, BSC, or start-stop terminal, and data is then read from that terminal. It is done as the result of a WRITE macro instruction having the CONV option code set in its RPL.

converted command. An intermediate form of a character-coded logon or logoff command produced by VTAM through use of a USS definition table. The format of a converted logon or logoff command is fixed; the USS definition table must be constructed in such a manner that the character-coded command (as entered by a logical unit) is converted into the predefined converted command format. If an unmodified IBM-supplied *USS definition table* is used or if no table is used, the character-coded command must be entered in the converted command form. By modifying

the USS definition table, the installation permits variations in the character-coded command formats. VTAM changes the converted command into an Initiate Self or Terminate Self command.

CS mode. See *continue-specific mode.*

D

DACTLU command. (SNA) See *Deactivate Logical Unit command.*

DACTPU command. (SNA) See *Deactivate Physical Unit command.*

data flow. The flow of messages containing data, messages containing commands to control further exchange of data messages, and responses to these messages that are exchanged between a VTAM application program and a logical unit.

data flow control. In relation to a VTAM application program-LU session, the control of data flow by messages and responses that are part of the data flow. A VTAM application program uses the SEND macro instruction and the CONTROL, CHNGDIR, and BRACKET parameters to send data flow control messages. Contrast with *session control*, which controls data flow from messages sent outside of the data flow.

data transfer. See *data transmission.*

data transmission. In data communications, the sending of data from a sender to a receiver.

Deactivate Logical Unit command. (SNA) The command sent by VTAM to a logical unit that terminates the *SSCP-LU session.*

Deactivate Physical Unit command. (SNA) The command sent by VTAM to a physical unit that terminates the *SSCP-PU session.*

definite response. (SNA) In sending a message, a request that a positive response or a negative response be returned to acknowledge how the message was received. In sending a response, any kind of response, whether positive or negative, and whether definite response 1, definite response 2, or both.

definite response 1. (SNA) In sending a message, a specification that a definite response be returned and that, whether positive or negative, it be identified as response type 1 (in addition, definite response 2 can be requested). Unless the message sender attaches some meaning to definte response 1 and definite response 2, specifying either or both of them mean no more than "Return a definite response." On receiving a response, if no meaning is attached to *response type 1* and *response type 2*, the receiver does not attempt to distinguish between them; it is sufficient to know that the response is positive or negative. *Definite response 1* and *response type 1* replace the term *FME response.*

definite response 2. (SNA) In sending a message, a specification that a definite response be returned and that, whether positive or negative, it be identified as response type 2 (in addition, definite response 1 can be requested). The significance of response type 2 as distinguished from response type 1 or response types 1 and 2 is determined by the user. *Definite response 2* and *response type 2* replace the term *RRN response.*

definition statement. The means of describing an element of the telecommunication system to VTAM.

device-control character. A control character that is embedded in a data stream to control mechanical and format operations at a terminal (for example, a line-feed character or carriage-return character). See also *line-control character.*

disconnection. In VTAM, the disassociation of VTAM control blocks in such a way as to end a session between a VTAM application program and a connected terminal. The disconnection process includes suspending the use of the network path between the program and the terminal. It is done as the result of a VTAM application program issuing a macro instruction.

E

EB. See *End Bracket indicator.*

EBCDIC. Extended binary-coded decimal interchange code.

emulation mode. The function of an NCP that enables it to direct a communications controller to perform the activities equivalent to those performed by the IBM 2701, 2702, or 2703 Transmission Control Unit.

End Bracket indicator. A specification or indication in a VTAM application program RPL that a bracket be terminated.

ERP. Error recovery procedure.

error lock. A condition established wherein communication with a local 3270, BSC, or start-stop terminal is suspended. The RESET macro instruction is used to reset the error lock.

exception message. In communicating with a logical unit, a message that indicates an unusual condition such as a sequence number being skipped. When VTAM detects such a condition, it notifies the VTAM application program VTAM and/or the VTAM application program provides sense information which is included in the response that is sent to the logical unit.

exception response. In sending a message, a specification that a response to the message is to be returned only if the message does not arrive successfully (that is, only a negative response is to be sent, if appropriate). In sending a response if a message does not arrive successfully, this term is not used; the response that is sent is always called a *negative response.* Contrast with *definite response.*

exit list (EXLST). In VTAM, a control block that contains the addresses of exit routines that receive control when specified events occur during VTAM execution. Exit routines handle events such as logon processing and I/O errors.

exit list routine. See *EXLST exit routine.*

EXLST. *Exit list.*

EXLST exit routine. A routine whose address has been placed in an exit list (EXLST) control block. The addresses are placed there with the EXLST macro instruction, and the routines are named according to their corresponding operand; hence DFASY exit routine, TPEND exit routine, RELREQ exit routine, and so forth. All exit routines are coded by the VTAM application programmer. See also *RPL exit routine.*

expedited-flow message. A message that is received independently from any messages that may be queued for the VTAM application program or logical unit and controls the normal flow of messages. Contrast with *normal-flow messages.*

F

field-formatted. (SNA) Pertaining to a logon or logoff command from an SNA terminal in the format of an Initiate Self or Terminate Self command. This format allows the logon or logoff request information to be in a specified format, such as binary codes, bit-significant flags, and symbolic names in defined fields. Contrast with *character-coded.*

formatted. Deprecated term for *field-formatted.*

FME response. See *definite response 1* and *response type 1.*

H

host CPU. The central processor for a VTAM telecommunication system. VTAM resides in the host CPU.

I

inactive. Pertaining to a major node that is unknown to VTAM and is unavailable for use in the network or pertaining to a minor node that is not connected to,nor available for connection to a VTAM application program. Contrast with *active.*

indicator. (1) A specification in a VTAM application program RPL pertaining to change-direction or bracket protocols. See *Change-Direction Request indicator, Change Direction Command indicator, Begin Bracket indicator,* and *End Bracket indicator.* (2) Any designated piece of information, such as a particular bit, that can indicate a condition.

Initiate Self command. (SNA) A command that is sent by a logical unit to VTAM (during the SSCP-LU session) requesting an LU session between the logical unit and a VTAM application program.

installation exit routine. Any of several types of special-purpose exit routine that an installation uses in common for all active VTAM application programs. The installation can code and and specify an accounting exit routine, an authorization exit routine, and a logon-interpret exit routine.

interpret table. In VTAM, an installation-defined correlation list that translates an argument into a string of eight characters. Interpret tables can be used to translate a logon message into the name of a VTAM application program for which the logon request is intended.

IPL. Initial program load.

L

LDO. Logical device order.

leading graphics. From one to seven graphic characters that may accompany an acknowledgment sent to or from a BSC terminal in response to receiving a block of data.

line. The communication medium linking a communication control unit to another communication control unit, or linking a communication control unit to one or more terminals.

line-control character. For start-stop or BSC terminals, a character in a data stream that controls the transmission of data over a network path. For example, line-control characters delimit messages and indicate whether a terminal has more data to send or is ready to receive data. See also *device-control character.*

line-control discipline. See *protocol.*

line group. One or more lines of the same type that can be activated and deactivated as an entity.

local. Pertaining to the attachment of devices directly by channels to a host CPU. Contrast with *remote.*

local 3270 major node. The major node whose minor nodes are locally attached 3270 terminals. A 3270 major node is defined by one LBUILD statement and one or more LOCAL statements.

local SNA major node. The major node whose minor nodes are locally attached physical and logical units of one or more 3790 Communication Systems. A local SNA major node is defined by one VBUILD statement (TYPE=LOCAL), one or more PU (local) statements, and one or more LU (local) statements for each PU statement.

logical connection terminal. For an NCP in network control mode, a description of a start-stop or BSC terminal (provided by a TERMINAL statement) to be used for a dial-in terminal whose identity is not known by the NCP when the terminal calls the communications controller.

logical device order (LDO). In VTAM, a set of parameters that specify a data-transfer or data-control operation to local 3270s and certain kinds of start/stop or BSC terminals.

logical error. An error that results from an invalid request.

logical unit. (SNA) The end point of an SNA terminal product with which a VTAM application program communicates.

log off. To request that a logical unit be disconnected from a VTAM application program. See *logoff request.*

logoff. See *logoff request.*

logoff request. A request from a terminal to be disconnected from a VTAM application program.

logoff command. A request by a logical unit to be disconnected from the VTAM application program to which it is connected—that is, to terminate the VTAM application program-LU session. Logoff commands occur either as field-formatted Terminate Self commands or as character-coded logoff commands. Either type of logoff command normally causes (1) the application program's LOSTERM exit-routine to be scheduled, (2) the application program to issue CLSDST, and (3) VTAM to send an Unbind command to terminate the VTAM application program to LU session.

log on. To request that a terminal be connected to a VTAM application program. See also *logon request.*

logon. See *logon request.*

logon command. A request by a logical unit to be connected to an application program—that is, to begin a VTAM application program-LU session. Logon commands occur either as field-formatted Initiate Self commands or as character-coded logon commands. Either type of logon command normally causes (1) the application program's LOGON exit-routine to be scheduled, (2) the application program to issue OPNDST, and (3) VTAM to send a Bind command to initiate the VTAM application program to LU session.

logon data. In VTAM, the data that can accompany a logon request received by the VTAM application program to which the request is directed. A VTAM application program, before connecting to the terminal with the OPNDST macro instruction, can use the INQUIRE macro instruction to obtain the logon data.

logon-interpret routine. See *APPLID routine.*

logon message. See *logon data.*

logon mode. A set of session parameters.

logon mode name. A one-to-eight byte representation of a logon mode (set of session parameters). A logical unit can use a logon mode name by including it in the LOGMODE portion of a logon command, and an application program can use a logon mode name by including it in the LOGMODE operand of an OPNDST or INQUIRE macro instruction. A LOGMODE may be established when the system is defined or by the network operator.

logon mode table. A table of macro-generated constants that associates a logon mode name with a set of session parameters. A logon mode table is created with MODETAB, MODEENT, and MODEEND macro instructions or an IBM-supplied logon mode table can be used.

logon request. A request by a terminal to be connected to an application program. For local 3270, BSC, and start-stop terminals (that is, for all non-SNA terminals), a logon request takes the form of an ordinary data message read by a network solicitor program. For SNA terminals, a logon request takes the form of a logon command (either an Initiate Self command or a character-coded logon command). Regardless of the type of terminal, the normal result of the logon request is (1) the scheduling of the application program's LOGON exit-routine and (2) the issuing of an OPNDST macro instruction by the application program to establish connection with the terminal.

LU-LU session. (SNA) See *VTAM application program-LU session.*

M

major node. A set of minor nodes that are capable of being activated, deactivated, and displayed as a group. Major nodes must consist entirely of one of the following five types of minor nodes: application programs, locally-attached 3270 terminals, locally-attached 3790 terminals, remotely-attached SNA terminals on switched links, or terminals defined to an NCP (SDLC terminals on leased links, BSC terminals, and start-stop terminals). The minor node definition statements that form the major node are filed together as a member of a VTAM definition data set; the name of the major node is the name used to file the definition statements and to activate and deactivate the major node.

message. (1) In VTAM, a single transmission of information between VTAM or a VTAM application program and an SNA terminal. A message contains data, information that controls the exchange of messages and responses, or both data and control information. A message is processed by VTAM as a request header and a request unit, which together make a basic information unit (BIU). To a VTAM application program, a message is sent by specifying parameters and a data area in a SEND macro instruction and received by issuing a RECEIVE macro instruction and examining fields of an RPL after the RECEIVE is executed. Contrast with *response.* (2) For BSC terminals, the unit of information between an STX character and an ETX character. For start-stop terminals, the unit of information that ends with an EOT character (thus, a message is the same as a transmission).

minor node. In VTAM, an element of the telecommunication network that is capable of being activated or deactivated by the VARY command. Included among minor nodes are: locally attached 3270 terminals; locally attached 3790 physical units and logical units; remotely attached physical units, logical units, BSC terminals and start-stop terminals; application programs; NCP lines and line groups. Each minor node is defined by a definition statement or macro instruction, and the name of each minor node is the label of the definition statement or macro instruction.

multiple terminal access (MTA). A feature of the network control program that permits it to communicate with a variety of dissimilar, commonly used start-stop terminals over the same switched network connection.

MTA. Multiple terminal access.

multithread program. A program that handles many processing requests for many terminals at the same time. Such a program usually uses one or more of these techniques for managing the sequence of its processing: ECB-posting, RPL exit routines and multitasking. Contrast with *single-thread program.*

N

NCP. Same as NCP/VS. See *network control program.*

NCP/VS. Network control program/virtual storage. See *network control program.*

NCP generation. See *network control program generation.*

NCP major node. The major node whose minor nodes are defined as part of NCP generation; specifically, lines and line groups, physical and logical units attached by leased SDLC links, BSC and start-stop terminals, and minor nodes defined by CLUSTER, INNODE, VTERM, and COMP macro instructions.

negative polling limit. For a start-stop or BSC terminal, the maximum number of consecutive negative responses to polling that the communications controller accepts before suspending polling operations.

negative response. A response indicating that a message did not arrive successfully. In some instances, VTAM determines that a message did not arrive successfully (for example, a message arrived with a sequence number which was not higher than the previous sequence number); in other instances, the application program makes this determination (for example, a message is not in a proper user-defined format). Contrast with *positive response.*

network control mode. The functions of an *NCP* that enable it to direct a communications controller to perform telecommunication activities such as polling, device addressing, dialing, and answering. Contrast with *emulation mode.*

network control program (NCP). A program, transmitted to and stored in a communications controller, that controls the operation of the communications controller.

network control program generation. The process, performed in a host CPU, of assembling and link-editing a macro instruction program to produce a network control program.

network definition. The process of defining, to VTAM, the identities and characteristics of each node in the telecommunication system and the arrangement of the nodes in that system.

network operator. The person responsible for controlling the telecommunication network.

network operator command. A command used by the network operator to monitor or control the telecommunication network.

network operator console. A system console from which a network operator controls a telecommunication network.

network operator logon. A logon requested in behalf of a terminal by means of a network operator command.

NIB. Node initialization block.

NIB list. A series of contiguous NIBs (node initialization blocks).

node. In VTAM, a major node, a minor node, or either.

node initialization block (NIB). A control block associated with each terminal at the time it is connected. The node initialization block contains information used by a VTAM application program to identify the terminal and to indicate to VTAM how communication with the terminal is to be done.

node name. In VTAM, the symbolic name assigned to a specific major or minor node during network definition.

non-SNA terminal. Devices supported by VTAM that utilize either start-stop or BSC line discipline to transmit data in a telecommunication system, or locally attached 3270 devices.

normal-flow message. A message that is received or queued in sequence between a VTAM application program and a logical unit. Contrast with *expedited-flow message.*

O

operator command. See *network operator command.*

option code. A value supplied by the OPTCD operand of the RPL macro instruction that indicates how a given connection or data transfer request is to be performed by VTAM.

orderly closedown. The orderly deactivation of VTAM and the telecommunication network. An orderly closedown does not take effect until all application programs have been disconnected from VTAM. Until then, all data transfer operations continue. Contrast with *quick closedown.*

P

partitioned emulation programming extension (PEP). A function of the NCP that enables a communications controller to operate some communication lines in network control mode while simultaneously operating others in emulation mode.

path. The intervening nodes and lines connecting a terminal with a VTAM application program in the host CPU.

PEP. Partitioned emulation programming extension.

physical unit. (SNA) (1) The control unit or cluster controller of an SNA terminal (for example, the 3601 Finance Communication Controller of a 3600 Finance Communication System, or the control unit of a 3767 Communication Terminal) (2) The part of the control unit or cluster controller that fulfills the role of a physical unit as defined by Systems Network Architecture (for example, the part of the cluster controller that controls SSCP-LU sessions or that handles recovery and resynchronization after a failure of a data link).

PIU. Path information unit.

positive response. A response that indicates a message was received successfully. Contrast with *negative response.*

processing option. One of the values supplied by the PROC operand of the NIB macro instruction that indicates how communication requests are to be performed for a terminal for the duration of its connection.

protocol. A general term for the set of rules, requirements, and procedures for transmitting information to and from a particular type of terminal in a telecommunications network.

Q

QC reply. See *Quiesce Completed reply.*

QEC command. See *Quiesce at End of Chain command.*

queued for connection. In VTAM, the state of a terminal that has logged on to an application program but has not yet been accepted for connection by that application program. Synonymous with *queued for logon.* Contrast with *connection.*

queued for logon. In VTAM, the state of a terminal that has logged on to an application program but has not yet been accepted for connection by that application program. Synonymous with *queued for connection.* Contrast with *connection.*

queued logon request. A logon request that has been directed to a VTAM application program but not yet accepted by that application program.

quick closedown. In VTAM, a closedown in which current data-transfer operations are completed, while new connection and data-transfer requests are canceled. Contrast with *orderly closedown.*

Quiesce at End of Chain (QEC) command. (SNA) A command sent by a VTAM application program or a logical unit indicating that the receiver should stop transmitting normal-flow messages after it has sent the last message of the chain being transmitted. A QEC is sent by issuing a SEND macro instruction with CONTROL=QEC set in the RPL.

Quiesce Completed (QC) reply. (SNA) A reply sent by a VTAM application program or a logical unit indicating that the sender will not transmit normal-flow messages again until it receives a Release Quiesce command. A QC reply is sent by issuing a SEND macro instruction with CONTROL=QC set in the RPL.

quiesce protocol. (SNA) A method of communicating in one direction at a time. Using this method, either the VTAM application program or the logical unit can assume the exclusive right to send normal-flow messages by getting the receiver to agree not to send normal-flow messages. When the VTAM application program or logical unit wants to receive, it can send the Release Quiesce command allowing the receiver to send.

quiescing. See *quiesce protocol.*

R

RDT. Resource definition table.

read operation. The transfer of data from VTAM buffers to VTAM application program storage.

read request. A request that causes VTAM to perform a read operation.

record mode. The facilities (and the macro instructions needed to use them) that enable the application program to communicate with logical units or with locally or remotely attached 3270 Information Display Systems terminals. SEND, RECEIVE, RESETSR, and SESSIONC are record mode macro instructions.

Release Quiesce (RELQ) command. A command sent by a VTAM application program or a logical unit indicating that the receiver can begin transmitting normal-flow messages. A RELQ command is sent by issuing a SEND macro instruction with CONTROL=RELQ set in the RPL.

RELQ command. See *Release Quiesce command.*

remote. Pertaining to terminals and communications control units that are attached to a central computer through a communications control unit. Contrast with *local.*

reply. In VTAM, a command that is sent in response to another command. For example, a Quiesce Complete reply is a reply to a Quiesce command.

request header. (SNA) A portion of a message that VTAM processes in an SNA-defined format. The request header is accompanied by a *request unit,* which contains data or control commands or replies. On receiving a request to send a message from a VTAM application program, VTAM determines from the SEND macro instruction parameters how to set request header bits. On receiving a message from a logical unit, VTAM converts information in the request header into information set in an RPL associated with a RECEIVE macro instruction in a VTAM application program.

request parameter list (RPL). A control block that contains the parameters necessary for processing a request for data transfer or a request for connecting or disconnecting a terminal or other operation.

request unit. (SNA) The unit of information (data or control information) that is sent or received by VTAM on behalf of a VTAM application program. Contrast with *response unit.*

resource definition table (RDT). In VTAM, a table that describes the characteristics of each node available to VTAM and associates each node with an address. The resource definition table is built during VTAM definition with APPL, LINE, GROUP, LU, and TERMINAL macro instructions, but it can be modified by the network operator while VTAM is running.

responded output. A type of output request that is completed when the logical unit receives the message and returns a response (if one is called for) for it. Responded output occurs if POST=RESP is specified for the RPL used by a SEND macro instruction. Contrast with *scheduled output.*

response. The unit of information that is exchanged between VTAM or a VTAM application program and an SNA terminal to describe how a message arrived. A VTAM application program sends a response with a SEND macro instruction that specifies STYPE=RESP; it receives a response by issuing a RECEIVE macro instruction that specifies RTYPE=RESP or by having VTAM schedule its RESP exit routine. VTAM processes a response as a response header and, if necessary, a response unit, which together make up a basic information unit.

response header. (SNA) A portion of a response that VTAM processes in an SNA-defined format. It may or may not be followed by a response unit. On receiving a request from a VTAM application program to send a response, VTAM determines from the SEND macro instruction parameters how to set response header bits. On receiving a response from a logical unit, VTAM uses information in the response header to set fields in an RPL associated with a RECEIVE macro instruction or in a VTAM RPL that is to be used by an application program's RESP exit routine.

response type 1. When receiving a response, whether positive or negative, an indication that the response is of type 1. Either or both response type 1 and response type 2 can be received in the same response. Whether the response type is examined by the receiver depends on whether meaning has been attached to response types by the user. For many users, it will be sufficient to know that the response is positive or negative. Replaces *FME response.*

response type 2. When receiving a response, whether positive or negative, an indication that the response is of type 2. Either or both response type 2 and response type 1 can be received in the same response. Whether the response type is examined by the receiver depends on whether meaning has been attached to response types by the user. For many users, it will be sufficient to know that the response is positive or negative. Replaces *RRN response.*

response unit. (SNA) The unit of information that is sent in response to a request unit specifically for the purpose of describing whether or not a request unit arrived successfully. Bracket and change-direction control information can also be included in a response unit.

RPL. *Request parameter list.*

RPL-based macro instruction. A macro instruction whose parameters are specified by the user in an RPL. For all RPL-based macro instructions, the user must specify the address of the RPL. All RPL-based macro instructions except CHECK permit RPL-modifying operands to be specified with the macro instruction.

RPL exit routine. A routine whose address has been placed in the EXIT field of an RPL. For asynchronous requests, this routine is automatically invoked by VTAM when the request associated with the RPL is completed. See also *EXLST exit routine.*

RRN response. See *definite response 2* and *response type 2.*

S

scheduled output. A type of output request that is completed (as far as the VTAM application program is concerned) when its output data area is free. Scheduled output occurs if POST=SCHED is specified for the RPL used by a SEND macro instruction. Contrast with *responded output.*

SDLC. Synchronous data link control.

SDLC cluster controller. A physical unit.

SDT command. See *Start Data Traffic command.*

sequence number. A number assigned to each message exchanged between a VTAM application program and a logical unit. The value increases by one throughout the life of the connection unless reset by the VTAM application program with a Set and Test Sequence Number of Clear command.

session. (SNA) A state during which two elements of the network can communicate with each other. See also *SSCP-PU session, SSCP-LU session,* and *VTAM application program-LU session.*

session control. Pertaining to a VTAM application program to LU session, the control of the data flow that is outside of the data flow. Session control includes starting and stopping the data flow and establishing data flow sequence numbers. A VTAM application program uses the SESSIONC macro instruction for session control. Contrast with *data flow control.*

session limit. In the communication controller's network control program, the maximum number of concurrent sessions that can be initiated on a non-SDLC multipoint line.

session parameter. (SNA) One of the set of characteristics, rules, or protocols that is specified when beginning a session (with the Bind command) between a VTAM application program and an SNA terminal (a VTAM application program-LU session). Session parameters determine such things as (1) what kinds of responses are allowable to messages that are sent, (2) whether a change-direction half-duplex protocol is to be used to take turns sending and receiving, and (3) whether one message must be responded to before another message can be sent. A logon request can specify a set of session parameters (a logon mode) that are to be adhered to, or, a VTAM application program can select the set of session parameters. The set of session parameters that is established is sent as part of the Bind command that results from an OPNDST macro instruction. Also see *logon mode name* and *session.*

SESSIONC command. A command that can be sent from a VTAM application program to a terminal without using SEND or RECEIVE macro instructions. SDT, Clear, and STSN are SESSIONC commands. All SESSIONC commands are sent with a SESSIONC macro instruction.

Set and Test Sequence Number (STSN) command. A command sent by a VTAM application program to a logical unit to establish the proper sequence number.

shared. (1) Pertaining to communication control units and communications lines that may be used concurrently by several teleprocessing programs to communicate with different nodes. (2) Pertaining to terminals that may be used by more than one

teleprocessing program; only one teleprocessing program may be connected to a shared terminal at any one time.

simulated logon request. A request initiated by a VTAM application program (via the SIMLOGON macro instruction) on behalf of a terminal, for connection between the terminal and a program. Contrast with *logon request* and *automatic logon request.*

single-thread program. A program that handles one processing request from any of its terminals at a time. Such a program usually requests VTAM to perform operations synchronously with relation to the program's execution, waiting until each operation is completed before proceeding further. Contrast with *multithread program.*

SNA. See *Systems Network Architecture.*

SNA terminal. In VTAM: (1) A SNA physical unit or a SNA logical unit. (2) A terminal designed to be compatible with Systems Network Architecture (SNA). For example, the 3790 Communication System is a product whose terminals (relative to VTAM) are compatible with SNA. (In addition to SNA terminals, VTAM also supports local 3270, and certain BSC and start-stop terminals.)

SNA terminal product. A control unit, attached to a channel or an SDLC link, that contains SNA terminals which are end points (relative to VTAM), and, depending on the type of control unit, any terminals that are local to the control unit. Examples of an SNA terminal product include: (1) a 3790 Communication System controller, the controller's SNA terminals (a physical unit and one or more logical units), and the terminals attached locally to the controller, such as 3793s and 2741s (2) a 3767 Communication Terminal (which contains a control unit as an integral part) and its SNA terminals (a physical unit and a logical unit).

solicit operation. The process of obtaining (or attempting to obtain) data from a local 3270, BSC, or start-stop terminal and moving that data into VTAM buffers.

solicit request. Any request that causes VTAM to perform a solicit operation. There are three ways to make such requests: (1) A SOLICIT macro instruction; (2) a READ macro instruction, if the SPEC option is in effect and if VTAM buffers hold no data from the device being read from; and (3) a WRITE macro instruction with the CONV option code set.

specific-mode. (1) The form of READ, RECEIVE, or SOLICIT operation that obtains data from one specific terminal. (2) The form of OPNDST operation that establishes connection with one specified terminal when a logon request is received from that terminal.

SSCP-LU session. (SNA) A state during which a portion of VTAM (in SNA terms, the "systems services control point") and the logical unit (LU) can communicate. The SSCP-LU session is initiated with the Activate Logical Unit command and is terminated with the Deactivate Logical Unit command. Among the commands that can be exchanged during the SSCP-LU session are the Initiate Self and Terminate Self commands, which request the initiation and termination of a VTAM application program to LU session.

SSCP-PU session. (SNA) A state during which VTAM (in SNA terms, the "systems services control point") and the physical unit (PU) can communicate. The SSCP-PU session is initiated with the Activate Physical Unit command and terminated with the Deactivate Physical Unit command. After initiating an SSCP-PU session, the Activate Logical Unit command and Deactivate Logical Unit command can be used to initiate and terminate an SSCP-LU session.

Start Data Traffic (SDT) command. (SNA) A command that allows data and data flow control messages and responses to be exchanged between a VTAM application program and a logical unit.

start options. The installation-specified (or defaulted) options that determine certain conditions that are to exist during the time that a VTAM system is operating. These options include: the size of VTAM storage pools, which major or minor nodes are to be traced by the VTAM trace facility, and which major nodes are to be made initially active. Start options can be predefined or specified by the network operator when VTAM is started.

STSN command. See *Set and Test Sequence Number command.*

subarea. A unique numerical value assigned by the installation to each major node (except VTAM application program major nodes) which is used by VTAM to create addresses for all of the minor nodes within that major node. A subarea value is assigned to each major node using the SUBAREA operand of the major node's LBUILD or VBUILD statement or the SUBAREA operand of the major node's BUILD macro instruction.

switched major node. The major node whose minor nodes are physical and logical units attached by switched SDLC links. A switched major node is defined by one VBUILD statement and one or more PU, PATH, and LU statements.

synchronous data link control (SDLC). A discipline for the management of information transfer over a communications channel. Transmission exchanges may be full-duplex or half-duplex; the communications channel configuration may be point-to-point, multipoint, or loop. SDLC includes comprehensive detection and recovery procedures, at the data link level, for transmission errors that may be introduced by the communication channel.

synchronous operation. A connection, communication, or other operation in which VTAM, after receiving the request for the operation, does not return control to the program until the operation is completed. Contrast with *asynchronous operation.*

synchronous request. A request that causes control to be returned to a VTAM application program only after the requested operation has been completed. A request is made synchronous by setting the SYN option code in its RPL. Contrast with *asynchronous request.*

Systems Network Architecture (SNA). The total description of the logical structure, formats, protocols, and operational sequences for transmitting information units through the communication system. Communication-system functions are separated into three concrete areas: the application layer, the function management layer, and the transmission subsystem layer. The structure of SNA allows the ultimate origins and destinations of information—that is, the end users—to be independent of, and unaffected by, the specific communication-system services and facilities used for information exchange.

T

TCU. Transmission control unit.

telecommunication network. In a telecommunication system, the combination of all terminals and other telecommunication devices and the lines that connect them.

telecommunication system. In a teleprocessing system, those devices and functions concerned with the transmission of data between the central processing system and the remotely located users. In VTAM, the telecommunication system includes the host CPU, VTAM application programs, VTAM, the telecommunication network, and the channels that link the host CPU and the network.

teleprocessing subsystem. In VTAM, a secondary or subordinate network (and set of programs) that is part of a larger teleprocessing system; for example, the combination consisting of a programmable controller, its stored program, and its attached input/output devices. An example of a teleprocessing subsystem is the IBM 3600 Finance Communication System.

teleprocessing system. The devices and functions of a data processing system that gives users at remote locations access to the data processing capabilities of a centrally located computer. A teleprocessing system has two major functions: the transmission of data between the central computer and the remote locations (performed by the telecommunication system) and the actual processing of the data in the central computer.

Terminate Self command. (SNA) A command that is sent by a logical unit to VTAM (during the *SSCP-LU session*) requesting the termination of the VTAM application program to LU session.

terminal. (1) To VTAM, an end point in a telecommunications network; that is a physical unit, a logical unit or a local 3270 start-stop, or BSC device. (2) relative to a VTAM application program, a logical unit or a local 3270, BSC, or start-stop device.

terminal component. A separately addressable part of a terminal that performs an input or output function.

terminal-initiated logon request. A logon request that originates from the terminal.

transmission. In data communication with start-stop and BSC terminals, a logical group of one or more blocks or messages. A transmission is terminated by an EOT character. See also *block* and *message.*

transmission control unit. A type of communication control unit whose operations are controlled solely by programmed instructions from the system to which the unit is attached; no program is stored or executed in the unit. Contrast with *communications controller.*

transparent text mode. A mode of binary synchronous transmission in which only line-control characters preceded by DLE are acted upon as line-control characters. All other bit patterns that happen to be line-control characters are transmitted as data.

U

Unbind command. (SNA) The command sent by VTAM to a logical unit that terminates a VTAM application program-LU session.

unformatted. Deprecated term for *character-coded.*

USS definition table. A table of macro-generated constants that enables VTAM to convert a character-coded logon or logoff command, first into a converted command and then into a field-formatted Initiate Self or Terminate Self command. The USSTAB, USSCMD, USSPARM, USSMSG, and USSEND macro instructions are used by the installation to generate a USS definition table.

USS. See *Unformatted System Services.*

Unformatted System Services. (SNA) A function of VTAM, defined by SNA, that converts character-coded logon or logoff commands into field-formatted Initiate Self or Terminate Self commands.

V

Virtual Telecommunications Access Method. An IBM program that controls communication between terminals and application programs running under DOS/VS, OS/VS1, and OS/VS2.

VTAM. Virtual Telecommunications Access Method.

VTAM application program. (1) To VTAM, the requests and control blocks that refer to a given ACB, or are pointed to by that ACB. (2) To the programmer, the set of instructions that perform a given function, including the processing of input and output data. One set of instructions may refer to more than one ACB.

VTAM application program to LU session. A state during which a VTAM application program and a particular logical unit can communicate. A VTAM application program-LU session is initiated with the Bind Command (sent by VTAM when the program issues an OPNDST macro instruction) and terminated with the Unbind Command (sent by VTAM when the program issues a CLSDST macro instruction).

VTAM definition. The process of defining the teleprocessing network to VTAM and modifying IBM-defined VTAM characteristics to suit the needs of the installation. VTAM definition is implemented by the installation with definition statements and macro instructions that are filed in a special data set or file.

VTAM definition library. The data set or file that contains the VTAM definition statements and VTAM start options filed during VTAM definition.

W

write operation. The transfer of data from VTAM application program storage to a terminal.

write request. A request that causes VTAM to perform a write operation.

Bibliography

This bibliography lists some of the non-VTAM IBM publications that are referred to in this publication or that relate to VTAM. (VTAM publications are shown in Figure P-1.) The listed publications (when available) contain a more detailed description of facilities that are used with VTAM.

General Teleprocessing

Introduction to Data Communications Systems, SR20-4461

Systems Network Architecture (SNA)

Systems Network Architecture General Information, GA27-3102

Systems Network Architecture Format and Protocol Reference Manual, GA27-3112

Operating Systems

DOS/VS

Introduction to DOS/VS, GC33-5370

DOS/VS System Generation, GC33-5377 (includes storage estimates)

DOS/VS System Management Guide, GC33-5371

DOS/VS System Control Statements, GC33-5376

DOS/VS System Utilities, GC33-5381

DOS/VS Serviceability Aids and Debugging Procedures, GC33-5380

DOS/VS and OS/VS TOLTEP for VTAM, GC28-0663

OS/VS

OS/VS Dynamic Support System, GC28-0640

DOS/VS and OS/VS TOLTEP for VTAM, GC28-0663

OS/VS1

OS/VS1 Planning and Use Guide, GC24-5090

OS/VS1 Storage Estimates, GC24-5094

OS/VS1 Services Aids, GC28-0665

OS/VS2

OS/VS2 System Programming Library:

Service Aids, GC28-0633

Supervisor, GC28-0628

Storage Estimates, GC28-0604

IBM 3704 and 3705 Communications Controllers

IBM 3704 and 3705 Communications Controllers: Network Control Program/VS Generation and Utilities; Guide and Reference Manual (for OS/VS and DOS/VS VTAM Users), GC30-3008

IBM 3704 and 3705 Communications Controllers: Network Control Program/VS; Program Logic Manual, SY30-3003

IBM 3704 and 3705 Communications Controllers: Network Control Storage and Performance Estimates (for OS and OS/VS TCAM and OS/VS and DOS/VS VTAM Users), GC30-3006

Introduction to the IBM 3704 and 3705 Communications Controllers, GA27-3501

Subsystem Support Services

Subsystem Support Services User's Guide, GC30-3022

Subsystem Support Services Logic, GC30-3017

OS/VS1 Subsystem Support Services Messages, GC38-1011

Terminals

IBM 3270 Information Display System

Introduction to Programming the IBM 3270, GC27-6999

IBM 3600 Finance Communication System

IBM 3600 Finance Communication System Installation Guide, GC27-0009

IBM 3600 Finance Communication System Programmer's Guide and Component Description, GC27-0004

IBM 3650 Retail Store System

IBM 3650 Retail Store System Subsystem Definition and Programmer's Guide, GC30-3023

SPPS Programmer's Guide, GC30-3024

IBM 3660 Supermarket System

IBM 3660 Supermarket System Subsystem Definition and Programmer's Guide, GC30-3025

IBM 3770 Data Communication System

IBM 3770 Communication System Programmer's Guide, GC30-3028

IBM 3790 Communication System

IBM 3790 Communication System Host System Programmer's Guide, GC27-0026

IBM 3790 Programming Statements Guide, GC27-0015

IBM Host Services Guide, GC27-0017

INDEX

Where more than one page reference is given, the major reference is first.

VTAM Concepts and Planning
Virtual Telecommunications
Access Method (VTAM)

GC27-6998-2

READER'S COMMENT FORM

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. This form may be used to communicate your views about this publication. They will be sent to the author's department for whatever review and action, if any, is deemed appropriate.

IBM shall have the nonexclusive right, in its discretion, to use and distribute all submitted information, in any form, for any and all purposes, without obligation of any kind to the submitter. Your interest is appreciated.

Note: *Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.*

How did you use this publication?

[ ] As an introduction
[ ] As a reference manual
[ ] For another purpose (explain) _____

[ ] As a text (student)
[ ] As a text (instructor)

_____

Is there anything you especially like or dislike about the organization, presentation, or writing in this manual? Helpful comments include general usefulness of the book; possible additions, deletions, and clarifications; specific errors and omissions.

Page Number: Comment:

What is your occupation? _____

Newsletter number of latest Technical Newsletter (if any) concerning this publication: _____

If you wish a reply, give your name and address:

IBM branch office serving you _____

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments.)

Cut or Fold Along Line

Reader's Comment Form

Cut or Fold Along Line

Fold Fold

First Class
Permit 40
Armonk
New York

Business Reply Mail
No postage stamp necessary if mailed in the U.S.A.

Postage will be paid by:

International Business Machines Corporation
Department 63T
Neighborhood Road
Kingston, New York 12401

Fold Fold

IBM

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
(U.S.A. only)

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)

VTAM Concepts and Planning (File No. S370-30) Printed in U.S.A. GC27-6998-2