GC20-1869-2

Systems

Systems Network Architecture – Introduction to Sessions between Logical Units

Logical Units (LUs) are the ports through which users communicate across an SNA network. The communication path through the network is represented by an LU to LU (LU-LU) session, and each user is represented by a logical unit.

This book discusses the use of logical units to support application-to-application and application-to-terminal communication. The types of LU-LU sessions are defined and the characteristics of each type are discussed.

The type number that is assigned to an LU-LU session identifies the combination of SNA functions that may be used. A given LU supports only sessions that draw from those allowable functions. A product is thus characterized by the LU-LU session type (or types) that it supports.

For a given LU-LU session, the session type and its associated profiles or specified in the BIND command when the session is established. The session type selected for the LU-LU session depends on the end-user's need. The type must be one that is supported by the implementation.

Third Edition (December 1979)

A form for readers' comments has been provided at the back of this publication. If this form has been removed, address comments concerning the contents of this publication to IBM Corporation, Technical Publications, Dept. 824, 1133 Westchester Avenue, White Plains, New York 10604. IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

PREFACE

This book introduces you to Systems Network Architecture (SNA) logical units and to sessions between logical units (LU-LU sessions). It discusses the concepts that apply to these sessions.

The book summarizes the relationship between users and the network, how information is transmitted over the network, and the types of LU-LU sessions that can be used. It is intended for systems programmers responsible for configuring and installing SNA products, and designing and developing application programs that use SNA products.

HOW THE BOOK IS ORGANIZED

Chapter 1, Introduction, tells what a communications system is and introduces the rules and structure for sessions between logical units.

Chapter 2, Protocols between Session Partners, discusses the basic protocols used to transmit information between session partners.

Chapter 3, Types of Logical-Unit to Logical-Unit Sessions, describes six types of LU-LU sessions that have been defined by SNA -- types 0, 1, 2, 3, 4, and 6. (Type 5 was an early attempt to define functions now known as type 6.) The characteristics of each type of session are discussed, including an overview of session rules, data stream characteristics, and profiles.

This book mentions IBM products as examples only; it does not define any specific equipment or programs that implement SNA. Specific implementation of the architecture in IBM products is documented in the publications for those products.

WHAT YOU MUST KNOW TO START READING

Some of the concepts described in Systems Network Architecture, General Information (GA27-3102) are not included in the introductory material in this book. If you have difficulty understanding the information in this book, you should read SNA General Information.

Since this book is an introduction to sessions between logical units, there is no further prerequisite reading. Only a basic understanding of data processing and data communications is assumed.

WHERE OTHER SNA INFORMATION IS FOUND

If you need more detailed information after reading this book, read:

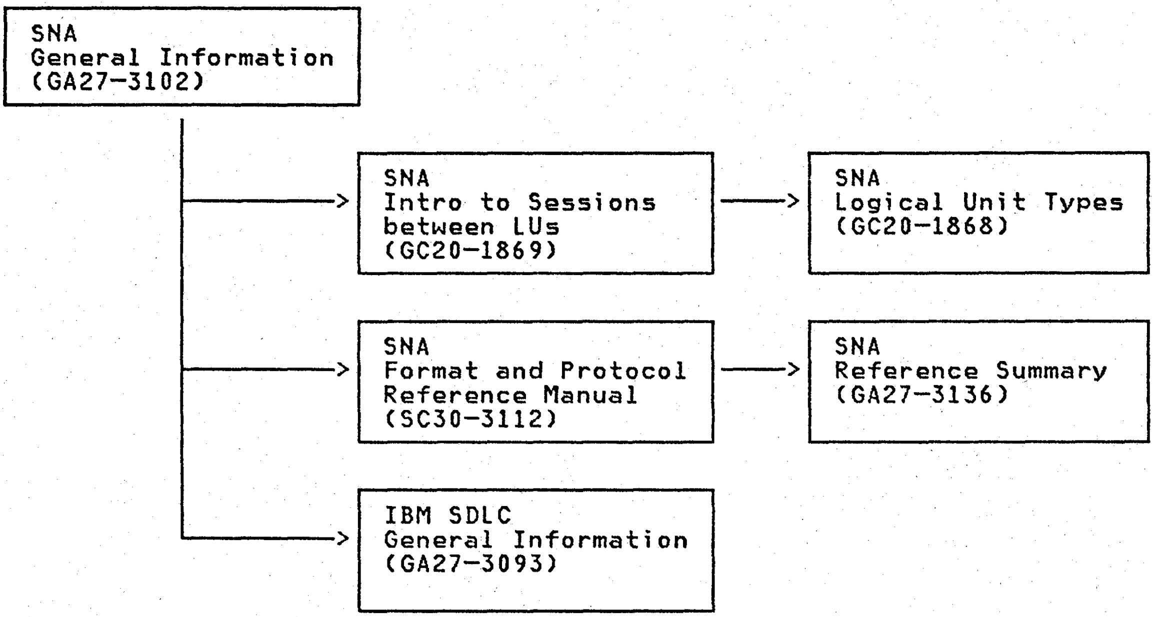
Systems Network Architecture, Logical Unit Types (GC20-1868), which provides design-level information on each type of LU-LU session.

Systems Network Architecture, Format and Protocol Reference Manual: Architecture Logic (SC30-3112), which contains detailed logic for the path control network and the layers in the logical unit that are responsible for preparing transmissions for or controlling the path control network. (The logical unit layers that handle transmissions are the data flow control and transmission control layers. The layers that handle the control functions are the service manager and network services.)

Systems Network Architecture, Reference Summary (GA27-3136), which provides selected reference information about SNA. This handbook contains summary material about the structure of SNA and SDLC (Synchronous Data Link Control), format and profile descriptions, and sense code definitions.

IBM Synchronous Data Link Control, General Information (GA27-3093), which introduces the terminology and concepts of SDLC.

A suggested reading order for SNA publications is shown below. In addition, there are many that discuss SNA from the perspective of specific IBM products. You may wish to ask your IBM representative for a list of these books.



CONTENTS

CHAPTER 1: INTRODUCTION

This chapter briefly describes data communications systems and how they are used. It shows what a communications system is and how Systems Network Architecture (SNA) guides the development of IBM communications networks. This chapter also describes SNA and data communications as they relate to logical units (LUs).

If you know that SNA is a set of rules, procedures, and structures for a network, and you understand the relationship between end-users, logical units, and the path control network, then you may begin with Chapter 2.

BACKGROUND

The first online telecommunications systems were created for large, complex, applications such as airline-seat reservations. Each system was often unique, and was constructed of customized hardware, software, and communications links. As technical knowledge and networking requirements expanded, it became apparent that most telecommunications systems should be assembled from commercially available products that were part of a unified communications system design. Systems Network Architecture (SNA) provides that unified design; it defines both the functions and the structure for IBM's communications products.

SNA has rules that network components must follow. For example, user A might be required to listen while user B talks, but B can interrupt A if necessary. SNA also has defined procedures that users must follow. For example, user A might tell user B to "be quiet for a minute and I will get back with you." User B knows precisely what this means. Said another way, the protocol, or code of etiquette, between users is defined so that there is no ambiguity when users interact.

Because telecommunications users have different requirements, their systems have different characteristics; but in the end, each system essentially receives information from one user and either transmits it in a usable form to another user or analyzes it and returns a reply to the originator.

Let's start with an example and build on it as ideas are presented. As a sales representative, you need to know the prices of products your company offers. You may get the prices from a sales catalog, but you and your company have to develop procedures to ensure that you use the latest version of the catalog. If your company keeps product prices in a computer data base, you may use a terminal attached to the computer to get the prices. You can obtain the latest price information and be freed from the responsibility of maintaining your copy of the latest prices.

If your company has a telecommunications network, you still need your terminal, but the terminal need not be directly attached to the computer. You and your terminal can be in a different city than the computer and the data base, but you can still obtain the product prices you need by "calling the computer." The computer is assigned a telephone number, and you dial the number as you would to call a person. The computer has equipment that automates the answering protocol, and when that protocol is finished, you request the price you need.

In an SNA network, you are one user (called an end-user by SNA) and the computer program that returns the price is the other user. In the simplest terms, the transactions appear as:

```
    |                         | Program
You | -----Price Request----> | to do
    | <----Price Reply------- | price
    |                         | lookup
```

Both you and the price program are end-users. The computer equipment (and programs) that connects you with your program is the SNA network (see Figure 1-1).

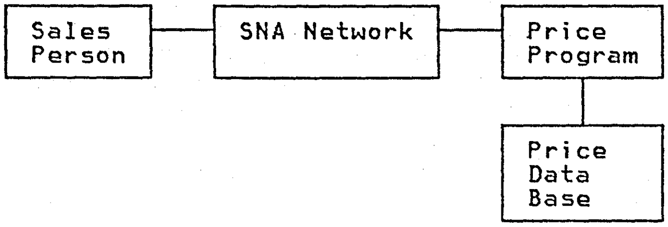


Figure 1-1. The SNA network is used to connect end-users who want to communicate with one another.

To a great extent, all you need to know is that there is a network and that you must follow certain procedures in order to obtain the desired price from the price program. If you follow these procedures, then the remainder of the network activity is transparent.

But within the SNA network, two components assist you in sending requests and responses: the logical unit (LU) and the path control network (see Figure 1-2).

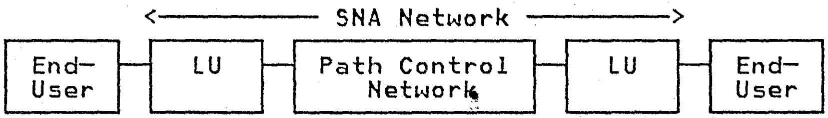


Figure 1-2. The logical unit (LU) and the path control network are the network components primarily responsible for sending requests and replies between end-users.

The LU assists you by providing services that simplify what you must do to talk to the other end-user. The LU also helps you avoid procedural mistakes. As Figure 1-2 shows, another LU represents your price program and provides it with similar services.

Continuing the example, your LU is in your display terminal. The LU helps you "sign on," or identify yourself to the network and to the price program. It may ensure that you are authorized to use the network, and help establish a communication path to the price program. It sends your price request to the path control network and waits for the reply.

The LU for the price program is in IMS/VS (Information Management System/ Virtual Storage). IMS/VS provides services for the price program just as your LU did for you. It receives your request from the path control network and gives it to the price program in a usable form. Finally, it sends the program's reply back to the path control network.

The path control network sends requests and responses between LUs. The path may be a direct link between the LUs or it may include several intermediate nodes. The path is transparent to the LUs; they are unable to detect if intermediate nodes are used.

The remainder of this chapter introduces the structure of SNA and shows, through logical units and layers of function, how the architecture supports the use of both individualized and standardized functions.

SNA - A SET OF RULES AND PROCEDURES

SNA is a set of rules that governs the format, definition, and sequencing of information sent through the network. Whether the information is data or a request (command) that controls network activity, it must be formatted and defined so as to be understood by network components. Since a variety of requests and responses can be sent and received, their sequencing also must be understood by network components. In other words, SNA defines formats and protocols to permit understandable communication between network components.

SNA - A STRUCTURE FOR A NETWORK

An SNA network provides services that enable pairs of end-users to communicate. The SNA structure defines three major elements (see Figure 1-2):

1. End-users. End-users are the sources and destinations of information that passes through an SNA network -- the users of the network. They are application-oriented components within IBM's hardware products or system and application programs. They could also be application programmers or operators of devices, although programmers and operators are generally considered users of IBM products.

2. Logical Units (LUs), Physical Units (PUs), and System Services Control Points (SSCPs).

   LUs provide the ports for end-user access to the network; they are the origins and destinations of message units flowing in the network.

   PUs manage device resources. They control resources common to all LUs in a node. A node is a grouping of LUs, PUs, and SSCPs joined by paths (communication links) to other nodes. We discuss this concept further under "The Network" below.

   SSCPs manage the configuration of the network. The PUs and the SSCP, in conjunction with one another and with LUs, provide a variety of configuration, maintenance, measurement, and network services. Since this book pertains to LUs, it discusses PUs and SSCPs only as they relate to logical units. For information on PUs and SSCPs, see SNA Format and Protocol Reference Manual, Architecture Logic.

3. Path Control Network. The path control network contains the elements that route information to its destination (a network address); it controls the flow of data between LUs, PUs, and SSCPs.

Sessions

Although end-users are the sources and destinations of most information flowing in an SNA network, they are not part of the network. They must be represented by an LU. For one LU to communicate with another, it must establish a formally bound pairing, called a session, between them. A session between LUs is called an LU-LU session. The session is a logical connection; it governs the protocol between and capabilities of the session partners.

Sessions can be started by LUs and SSCPs. The sessions that can be established are:

LU-LU
SSCP-LU
SSCP-PU
SSCP-SSCP

There are six types of LU-LU sessions. The session types meet at least these criteria:

LU-LU Session Type	Session Description
0	Data communications between two product-defined LUs using SNA-defined protocols.
1	Data communications between an application program and single- or multiple-device data processing terminals in an interactive, batch data transfer, or distributed processing environment. For example, an application program using IMS/VS and an IBM 3767, where the 3767 operator is correcting a data base maintained using the application program.
2	Data communications between an application program and a single display terminal in an interactive environment. For example, an application program using IMS/VS and an IBM 3277, where the 3277 operator is creating data and sending it to the application program.
3	Data communications between an application program and a single printer. For example, an application program using CICS/VS (Customer Information Control System/ Virtual Storage) to send data to an IBM 3284 attached to an IBM 3791 Controller.
4	Data communications between an application program and a single- or multiple-device data processing or word processing terminal in an interactive, batch data transfer, or distributed processing environment.
6	Data communications between two application programs in a distributed processing environment. For example, an application program using CICS/VS communicating with an application program using IMS/VS.

Session partners must use the same LU-LU session type. SNA does not permit, for example, one half-session to use session type 1 and the other to use session type 4.

Half-Sessions

The session is composed of two halves, or half-sessions. In an LU-LU session, each LU allocates some of its resources to support the session. These resources are called a half-session, and an LU can activate as many half-sessions as it has the resources to support. The LU can support simultaneously more than one session type. For example, it can support session types 1 and 2, but the type 1 half-sessions must be in session with type 1 half-sessions in another LU (and type 2 with type 2).

On the other end of the path, the session partner also allocates resources to support the session (its half-session) once the request to start a session is received. The session partner rejects the request if it cannot support the session type requested.

When LUs are bound together (in session), each agrees to use only those functions that its session partner can support.

You will find more information on sessions under "Types of Sessions" in Chapter 3.

The Network

The next piece in the structure, the path control network, provides transmission services for the sessions. It routes traffic between session partners. This relationship can be seen in Figure 1-3.

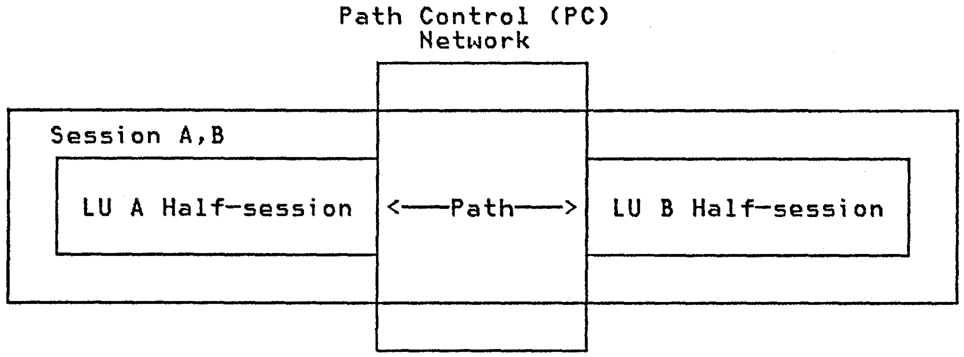


Figure 1-3. Each LU-LU session is composed of two half-sessions. The path control network is responsible for linking the half-sessions.

To route traffic, the path control network must know where each LU is located. This locating is done through network addresses assigned to the LUs, PU, and SSCP in the node. A <u>node</u> is a grouping of network addresses (one for a PU, optionally one for an SSCP, and optionally zero or more for the LUs) joined by links to other nodes (see Figure 1-4). Each node has one physical unit (PU). It may also have one System Services Control Point (SSCP) and many logical units (LUs). The PU is a grouping of functions that pertain to the physical resources of the node. The PU is not a piece of physical hardware, but a logical concept -- it's a way of defining (1) the contents of a node and (2) the way the node transmits data to other nodes.

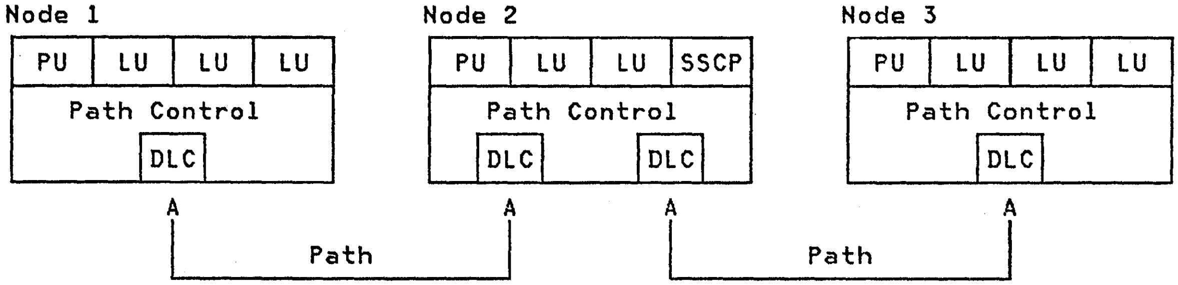


Figure 1-4. A node is a grouping of LUs, PUs, and SSCPs. Path Control knows the locations of adjacent nodes in the network and it uses data link control (DLC) functions to transmit messages to those nodes.

One SSCP is required in every SNA network. In simple networks where there is only one session, the SSCP functions can be performed by one of the session partners. As networks become more complex, one or more SSCPs become necessary, and each SSCP is assigned a portion of the network (a domain) to manage.

An SSCP's responsibilities include network initialization, session activation, and configuration management for all LUs and PUs within its domain. It manages, in coordination with the PUs in its domain, the resources that move data within the domain. It manages, in coordination with the LUs in its domain, the resources used by active LU-LU sessions. (For more information on sessions with the SSCP or PU, see <u>SNA Format and Protocol Reference Manual, Architecture Logic</u>.)

SNA - A SET OF PRODUCTS

Many IBM products use Systems Network Architecture protocols so that they can communicate in a compatible way with other IBM products. Although the protocols are standardized, the products vary widely in the way that their functions are packaged. IBM products package function as programs, hardware, or both. The function can be from one component of the architecture or from many.

Previously we said that an end-user is the source or destination of information and that an end-user is "external" to the SNA network. The architecture does not restrict the form of an end-user; it can be, for example, an application program, a product's microcode, or a storage medium such as a disk, tape, or card.

From a network point of view, the end-user is the final destination. The format and protocol used to communicate between end-users are defined by the architecture, but products that implement the architecture define additional product-unique protocols, called an interface, for the programmers and operators using the product. The product's interface allows for specialized and unique function; it lets the product "talk to" its users in a language the users can understand. For example, a product could have a language tailored for application programmers or specialized equipment for terminal operators.

IBM products package LU and end-user functions in many forms. Functions are frequently split between application subsystems and the application programs that those subsystems control. Although the combinations are nearly limitless, this book assumes, when discussing LU functions, that the LU and end-user functions are packaged separately (Figure 1-5).

Another important concept is that products implement only a portion of the architecture. SNA provides a framework for all products, and each product implements only those functions that are needed by that product. SNA regulates the way functions are grouped into sets by defining which functions apply to each LU-LU session type. These session types were introduced above and are discussed in detail in Chapter 3.

NETWORK STRUCTURE

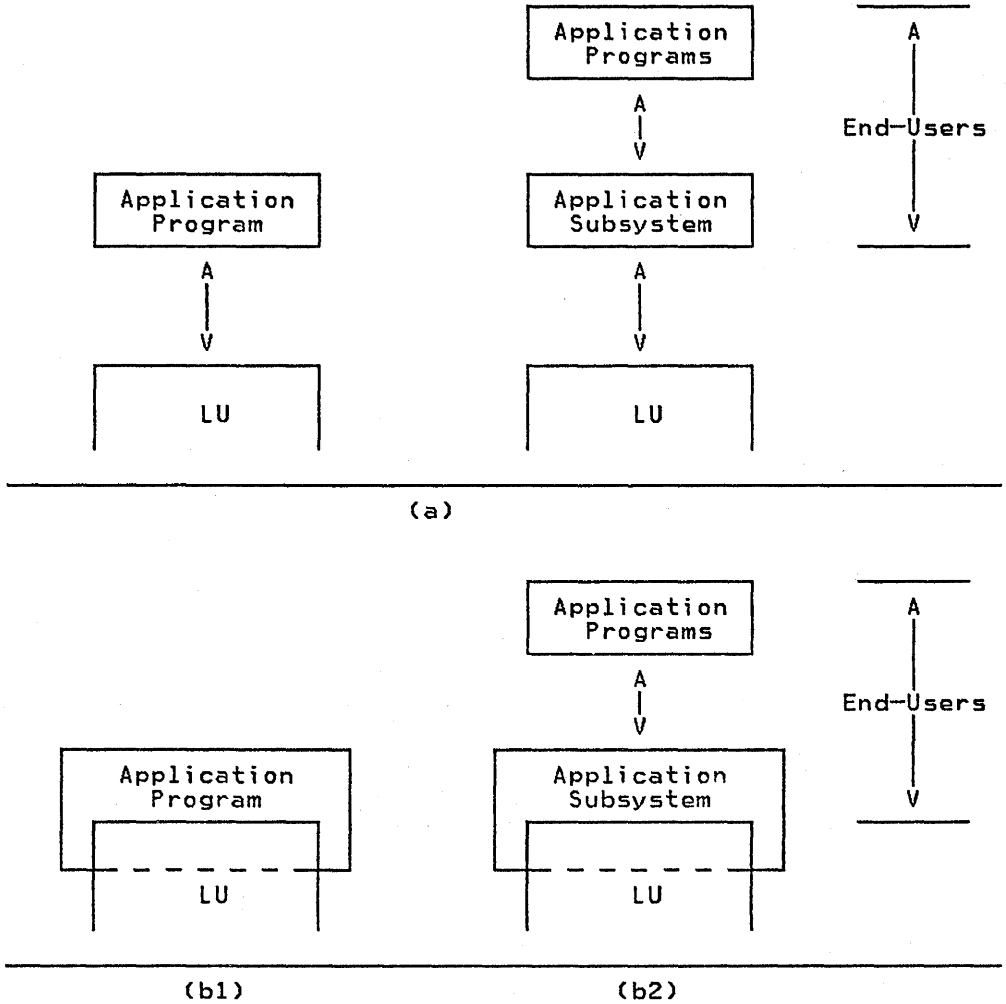
Although the architecture does not dictate product packaging, it shapes product packaging through the nodes we discussed above. SNA defines four node types -- 1, 2, 4, and 5. These numbers correspond to the node's physical unit type.

Informal usage also associates the following terms and PU types:

Term	PU Type
Host	5
Communication controller	4
Cluster controller	1 or 2
Terminal	1 or 2

A PU type 5 is at a node that (1) has an SSCP and (2) receives transmissions and passes them on, if required, to other nodes in the network. A PU type 4 is at a node which receives transmissions and passes them on, as required, but which doesn't have an SSCP. A PU type 2 is at a node with one or more LUs. Within each LU, secondary half-sessions can be active with primary half-sessions in multiple LUs. A PU type 1 is also at a node with one or more LUs, but secondary half-sessions can be active only with primary half-sessions in one LU.

The nodes containing either PU type 4 or 5 control *subareas* of the network (see Figure 1-6) and transmit information using network addresses. Many of the IBM products that use a node containing a PU type 5 have LUs that support application programming, but the node contains an SSCP and performs other tasks that help coordinate the activities within the SSCP's domain.



(a)

(b1) (b2)

Figure 1-5. Architecture is described assuming that LU and end-user functions are packaged separately (a). When products implement the architecture, LU and end-user functions may be packaged together, such as with CICS/VS (b1) and IMS/VS (b2).

The node containing the PU type 4 controls the communication paths for its subarea and provides transmission services to other nodes and other subareas. It does not normally contain LUs, but acts as a "pipe" for transmitting information between LUs, PUs, and SSCPs located in nodes other than its own.

The node containing a PU type 2 has no subarea responsibilities. Its primary responsibility is to control LUs supporting applications and devices. It requires help from a node containing a PU type 4 or 5 to complete the network services it offers the LUs. (A PU type 4 or 5 that completes these services is said to have a boundary function.)

A node containing a PU type 1 also controls LUs supporting applications and devices, but it has less function than a node containing a PU type 2. It also requires help from a node with a boundary function.

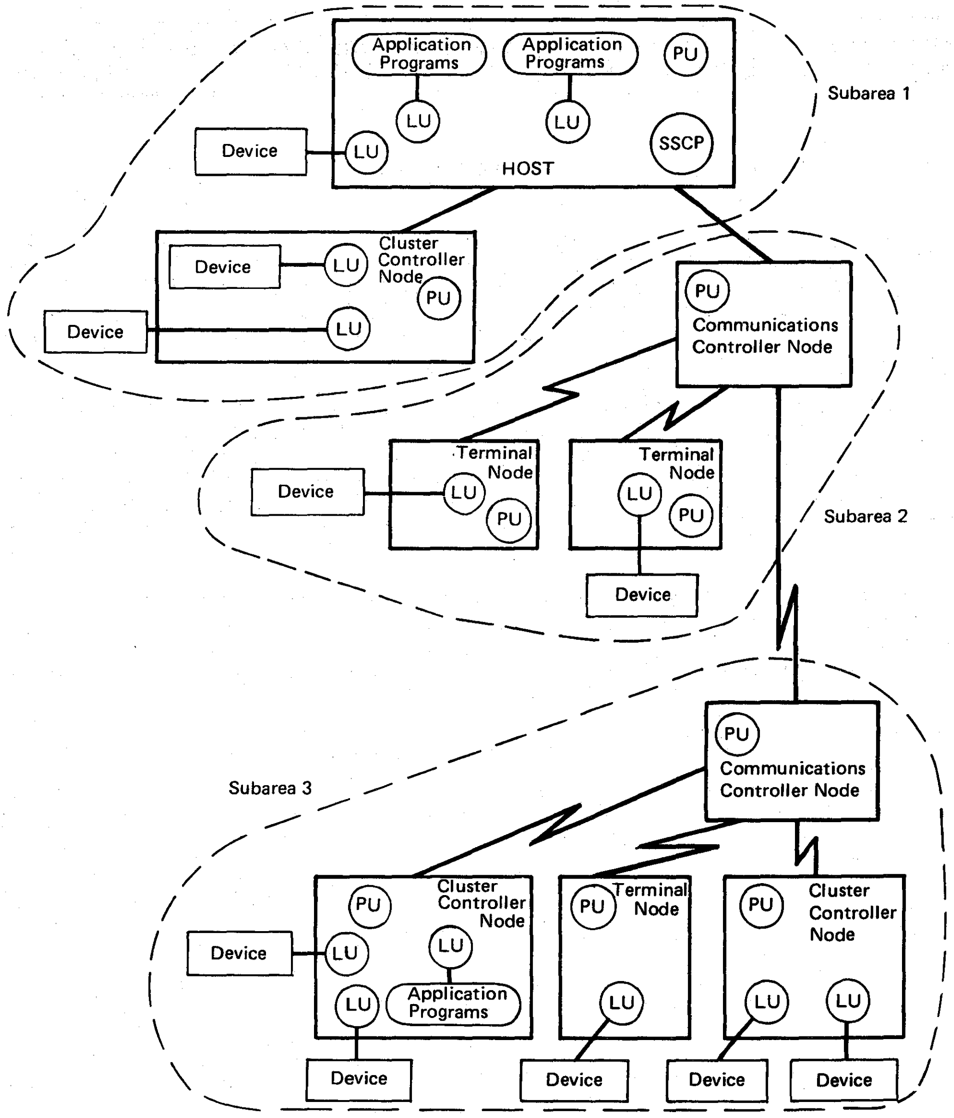


Figure 1-6. Three subareas within one domain.

THE PROTOCOL HEADERS

To this point, we have stated that path control routes message traffic between the session partners using standardized protocols. That message traffic also has a standardized format, and when it is sent over a communication link, it is known as a BLU (basic link unit). A number of headers are prefixed to the data sent from one LU to another. The headers are used to pass information between network components (see Figure 1-7).

Link Headers and Trailers: The path control network creates link headers and link trailers to control the physical flow of message units on the communication links. The link headers enable the path control network to route a BLU, for example, to one station on a link with multiple stations.

Link Header	Transmission Header (TH)	Request/Response Header (RH)	Request/Response Unit (RU)	Link Trailer

Figure 1-7. Structure of a BLU (basic link unit).

Transmission Header: Path control uses the transmission header (TH) to control the routing of message units through nodes of the network. The TH is built by the half-session and contains the addresses of the LU that originated the message and the LU that is to receive it. The TH also contains a sequence number that is used by the LU to aid in resynchronizing the session after errors.

Request/Response Header: All message traffic between LUs is sent either as a request or as a response to a request. The LU builds the request/response header (RH) to send control information and to define the content and format of the request unit or response unit. Within the LU-LU session, the half-session handles this responsibility. For details on the control information included in the RH, see "Data RU" in Chapter 2.

Request/Response Unit: The request/response unit (RU) contains data and data stream controls. The data is used by the end-user or LU, and the controls identify what is to be done with the data. (There is more information on data streams in Chapter 3 under the discussions of LU-LU session types.)

CHAPTER 2: PROTOCOLS BETWEEN SESSION PARTNERS

Chapter 1 introduced the structure of an SNA network and showed how messages are sent between end-users. It showed that information is exchanged between logical units in the request/response headers (RHs) and request/response units (RUs) of message units. It also showed that an RH can contain a request or a response, and that an RU can contain data and data stream controls.

To transmit messages, a session must be started and the half-sessions must agree on the protocols to follow. This chapter describes many of the protocols that are used by LU-LU sessions and shows how they are combined into a dialog between end-users.

STARTING LU-LU SESSIONS (THE BIND COMMAND)

Before end-users can communicate, you must establish a communication path through the various devices and programs that will handle your session messages. The process you use to establish this communication path is called session initiation.

You must complete many steps to initiate a session, but a key step is to send a BIND command from one LU to the desired session partner. The BIND command is the way that you specify the protocols for the LU-LU session, and is discussed below. If you need to understand all the steps to establishing a session, see SNA Format and Protocol Reference Manual, Architecture Logic.

SNA lets you designate one of the session partners as the primary logical unit (PLU) and one as the secondary logical unit (SLU). The PLU is responsible for deciding which session protocols will be used (by sending BIND), and for handling error recovery. The SLU is responsible only for adhering to the protocols specified in BIND. In a sense, there is a master/slave relationship between the PLU and SLU.

All LU-LU session types allow PLU-SLU sessions, but LU-LU session types 0, 4 and 6 do not require that half-sessions use this master/slave relationship. These session types allow either half-session to send BIND, and give the half-sessions equal responsibility. For example, both partners are responsible for correcting their own errors.

You use the BIND command to specify the relationship between session partners. Valid relationships are:

Session Type	Partner Relationship
0	PLU-SLU or Equals
1	PLU-SLU
2	PLU-SLU
3	PLU-SLU
4	PLU-SLU or Equals
6	PLU-SLU or Equals

The BIND Command

During session initialization, an LU sends a BIND command to its desired session partner to establish the protocols for the session. The LU sending BIND is called the BIND sender. The LU receiving it is called the BIND receiver. When a BIND command is received, the BIND receiver examines the proposed session protocols and, depending on their acceptability, either accepts or rejects the session.

The process the BIND receiver uses to accept or reject a BIND command depends on the type of BIND received. There are two types of BIND: non-negotiable and negotiable. The non-negotiable BIND has the BIND sender establishing the session parameters without compromise on potentially negotiable characteristics. If the SLU's send capability is consistent with the BIND parameters specified by the PLU, it responds positively; otherwise, it responds negatively. You may use the non-negotiable BIND on any LU-LU session type.

With the negotiable BIND, the BIND sender proposes a set of session parameters. The BIND receiver examines the parameters and (1) accepts them by returning a positive response with the same session parameters, (2) "negotiates" them by returning a positive response with different parameters, or (3) rejects them by returning a negative response. You may use the negotiable BIND when establishing LU-LU session types 0, 4 and 6.

Understanding the BIND Parameters

Once the BIND is accepted by the session partners, the session is activated and transmissions can begin according to the protocols specified in BIND. These protocols are discussed in this section and Chapter 3, which contains the format of the BIND command.

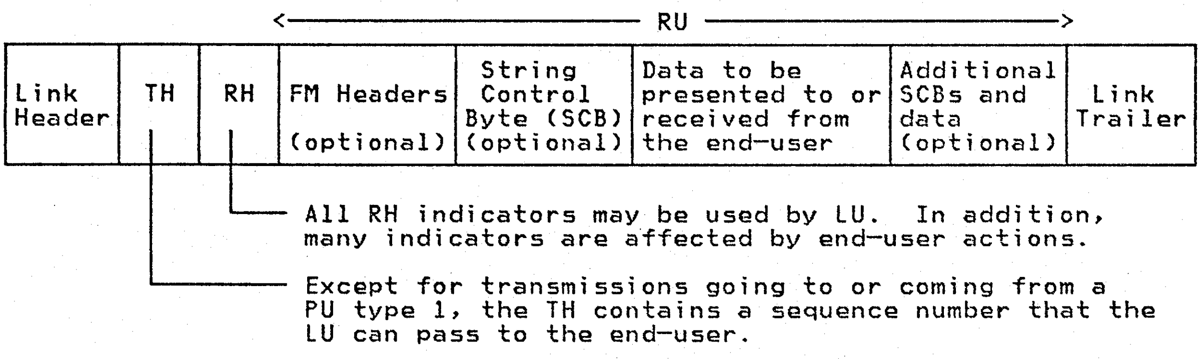
Transmissions can contain data, commands, and responses. Each category is discussed below.

Before reading about the BIND parameters that specify protocols, you must understand more about request and response units (RUs). Request units perform several functions; they are used to send end-user data between LUs, to send commands, and to acknowledge receipt of another RU.

Data RUs: Data RUs are request units that contain information to be exchanged between end-users. They are the SNA vehicle for transmitting user information.

When one session partner is ready to send data, or is in the process of sending data to the other session partner, it is said to be the sender. When one session partner is ready to receive data, or is in the process of receiving data from the other session partner, it is said to be the receiver. These terms become important in subsequent discussions, because most sessions use protocols that make it impossible for both partners to either send simultaneously or receive simultaneously.

Data RUs are formatted as shown in Figure 2-1. An indicator (RRI field) in the request/response header (RH) is used to indicate a request. Another RH field, the RU Category field, is used to show that the RU is a data RU.



All RH indicators may be used by LU. In addition, many indicators are affected by end-user actions.

Except for transmissions going to or coming from a PU type 1, the TH contains a sequence number that the LU can pass to the end-user.

Figure 2-1. Format of a data RU with its associated headers.

The end-users, by their actions during a session, can affect the TH and RH. They can affect only the sequence number in the TH, but can affect the RH indicators shown below. (We will explain the meaning of the indicators as they are discussed in later sections of this chapter. If you need to know the position of each indicator in the RH, see <u>SNA Reference Summary</u>.)

<u>RH Indicator</u>	<u>Meaning</u>
BBI	Begin bracket
BCI	Begin chain
CDI	Change direction
CSI	Code selection
DR1I	Definite response 1
DR2I	Definite response 2
EBI	End bracket
ERI	Exception response
ECI	End chain
FI	Format (of RU)
RRI	Request/response
RTI	Response type
SDI	Sense data included

Except for LU-LU session types 2 and 3, the data RU can contain one or more headers at the beginning of the first RU in a chain. These headers are called function management headers (FM headers), and their position in the RU is shown in Figure 2-1. The headers describe how data in this or subsequent RUs must be processed. The format indicator (FI) is set on in the RH of a data RU when an FM header is present. (FM headers are discussed in Chapter 3 under "Types of Sessions".)

You use BIND parameters to indicate how the half-sessions support these headers. You can specify that half-sessions cannot use headers, can have limited capability, or can have full capability. The option you select depends on whether the LU-LU session type allows headers and on how much header capability you need.

For LU-LU session types 1 and 4, the data RU also can contain one or more SCBs (string control bytes). SCBs describe how the data stream is compressed or compacted. Their position in the RU is shown in Figure 2-1. They are described under "Data Compression and Data Compaction" below.

<u>Command RU</u>: Request units that control the flow of information are called <u>command RUs</u>. Each command RU contains one command. A command RU can be sent by any half-session in the network, but the commands that are authorized are different for each session type. For LU-LU sessions, the commands that are authorized are defined by the <u>profiles</u> specified in BIND (see "Profiles and Usage Fields" in Chapter 3).

There are two categories of commands on LU-LU sessions: data flow control and session control. The commands that control the flow of data are:

BID	Bid to send data
BIS	Bracket initiation stopped
CANCEL	Cancel chain
CHASE	Chase (request all responses outstanding)
LUSTAT	LU status information
QC	Quiesce complete
QEC	Quiesce at end of chain
RELQ	Release quiesce
RSHUTD	Request shutdown
RTR	Ready to receive
SBI	St bracket initiation
SHUTC	Shutdown complete
SHUTD	Shutdown
SIG	Signal session partner

The commands that control the session are:

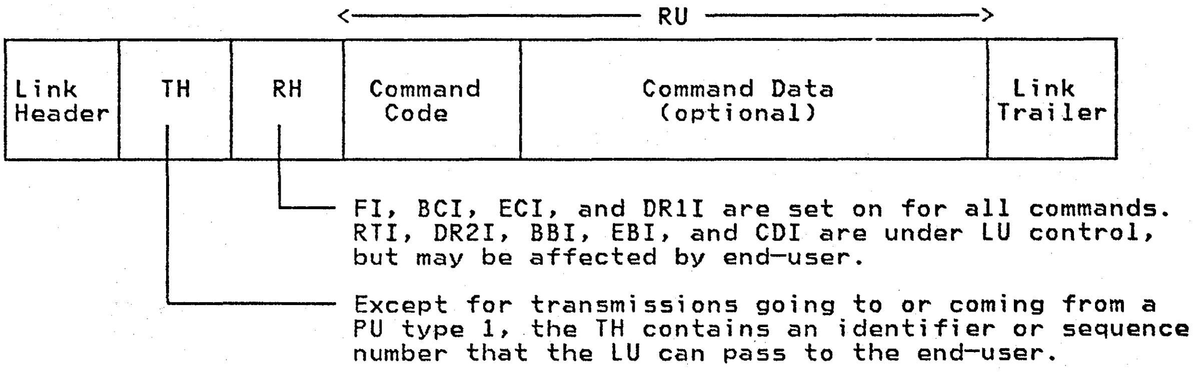
CLEAR	Clear data traffic
CRV	Cryptographic verification
SDT	Start data traffic
STSN	Set and test sequence numbers
RQR	Request recovery

We will explain the meaning of the commands as they are discussed in later sections of this chapter. If you need to know the restrictions on these commands by LU-LU session type, see "Profiles and Usage Fields" in Chapter 3. If you need a complete description of a command or need its format, see <u>SNA Reference Summary</u> or <u>SNA Format and Protocol Reference Manual, Architecture Logic</u>.

Although data flow control and session control commands affect the LU-LU session, they may or may not be passed directly to the end-user. The LU could, on receipt of the command, process it and send an acknowledging response to the sender, and only report its receipt via a message or a code to the end-user.

Because their responsibilities differ, primary LUs may have command capabilities that differ from secondary LUs. The profiles and usage parameters specified in the BIND identify which commands may be sent by each partner.

Command RUs use the same transmission format as data RUs, except that FM headers and SCBs are not used (see Figure 2-2). Instead, the RU contains a command code followed by optional command data.



FI, BCI, ECI, and DR1I are set on for all commands. RTI, DR2I, BBI, EBI, and CDI are under LU control, but may be affected by end-user.

Except for transmissions going to or coming from a PU type 1, the TH contains an identifier or sequence number that the LU can pass to the end-user.

Figure 2-2. Format of a command RU with its associated headers.

The SIG (signal) and LUSTAT (LU status) commands enable LUs to send command data to each other. The command data can be any information that is meaningful to the session partners.

Remember that commands are requests; sometimes they can be ignored.

<u>Response Units</u>: When a half-session sends a data RU for the end-user or a command RU for itself, it has no way of knowing if the session partner received the RU. To avoid this ambiguity, the session partner acknowledges receipt of the RU by sending an acknowledgment called a <u>response RU</u>. Either session partner may send a response unless the BIND parameters prohibit responses (see "No Response" below).

There are two types of responses -- positive and negative. There are three response protocols -- definite response, exception response, and no response. The response types are shown in Figure 2-3.

A <u>positive response</u> acknowledges the successful receipt of an acceptable request.

A <u>negative response</u> acknowledges the receipt of a request that the receiver determines to be unacceptable. A negative response would be

Half-Session	Session Traffic	Half-Session
Data request sent asking for an acknowledging response	Data ——————————————>	Processed by LU successfully
	+RSP <- - - - - - - - - - - - - -	Return positive response
Data request sent asking for an acknowledging response	Data ——————————————>	Processed by LU unsuccessfully
	-RSP <- - - - - - - - - - - - - -	Return negative response (with sense data) notifying session partner of the error
SNA command sent	Command ——————————————>	
	+RSP <- - - - - - - - - - - - - -	Return positive response on receipt of all commands (except for certain BID and RTR conditions)

Figure 2-3. Three uses of positive and negative responses.

made when the sender violated an SNA protocol, the receiver could not understand the transmission, or an error or an unusual condition occurred.

The response type indicator (RTI) in the RH indicates a positive (0) or negative (1) response. Figure 2-4 shows the format of positive and negative responses.

Sense data is sent with a negative response to notify the sender of the type of error. Sense data is a four-byte code that provides specific information about the error. The first two bytes are SNA-defined and indicate the error category. Some codes are reserved for user-defined sense data, which must follow in the last two bytes. In the other categories, specific system sense code information may follow in the last two bytes.

Most positive responses are sent without data. Positive responses to some commands, such as BIND and STSN (set and test sequence numbers), require accompanying data. Responses to these commands are discussed in the _SNA Reference Summary_.

SNA defines three response protocols that govern when positive and negative responses are to be sent:

Definite response. A half-session requests a definite response to its transmission if it wants an acknowledging receipt from the session partner. The session partner returns a positive response if the transmission is acceptable; it returns a negative response if unacceptable. A definite response is requested by the sender when DR1I, DR2I, or both are set on in the RH (and ERI is set off). These indicators remain on in the response.

A half-session must request a definite response (sets DR1I on) whenever it sends a command RU. If the session partner receives the

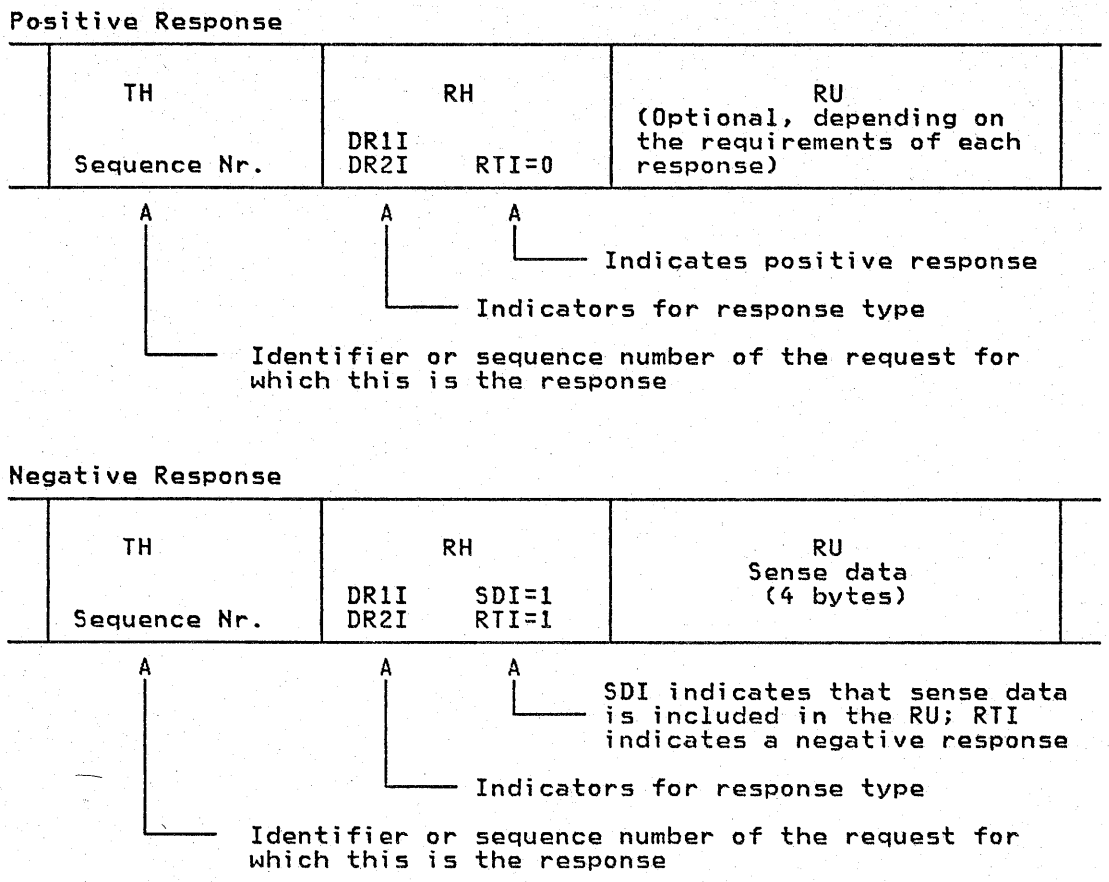


Figure 2-4. Format of a response.

command and no errors are detected, a positive response is returned. There are normal conditions that may require a negative response to BID and RTR (ready to receive), however, and these are discussed later.

Exception response. A half-session requests an exception response to its transmission when it wants to be advised only of error conditions. The session partner returns a negative response only if the transmission is unacceptable; otherwise it does not respond. An exception response is requested by the sender when ERI is set on in conjunction with DR1I, DR2I, or both. These indicators remain on in the response, but you should note that ERI becomes the negative response indicator (RTI) in the response.

No response. A half-session wishes no response returned when data RUs are sent. The session partner does not respond to the sender; there is no indication whether the transmission was acceptable or unacceptable. A no-response situation is created when the sender has set neither DR1I, DR2I, nor ERI.

Half-sessions might create errors or lose data if they use the no-response protocol; they should not use it when transmission accuracy is required.

Implementors should select (in BIND) a response protocol that is consistent with the half-sessions's data integrity requirements. You may add unnecessary line traffic if you use definite response protocols for all transmissions, so you are permitted to use a combination of definite and exception responses. Remember, however, that you cannot change the response protocol(s) during the session as the data integrity requirements change.

Figures 2-5 through 2-7 illustrate the types of responses and their protocols. If an RH indicator is shown, it is set to 1 unless otherwise stated. The LUs are shown communicating in immediate response mode, where responses must be returned in the order in which the requests are received. Request and response modes are discussed after normal and expedited data flows are discussed.

Half-Session	Session Traffic	Half Session
Data request sent requiring a definite response (DR1I, DR2I, or both)	Data ————————————>	Processed by LU successfully
	+RSP <- - - - - - - - - - - - -	Positive response returned (DR1I, DR2I, or both, as requested)
Data request sent requiring a definite response (DR1I, DR2I, or both)	Data ————————————>	Processed by LU unsuccessfully
	-RSP <- - - - - - - - - - - - -	Negative response returned (DR1I, DR2I, or both, as requested, plus RTI)

Figure 2-5. Responses are returned if session partner requires definite responses.

Half-Session	Session Traffic	Half Session
Data request sent requiring an exception response (DR1I, DR2I, or both, plus ERI)	Data ————————————>	Processed by LU successfully (no response returned)
Data request sent requiring an exception response (DR1I, DR2I, or both, plus ERI)	Data ————————————>	Processed by LU unsuccessfully
	-RSP <- - - - - - - - - - - - -	Negative response returned (DR1I, DR2I, or both, as requested, plus RTI)

Figure 2-6. Responses are returned if session partner requires exception responses.

Half-Session	Session Traffic	Half Session
Data request sent requiring no response (DR1I, DR2I, and ERI are set to zero)	Data ——————————>	Processed by LU unsuccessfully
Data request sent requiring no response	Data ——————————>	Processed by LU successfully

Figure 2-7. Responses are not returned if the session partner requires no responses.

Normal and Expedited Flows: Data requests, some command requests, and their respective responses make up the *normal flow* between LU half-sessions. Normal flow is handled sequentially (first in, first out). If responses are demanded for any two requests, the response to the earlier request must arrive first (immediate response mode).

An independent *expedited flow* comprises selected command requests and their responses that bypass the normal-flow queues and protocol sequences. End-user data is not carried on the expedited flow, and end-users cannot select whether normal or expedited flow will be used. If you want to know which commands flow expedited, see *SNA Reference Summary*.

Session partners may exchange information using the expedited flow when normal flow has been closed. For example, an LU can send an expedited SIG (signal) command to its session partner with four bytes of sense data. Only one expedited command may be sent at any one time; an acknowledging response must be received before another expedited command can be sent.

Request and Response Modes: Control modes can be specified in BIND to let implementations regulate data traffic and to better manage error recovery situations. The modes to be used for traffic in one direction may be chosen independently of, and do not affect, the modes to be used for traffic in the other direction. The modes are:

Immediate and Delayed Request Modes

For immediate request mode, the following rules apply:

After sending a transmission that requires a definite response, the sender must wait for the response before sending another request. (If the sender chains transmissions together, it must wait for the response to the chain, but we will discuss chains in the next section.)

Many no-response RU transmissions or exception-response RU transmissions may be outstanding at any one time, but only when there is no outstanding definite-response RU transmission.

In delayed request mode, there are no constraints on the sending of data or command RUs on the normal flow.

Immediate request mode is used on the expedited flow in each direction. Either request mode may be used on the normal flow.

Immediate and Delayed Response Modes

Immediate response mode means that responses are always returned in the order in which the transmissions are received. When request 1 is received before request 2, the response to request 1 must be returned before the response to request 2. A response need not follow the request immediately.

Delayed response mode means that responses may be sent in any order.

The normal flow uses either immediate or delayed response mode, as specified in BIND. The expedited flow requires immediate response mode.

DATA CHAINING

The size of each single data request between session partners is limited by an RU-size parameter specified in BIND. To exchange a message that is longer than the maximum RU value specified in BIND, you may split the information and send it as a series of separate requests (RUs). A series of data RUs of this kind is called an RU chain. Although you send the message as separate RUs, you consider each chain as a single message.

Chains consist of one to any number of data RUs (requests). Command RUs are chains, although they are always single-RU chains. Responses (RUs) are single-element chains also.

The beginning of a chain is specified by the BCI (begin chain indicator) in the RH of the first transmission in the chain. The end of the chain is specified by the ECI (end chain indicator) in the RH of the last transmission in the chain. Middle elements in the chain have both BCI and ECI set off. A single-element chain has both BCI and ECI set on. For those who are reading this book as a prerequisite to more detailed books on SNA, a more precise notation is:

First RU in chain	(BC,¬EC)
Middle RU in chain	(¬BC,¬EC)
Last RU in chain	(¬BC, EC)
Only RU in chain	(BC, EC)

When reading the middle RU in chain notation, for example, you read "not begin chain, not end chain."

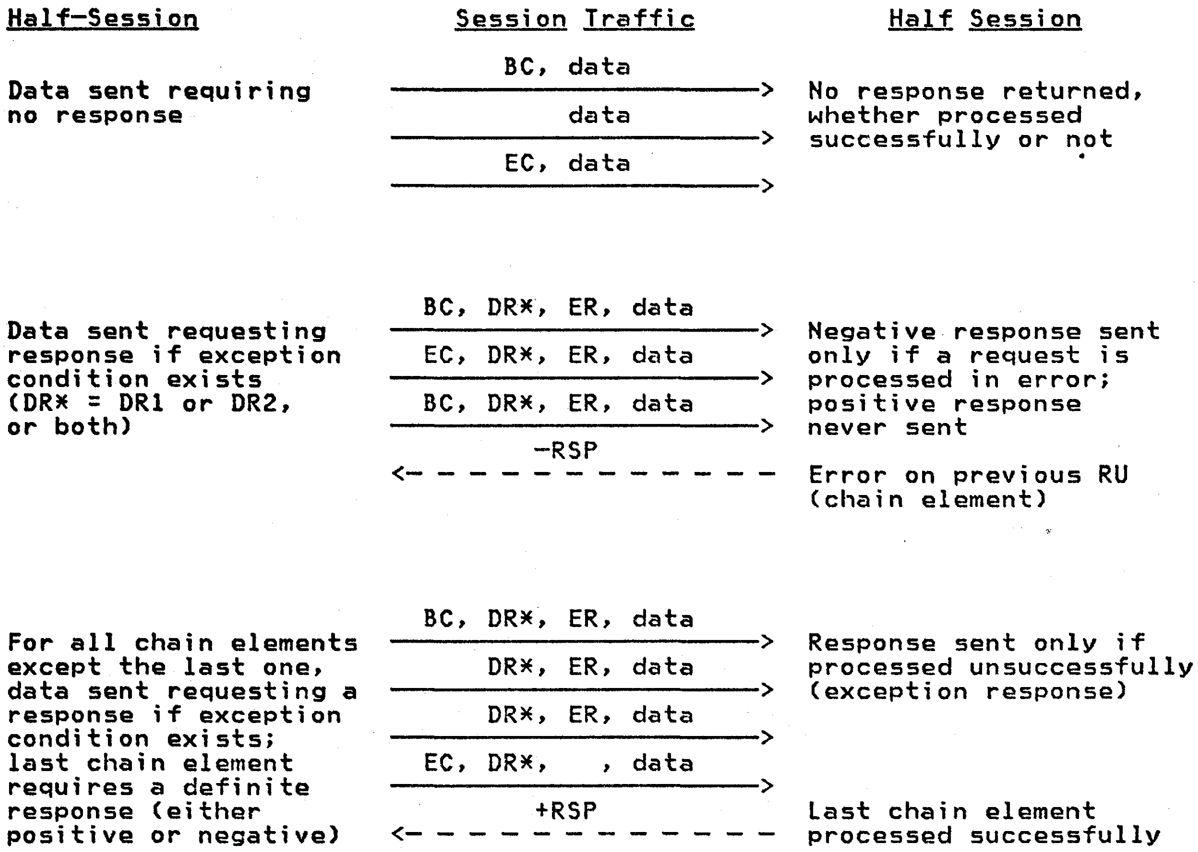


Figure 2-8. Rules for responding to chains.

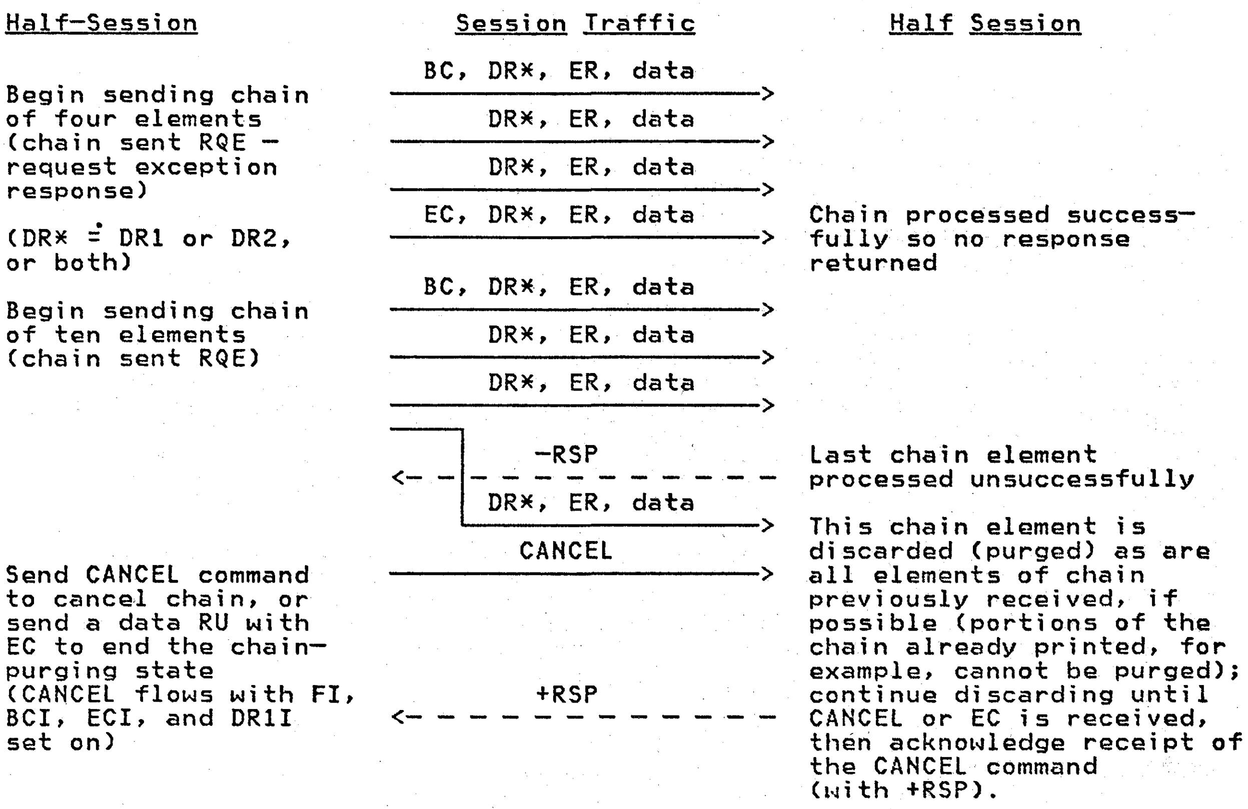
We said above that session partners could use definite-response, exception-response, a combination of definite- and exception-response, or no-response protocol. Once a protocol is specified in BIND, it is used for the entire session. Since the chain is the basic unit for error recovery, a single chain can receive no more than one response. The rules for responses to chains are described below and shown in Figures 2-8 through 2-10.

<u>No-response chain</u>. Each element in the chain indicates no response (DR1I,DR2I,ERI combined = 000).

<u>Exception-response chain</u>. Each element in the chain asks for an exception response (a negative response if that specific element is unacceptable, and no response if the element is acceptable) (DR1I,DR2I,ERI = 011 | 101 | 111).

<u>Definite-response chain</u>. Each element in the chain except the last asks for an exception response; the last element in the chain asks for a definite response (for the last element, DR1I,DR2I,ERI = 010 | 100 | 110). The definite response may be a positive or negative response. If negative, the entire chain is rejected.

A positive response to each element of a chain is not allowed. A positive response is reserved for the last element. When a sender receives no negative responses and then receives a positive response to the last element in the chain, it knows, with miminal network overhead for acknowledgments, that the entire chain was processed successfully.



Begin sending chain of four elements (chain sent RQE - request exception response)

(DR* = DR1 or DR2, or both)

Chain processed successfully so no response returned

Begin sending chain of ten elements (chain sent RQE)

Last chain element processed unsuccessfully

This chain element is discarded (purged) as are all elements of chain previously received, if possible (portions of the chain already printed, for example, cannot be purged); continue discarding until CANCEL or EC is received, then acknowledge receipt of the CANCEL command (with +RSP).

Send CANCEL command to cancel chain, or send a data RU with EC to end the chain-purging state (CANCEL flows with FI, BCI, ECI, and DR1I set on)

Figure 2-9. Use of chaining of requests.

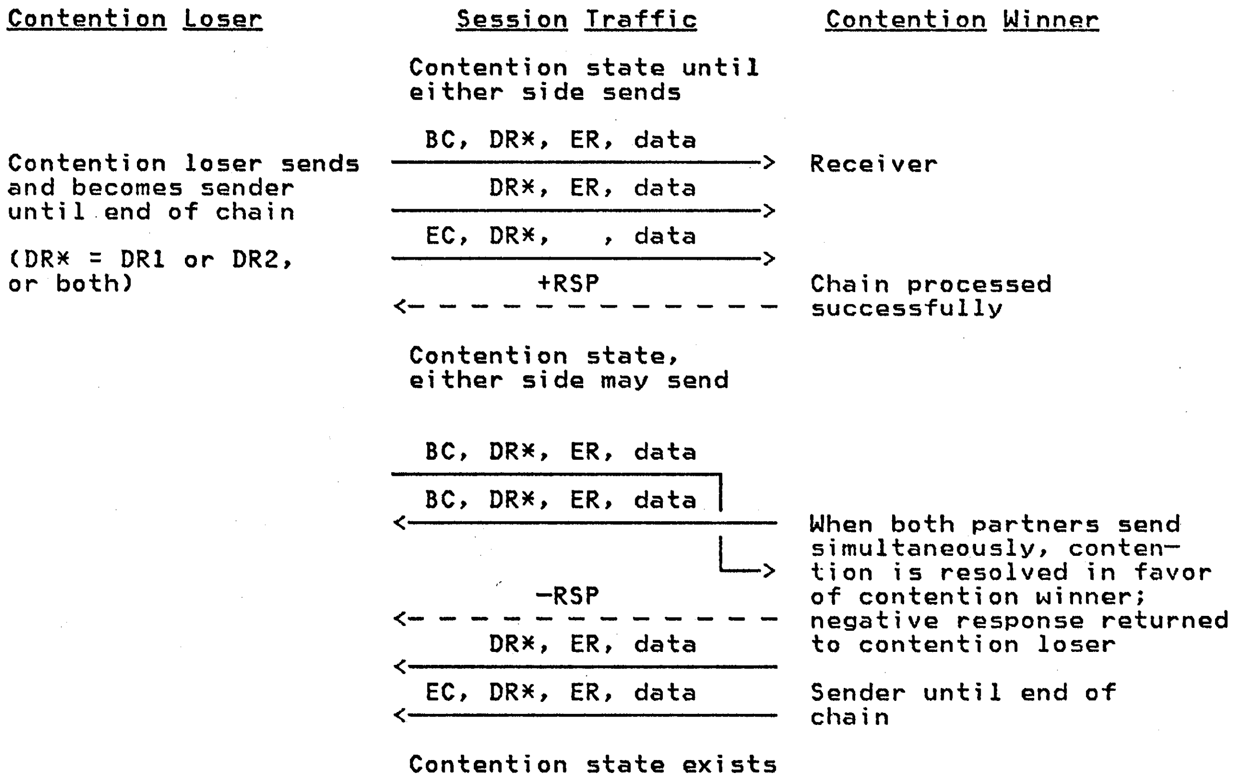


Figure 2-10. Communicating using HDX-Contention (half-duplex contention).

CANCELING A CHAINED TRANSMISSION

While sending a chain, the sender may decide to cancel it. This decision may be made independently or as a result of receiving a negative response to a chain element from the session partner. To cancel as a result of a negative response, the sender turns on the end-chain indicator (ECI) in the next data RU sent. To cancel a chain prematurely, the sender issues a CANCEL command. CANCEL tells the receiver that the chain has ended and that all elements of the chain already received should be discarded. CANCEL may be sent only when a chain is in process.

While receiving a chain, the receiver implicitly "cancels" a chain by sending a negative response to a chain element. After sending a negative response or after receiving CANCEL, the receiver discards the elements of the current chain already received. If a negative response is sent to the last chain element, the entire chain is discarded (purged).

TRANSACTION MODES

To this point, we have talked about sending and receiving requests and how they are chained, but have not discussed how the session partners know when they can send and when they must receive (or, thought of another way, who talks and who listens).

To send data RUs, or to send command RUs on the normal flow, a half-session must be in send state. A half-session cannot send data RUs or normal-flow commands when it is in the receive state, but it can send expedited commands and all responses. The protocols for entering the send or receive state are determined by the following transaction modes.

Full Duplex (FDX): Both session partners can send and receive data simultaneously; that is, both are in send state and receive state concurrently. Session traffic in one direction is independent of traffic in the other direction.

Half-Duplex Contention (HDX-Cont): In this mode, either session partner can begin sending a chain. If the other partner does not simultaneously want to send, the sending continues until the chain is completed.

A contention situation exists only when both partners try to send simultaneously. Both session partners cannot be the sender at the same time. The contention situation is resolved according to the parameters specified in the BIND, which identifies the LU that is the contention winner. This designation lasts only for the duration of the current chain. After that, contention is entered again. Figure 2-10 shows this protocol.

One of the items you consider when selecting a contention winner is the half-session's processing capability. For example, you would normally choose a terminal over a program because the program can buffer the data it wishes to send and resend it at the next contention opportunity.

Half-Duplex Flip-Flop (HDX-FF): In this mode, the session partners take turns being the sender. The sender allows the other partner to become the sender by setting on the change-direction indicator (CDI) in the RH of the last element of a chain. Figure 2-11 shows this protocol. A BIND parameter indicates which half-session has permission to send first.

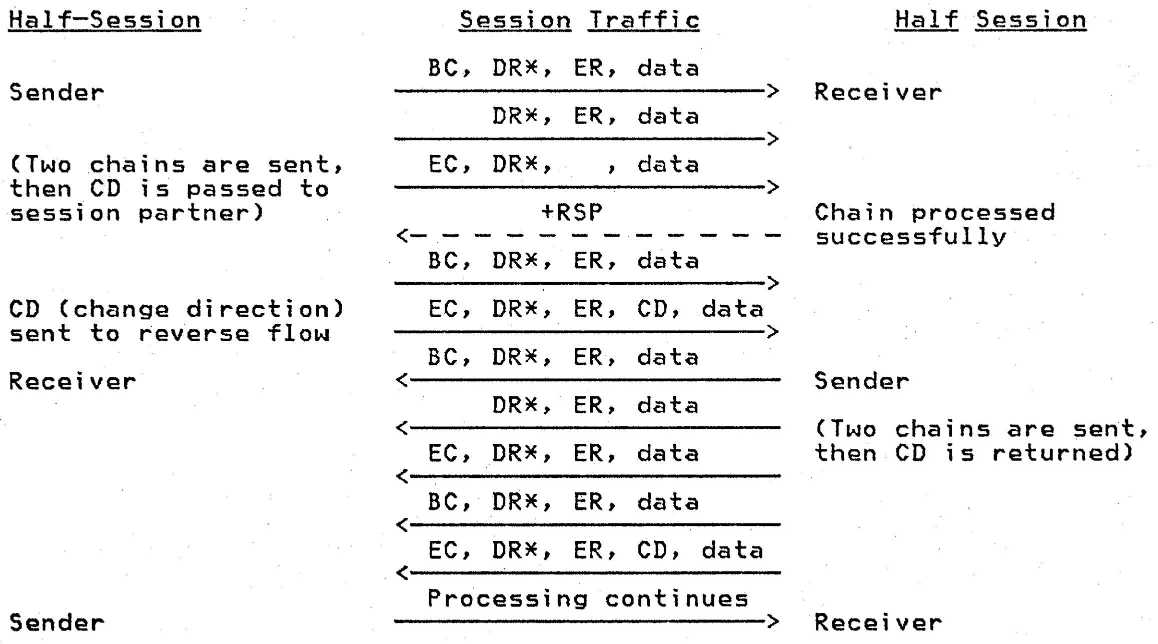


Figure 2-11. Communicating using HDX-FF (half-duplex flip-flop).

You can have a contention situation when in HDX-FF mode, but only if you use brackets to group your message traffic (brackets are explained in the next section). When you are between bracketed groups of messages, you have contention and either session partner can send. Once you start a bracketed group of messages, you follow the HDX-FF protocol. (For more information, see "Bracket Protocol" below.)

The HDX-FF protocol applies only to requests on the normal flow. Responses to requests and expedited requests are, in effect, independent.

The SIG (signal) command, an expedited request, can be used by the receiver to request that the data flow direction be changed. When SIG is sent, the receiver asks either that the direction of flow be changed immediately or that it be changed at the next logical break point. The direction of flow changes when the half-session that currently is sending data sets on the CD (changed direction) indicator in the RH of the last segment of a chain.

If a receiver receives CD (change direction) but has no data to send, it can return CD by sending the LUSTAT (LU status) command or a data RU with no data.

Most SNA implementations use half-duplex flip-flop protocol.

BRACKET PROTOCOL

Just as data RUs may be grouped into larger units called RU chains, RU chains may be grouped into larger units of work called brackets. The exact elements grouped into a bracket depend on the type of work being done by the session partners. A bracket may be a monolog, where one session partner sends a series of RU chains, or it may be a dialog, where both session partners exchange RU chains. In all cases, a bracket defines what that session partner considers a single unit of work.

The BIND parameters specify whether brackets will be used and which session partner will be the first speaker. The first speaker has the freedom to begin a bracket without first asking permission from the session partner. The other partner must ask permission of the first speaker to begin a bracket; this partner is called the bidder. If both begin a bracket simultaneously, the first speaker wins the contention and may continue transmitting.

The bidder can use the BID command to request permission to begin a bracket. A positive response to the BID command indicates that the first speaker grants permission to begin the bracket; a negative response denies permission.

When permission is denied, the first speaker may tell the bidder that a RTR (ready to receive) command will follow when the first speaker can grant permission to start a bracket.

If brackets are used, all data RUs must be contained within brackets.

Figure 2-12 illustrates how brackets are used. Brackets are delimited by a begin-bracket indicator (BBI) set on in the RH of the first RU of the first chain of the bracket, and an end-bracket indicator (EBI) set on in the RH of the first RU of the last chain of the bracket. A single chain may be a bracket if it contains both the BBI and EBI set on in the first RU of the chain.

The bidder may attempt to begin a bracket without first receiving permission, but only if a definite response is requested in the last RU of the first chain in the bracket. This protocol is an implied bid; the first speaker can accept it by returning the requested response at the end of the chain, or reject it by returning a negative response to the first RU of the chain (the RU carrying the BBI).

A bracket is terminated using the EBI, but the termination protocols can differ, so the protocol to be used is specified in the BIND parameters.

QUIESCE PROTOCOL

One session partner may inform the other to stop sending data and to enter quiesce state. Once in quiesce state, a half-session cannot send data RUs or normal-flow command RUs; it can send expedited command RUs and respond normally to RUs received from its session partner.

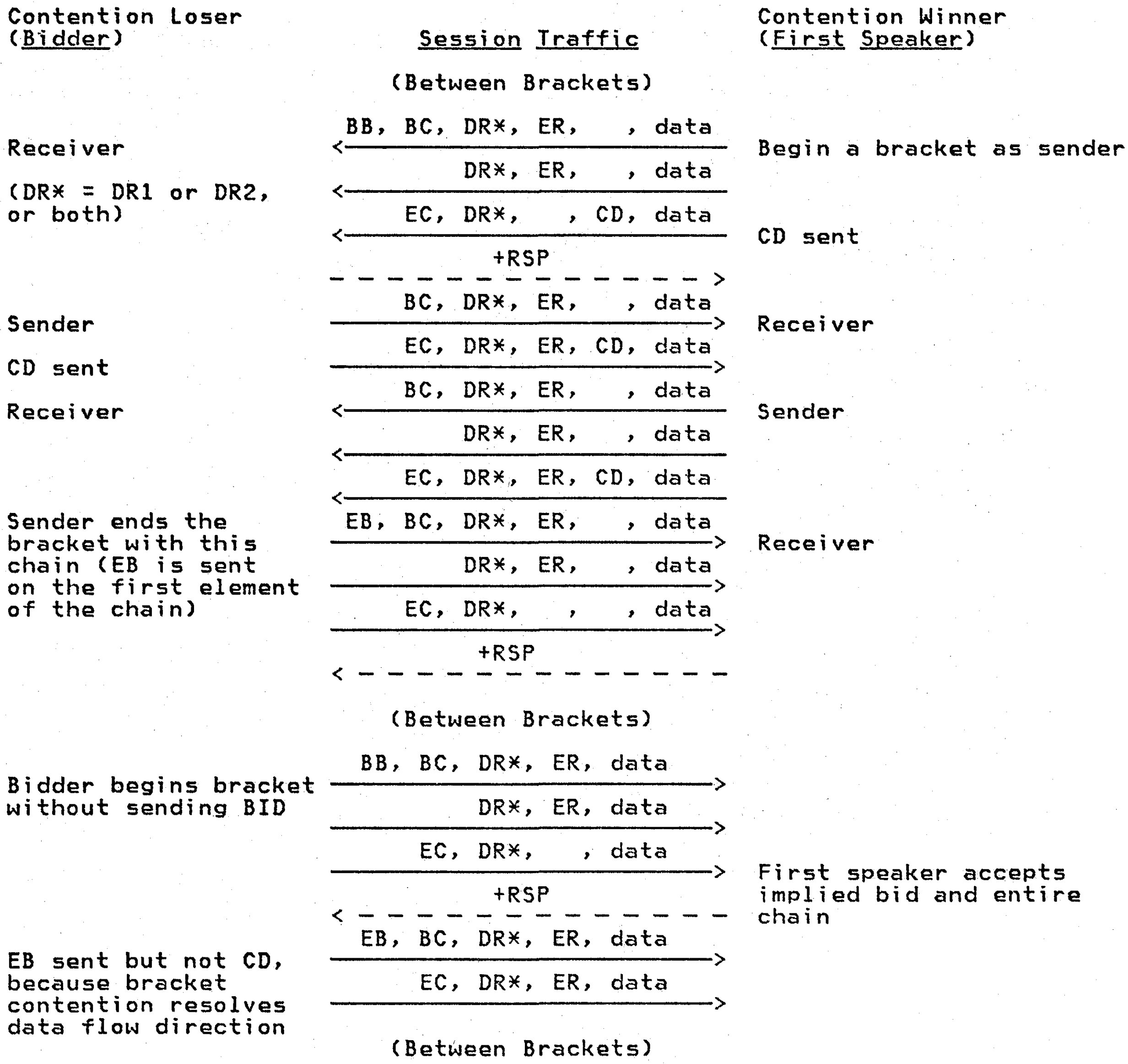


Figure 2-12 (Part 1 of 2). Communicating using brackets in HDX-FF.

You use the QEC (quiesce at end of chain) command to request that the session partner enter quiesce state. It is the command we were talking about in Chapter 1 when we used the expression "be quiet for a minute and I will get back to you." For example, you can use QEC to reduce the load on a component during peaks in the session activity. After the component completes the current chain, it returns a QC (quiesce complete) command and enters quiesce state.

An LU is released from quiesce state when a Release Quiesce (RELQ) command is received.

Figure 2-13 illustrates the flow of commands and data preceding and during quiesce protocol.

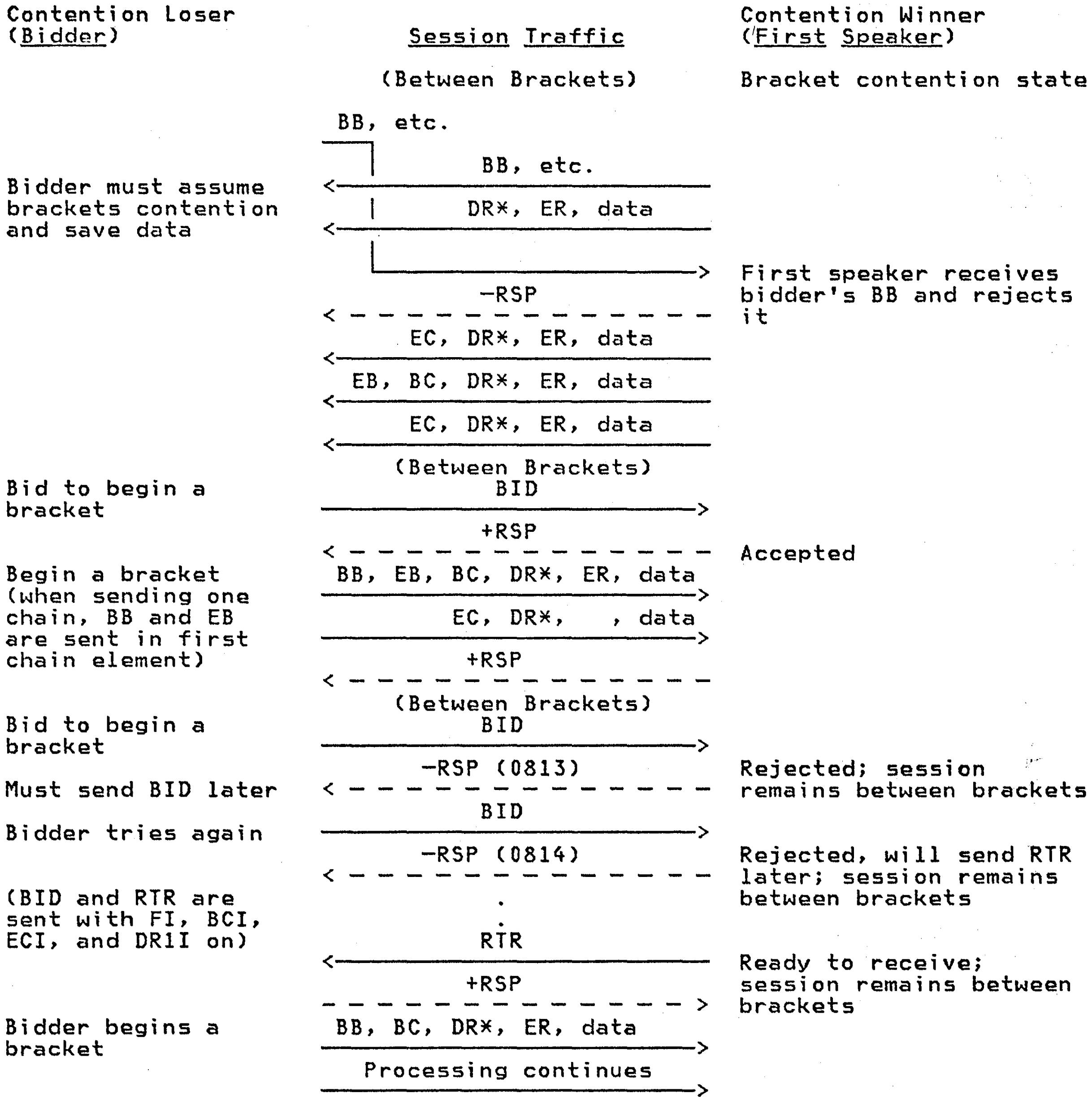


Figure 2-12 (Part 2 of 2). Communicating using brackets in HDX-FF.

REPORTING STATUS AND SIGNALING THE SESSION PARTNER

It may be necessary for one session partner to inform the other about local conditions that affect the session. A half-session may choose to do this using a data RU, or it can use specially defined command RUs.

The LUSTAT (LU status) command is one way to report this status. A status code in the command reports the condition of a system component or the status of a transmission. LUSTAT travels on the normal flow.

There will be times when an LU cannot use (or prefers not to use) the normal flow to send status information to the session partner. LUSTAT cannot be sent, for example, while the half-session is in quiesce state or while it is receiving normal-flow data. The LU can use the SIG (signal) command, however, because it sends SIG on the expedited flow. An example of a signal command is a request for permission to send data

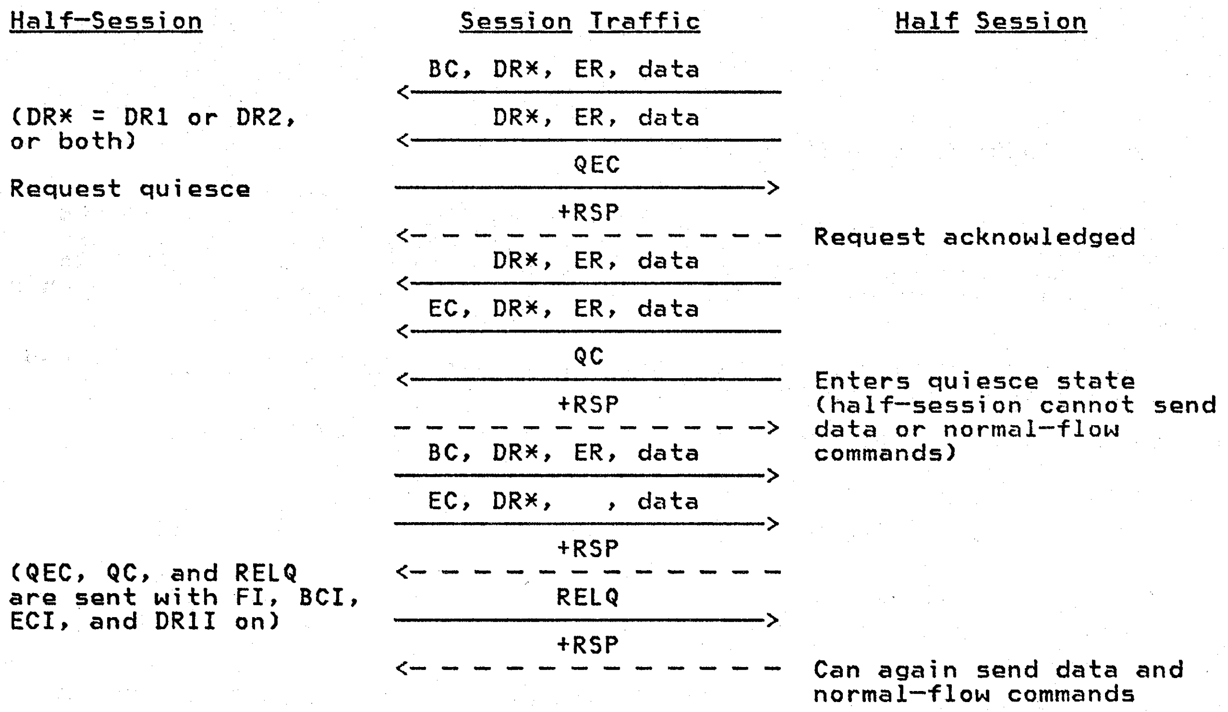


Figure 2-13. Quiesce protocols.

(requesting the change direction indicator). Signal and LU status codes are defined in SNA Reference Summary.

DATA HANDLING PROTOCOLS

You use data handling protocols for device control, data management, and data compression and compaction. The protocols change the way information is presented so as to match the needs or language of each end-user. If we use a printer as an example, these data handling protocols let you take data that is packaged within brackets, chains, and RUs, and transform it into the lines of data that are printed on the printer and the printer controls needed to position the data.

When the LU receives data from Path Control, it determines the destination of the data and selects the correct end-user process to send the data to. The end-user process then translates, forms, transforms, or combines the data as appropriate before it passes the data to the application. These data handling services, called presentation services, provide common formats and protocols to insulate one end-user process from unnecessary details of another's operation.

One way for you to express these protocols in LU-LU sessions is by including control information in the data stream. All LU-LU session types allow controls in the data stream, but this is the only way controls are passed in session types 2 and 3.

In the other session types (0, 1, 4, and 6), you can also send control information in FM headers, and, except for session type 6, in string control bytes (SCBs). Both are part of a data RU, as shown in Figure 2-1. Data RUs may contain neither, but many data RUs contain one or more FM headers and one or more SCBs.

USE OF FUNCTION MANAGEMENT (FM) HEADERS

There are parameters in BIND that indicate whether FM headers will be used during the session and, in some cases, whether there are limitations on their use. When they are used, the format indicator (FI) in the RH identifies whether the subsequent RU contains an FM header. Only data RUs contain FM headers.

One or more FM headers may be used. When used, they must be positioned immediately following the RH.

FM headers are discussed in Chapter 3 under "Types of Sessions."

DATA COMPRESSION AND DATA COMPACTION

Data compression involves the recognition of repeated characters. Instead of repeating the string of characters, a one- or two-byte code is substituted.

Data compaction involves restructuring portions of the data stream so that some bytes represent more than one character of data. The characters to be compacted are selected by the end-user.

By using data compression or compaction, you reduce significantly the number of characters transmitted in a data RU, thus increasing the information that can be handled by the data link. When you compact data, you use computation time to save transmission time on the data link. Compaction may improve overall throughput when the network's communication paths are overloaded or are slow-speed paths. Compaction may not be efficient when your data has few occurrences of character strings that can be compacted and when the network has high-speed or broadband communication paths.

If the FM header indicates that an SCB (string control byte) follows, the data following the FM header begins with an SCB. The SCB identifies the compression or compaction characteristics of the data, as discussed below. The SCB describes up to 63 bytes of compressed or compacted data; the data, however, must have the same compression or compaction characteristics. Additional SCBs are included in the data stream as its characteristics change or every 63 bytes, whichever occurs first.

The first two bits of the SCB identify the function to be performed; the last six give a count:

Function	Count
No characters transformed	Number of bytes between this SCB and the next one
Repeated prime character	Number of prime characters represented by this SCB (the next byte is another SCB)
Repeated next character	Number of times the next character is repeated (the next SCB follows the character)
Compaction code	Number of compacted bytes between this SCB and the next SCB

Once compression or compaction is selected, one of the end-users must send an FM header to establish the compression or compaction criteria. The end-user originating the data must compress or compact it and build the SCBs. The end-user receiving the data must analyze the SCBs and decompress or decompact the data.

ERROR RECOVERY

Errors are a naturally occurring phenomenon in the complex environment for which the SNA protocols are defined. They have a number of causes, ranging from random electrical noise to mismatched definitions for session partners. They can vary in persistency from transient to fully disruptive.

An error, as defined by SNA, is a violation of:

> A general architectural rule, such as using an undefined RU request code
>
> A session rule, such as using a FM protocol not defined within the FM profile or usage field of BIND
>
> A rule established by an FM header within a session
>
> A state-dependent rule, such as a sequence number received on the normal flow that was not one greater than the last

Either session partner can be responsible for recovering from errors and unsuccessful transmissions. BIND parameters indicate which partner has the responsibility, and in some cases this responsibility may be shared.

The sending LU has the responsibility to send an error-free data stream, and thus must check and return to the end-user any errors that are detected. Thus, many of the remaining errors are temporary. They are corrected either by the path control network or by an operator (for example, a printer being out of forms) and are not seen by the LU. For the temporary errors found by the LU's half-session, the half-session frequently can recover from the error by retransmitting the RU chain that failed.

One of the main recovery problems with LU-LU sessions is resynchronizing the session after an unrecoverable transmission error. If a half-session has little recovery capability, it may send to its session partner only CLEAR or UNBIND, where the transmission in error is deleted or the session is terminated. If a half-session has a capability to restart the transmission, it may use sequence numbers (shown in the TH in Figure 2-1) to recover the session.

Most half-sessions that have no session recovery responsibility begin recovery by sending an RQR (request recovery) command. The half-session could also request that it be shut down (RSHUTD) if it detected an error that clearly shouldn't have occurred.

The half-session with recovery responsibility may then respond by sending CLEAR to clear all session traffic, CANCEL to cancel and purge the current chain, or a data RU (without data) that specifies EC (end chain).

When session traffic has been cleared, the half-session with recovery responsibility may send STSN (set and test sequence numbers) and SDT (start data traffic), if it has this capability.

There may be times when the session cannot be resumed, but the end-user's job is not complete. If this occurs, the session can be concluded and a new session started where the previous session ended. Sequence numbers of the new session may be reset to the nonzero value of the previous session and the new session may be started.

We have shown, in the paragraphs above, only a few of the ways half-sessions can recover from in-session errors. The recovery approach a half-session uses depends on the type of error and its error recovery capability. That capability can range from issuing an UNBIND in order to terminate a session to performing sophisticated error correction (for example, by using STSN and SDT).

SESSION DEACTIVATION

You can conclude a session immediately or in an orderly manner. With most LU-LU session types, you conclude a session immediately by sending the UNBIND command from one half-session to its session partner. UNBIND is sent expedited and any normal-flow data traffic is ignored. On LU-LU sessions where UNBIND is not used, you can select one of the procedures in the next few paragraphs.

Sessions are concluded in an orderly manner when one session partner notifies the other that the session should be ended as soon as all pending data traffic has been sent. In many cases, a primary LU begins its deactivation procedure by sending a SHUTD (shutdown) command, while secondary LUs and LUs with equal status begin their procedure by sending a RSHUTD (request shutdown) command. The protocols associated with these commands are shown in Figures 2-14 and 2-15.

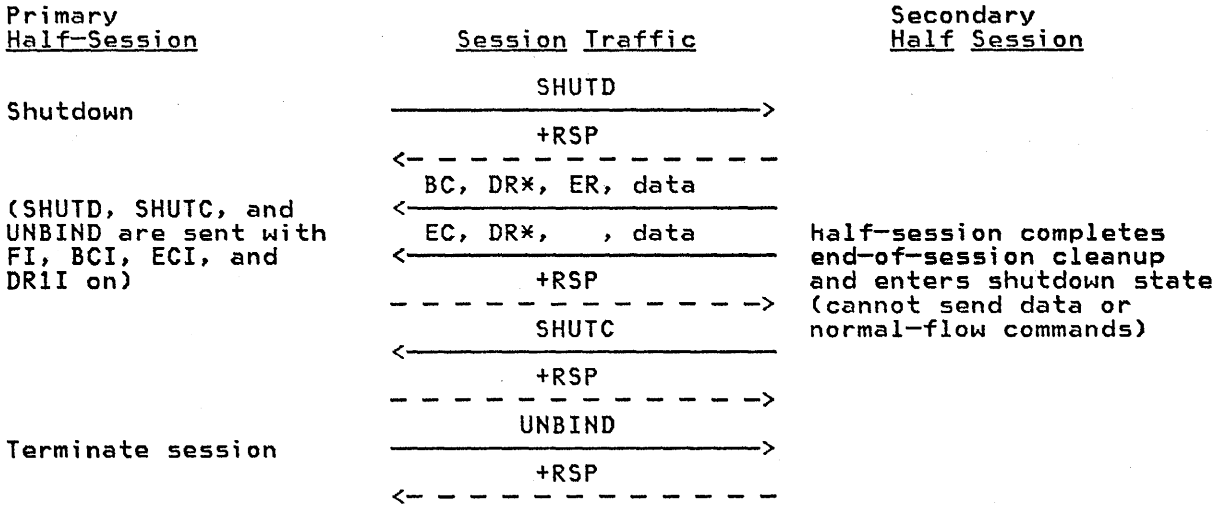


Figure 2-14. Orderly shutdown initiated by primary half-session.

Shutdown state is a form of quiesce state where a half-session cannot send data or normal-flow commands but can send expedited-flow commands and responses. As with quiesce, a half-session may send CHASE before SHUTC (shutdown complete) to ensure that all outstanding responses are returned before shutdown state is entered. Usually, shutdown state is entered only when a session is being ended.

Another approach to session termination is for a terminal operator to log off, causing a secondary LU to send a message to its SSCP on the SSCP-LU session requesting that the session be terminated. The SSCP advises the primary LU to send UNBIND.

Once again, these approaches to session deactivation are not the only approaches, but they are the most widely used approaches.

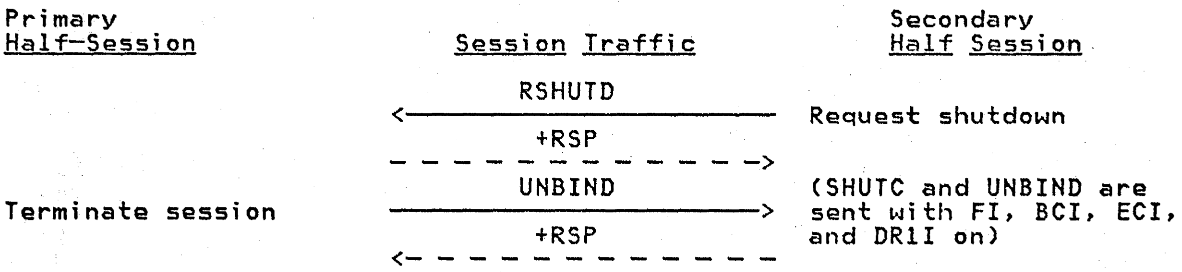


Figure 2-15. Orderly shutdown initiated by secondary half-session.

SUMMARY

In this chapter, we have shown the most widely used protocols for starting sessions, communicating between session partners, and terminating sessions. The information is a sampling only and is not complete, and there are protocols that augment the ones we have discussed. In a few cases, IBM products have deviated from a protocol because of a unique characteristic of the particular device.

For more detailed information on protocols, you should use *SNA Format and Protocol Reference Manual*, *Architecture Logic* and *SNA Logical Unit Types*. All protocols in the reference manual are common across LU-LU session types, while the protocols shown in *SNA Logical Unit Types* are by LU-LU session type.

CHAPTER 3: TYPES OF LU-LU SESSIONS

After reading about the three functional layers of SNA (in Chapter 1) and the variety of functions offered by the LU (in Chapter 2), you can see that a large number of combinations can be specified. To provide a structure for unifying all these situations, SNA defines physical unit types and logical unit session types. Each type of physical-unit and logical-unit session is, in effect, a grouping of compatible functions. For example, if the architecture contained ten LU functions, SNA could group functions 1, 2, 3, 6, and 8, identifying them as an LU-LU session type.

The major difference between physical unit types occurs in the transmission facilities (the way that path control sends information). These facilities are usually built into the products at design time. SNA's four physical unit types are discussed in Chapter 1.

In this chapter, we will discuss the way LU functions are grouped. Six session types have been defined for sessions between LUs (LU-LU sessions). Each product that contains an LU can support one or more LU-LU session types. The LU-LU session type is selected by the end-users (or preselected) when the session is established between them. The LU that is establishing the session takes this selected session type and converts it to a parameter in the BIND command.

Selecting the session type, however, is only one of the decisions that the LUs make when activating an LU-LU session. Within each session type, the LUs can choose variations of the functions, which can include or exclude some functions. Again, the LUs use the BIND command to specify which functions they will use.

THE BIND COMMAND

The format of the BIND command is shown in Figure 3-1. Each field is discussed in subsequent sections of this chapter, but not all field values are defined. If you need to know field values, see SNA Logical Unit Types.

PROFILES AND USAGE FIELDS

LU-LU sessions may not require all of the functional capability provided by a logical unit's components. Related sets of these functions are grouped together in what are called profiles.

Profiles are assigned numbers so that they can be specified in the BIND command that activates the LU-LU session. Both half-sessions must use the same number (Figures 3-2, 3-3, and 3-4).

In addition to profiles, the BIND command contains usage fields for Presentation Services (PS), Data Flow Control (DFC), and Transmission Control (TC). The usage fields further qualify PS, DFC, and TC functions that are not part of the profiles.

Figure 3-2 shows examples of functions that are selected using the profile and usage fields of the BIND command. Figure 3-4 shows the profiles that are valid for each LU-LU session type.

Byte	Field Description
0	Identifies this RU as a BIND command
1	Negotiable or non-negotiable BIND
2	FM (function management) profile
3	TS (transmission subsystem) profile
	4-7 FM usage: Primary half-session protocols Secondary half-session protocols Common protocols
	8-13 TS usage: Pacing Maximum RU (request unit) size
14	PS (presentation services) profile (identifies the type of LU-LU session)
15-25	PS Usage: Primary half-session usage FM headers Data streams Secondary half-session usage FM headers Data streams (For session types 2 and 3, the PS Usage field describes screen sizes and buffer sizes, respectively.)
26-27	Cryptography
28-n	Names and end-user data

Figure 3-1. Format of the BIND command.

Half-Session Component	Selectable Functions (not a complete list)	BIND Fields Used
Presentation Services (PS)	• SNA Character String Usage • Code Repertoire (EBCDIC, ASCII, other) • Attended or Unattended Mode • FM Header Usage	PS Profile and Usage
Data Flow Control (DFC)	• Request and Response Modes • Half-Duplex or Full-Duplex Send-Receive Mode • Brackets and Chaining Rules • Data Flow Control Techniques and Requests	FM Profile and Usage
Transmission Control (TC)	• RU Size • Pacing • Session Control Requests • Cryptography Options	TS Profile and Usage

Figure 3-2. Each LU has three half-session components that support the end-user. The profile and usage parameters of the BIND command are used to define the half-session functions desired.

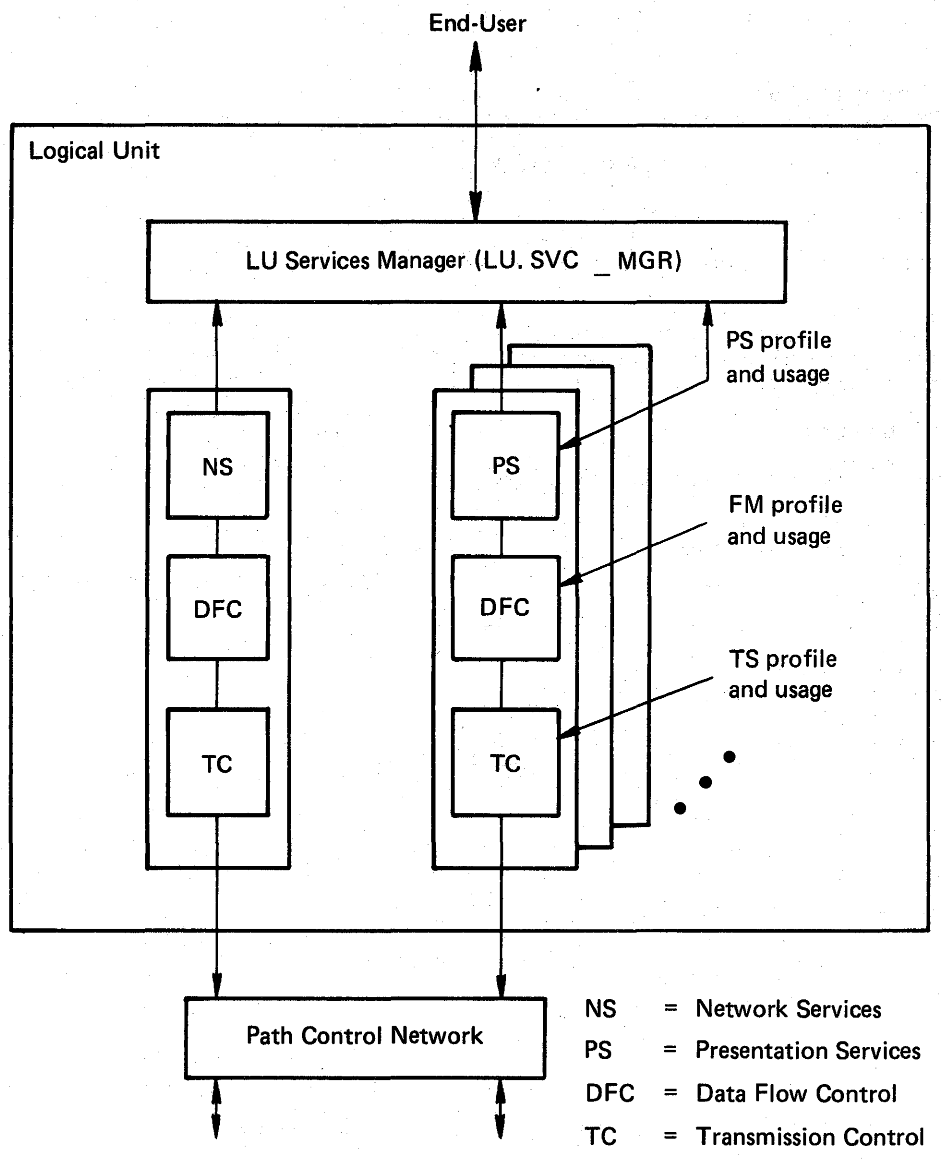


Figure 3-3. Each LU has three half-session components. (Figure 3-2 shows how profile and usage parameters affect these components.)

PS Profile and Usage

The PS profile field contains the LU-LU session type. The PS usage field describes the use of FM headers and defines the data streams for the session.

FM Profile and Usage

FM profiles 2, 3, 4, 7, and 18 are permitted with LU-LU sessions. All are described below except for profile 2, which is used only with session type 0 and is described in SNA Reference Summary. See Figure 3-2 for key usage options.

FM Profile 3: Profile 3 specifies the following session rules:

Both half-sessions use immediate response mode.
Both half-sessions support these commands:

CANCEL
SIG
LUSTAT (allowed secondary-to-primary only)
CHASE
SHUTD
SHUTC
RSHUTD
BID and RTR (allowed only if brackets are used)

FM Profile 4: Profile 4 specifies the following session rules:

Both half-sessions use immediate response mode.
Both half-sessions support these commands:

CANCEL
SIG
LUSTAT
QEC
QC
RELQ
SHUTD
SHUTC
RSHUTD
CHASE
BID and RTR (allowed only if brackets are used)

FM Profile 7: Profile 7 specifies the following rules:

Both half-sessions use immediate response mode.
Both half-sessions support these commands:

CANCEL
SIG
LUSTAT
RSHUTD

FM Profile 18: Profile 18 specifies the following session rules:

Both half-sessions use immediate response mode.
Both half-sessions support these commands:

CANCEL
SIG
LUSTAT
BIS and SBI (allowed only if brackets are used)
RSHUTD
CHASE
BID and RTR (allowed only if brackets are used)

FM Usage: This book does not discuss the FM usage values that you select in BIND. You will find usage values for each session type in SNA Logical Unit Types, and a detailed discussion of FM profiles and usage options in SNA Format and Protocol Reference Manual, Architecture Logic.

TS Profile and Usage

TS profiles 2, 3, 4, and 7 are permitted with LU-LU sessions. All are described below except for profile 2, which is used only with session type 0 and is described in SNA Reference Summary. See Figure 3-2 for key usage options.

TS Profile 3: Profile 3 specifies the following session rules:

Normal flow is paced in both directions.
Sequence numbers are used on the normal flow.
CLEAR and SDT are supported.
RQR and STSN are not supported.
CRV is supported when session-level cryptography is selected.

TS Profile 4: Profile 4 specifies the following session rules:

Normal flow is paced in both directions.
Sequence numbers are used on the normal flow.
SDT, CLEAR, RQR, and STSN are supported.
CRV is supported when session-level cryptography is selected.

TS Profile 7: Profile 7 specifies the following session rules:

Normal flow is paced in both directions.
Sequence numbers are used on the normal flow.
SDT, CLEAR, RQR, and STSN are not supported.
CRV is supported when session-level cryptography is selected.

TS Usage: The TS usage subfields define pacing counts and the maximum RU size on the normal flow. You will find usage values for each session type in *SNA Logical Unit Types*, and a detailed discussion of FM profiles and usage options in *SNA Format and Protocol Reference Manual, Architecture Logic*.

PS Profile (LU Type)	TS Profile	FM Profile	PS Characteristics
0	2,3,4	2,3,4	Any option desired
1	3,4	3,4	SNA character string FM headers (none, or one or more of FMH-1, FMH-2, FMH-3) Data processing media support
2	3	3	3270 data stream No FM headers Display support
3	3	3	3270 data stream No FM headers Printer support
4	7	7	SNA character string FM headers (none, or one or more of FMH-1, FMH-2, FMH-3) Data processing and word processing media support
5			Reserved
6	4	18	SNA character string or field format FM headers (FMH-4 through FMH-8) Program-to-program support for data bases, files, queues, and programs

Figure 3-4. Types of LU-LU sessions and the allowable subsets that can be specified with the TS (Transmission Subsystem), FM (Function Management), and PS (Presentation Services) profiles. Data processing media (devices) recognize different sets of SNA string controls than word processing media.

TYPES OF SESSIONS

The type number that is assigned to an LU-LU session identifies the combination of profiles and usage values that may be used. A given LU supports only sessions that draw from its allowable PS, FM, and TS profiles. A product is thus characterized by the LU-LU session type (or types) that it supports.

The LU-LU session types defined by SNA are shown in Figure 3-5 and are discussed below. A representative sample of IBM products is included in the figure so that you can relate architecture with implementation. For a given LU-LU session type, not all PLU products can have sessions with all SLU products. See your IBM representative to ensure that a desired combination is valid.

	LU-LU Session Type					
	0		1		2	
	PLU	SLU	PLU	SLU	PLU	SLU
Representative Product Set	IMS/VS CICS/VS	3270 3600 3650 3660 3790 S/34 Series/1	IMS/VS CICS/VS RES JES2 JES3 POWER/VS TSO NOSP NCCF VSPC TCAM 3630 DPPX	DPCX DPPX 3270 3630 3767 3770 3790 S/32 S/34 S/38	IMS/VS CICS/VS NOSP NCCF TSO VSPC TCAM	DPCX DPPX 3270 3790
Data Stream	Product Defined		SCS		3270	
FM Header Set	Product Defined		None, FMH 1,2,3		None	
FM Profile	2,3,4		3,4		3	
TS Profile	2,3,4		3,4		3	

Figure 3-5 (part 1 of 2). Summary of LU-LU session types and representative IBM products.

	LU-LU Session Type					
	3		4		6	
	PLU	SLU	PLU	SLU	PLU	SLU
Representative Product Set	CICS/VS TCAM	DPCX 3270 3790	IMS/VS CICS/VS RES S/34 S/38	5250 6670 S/34	IMS/VS CICS/VS	IMS/VS CICS/VS
Data Stream	3270		SCS		SCS, Field Formatted	
FM Header Set	None		None, FMH 1,2,3		FMH 4,5,6,7,8	
FM Profile	3		7		18	
TS Profile	3		7		4	

Figure 3-5 (part 2 of 2). Summary of LU-LU session types and representative IBM products.

Type 0 LU-LU Sessions

Type 0 sessions are for LU-LU communications that are implementation-dependent and do not fall within the "groupings" of profiles defined by SNA. Type 0 sessions allow any FM and TS profile shown in Figure 3-4 and any FM and TS usage protocols associated with these profiles. The format of the PS usage field in BIND is defined by the users of the type 0 session, but the remaining BIND fields must be specified as defined in SNA Format and Protocol Reference Manual, Architecture Logic. The content of the PS usage field, however, must specify formats and protocols defined by SNA.

Type 0 LU-LU sessions use the path control network and must conform to network protocols.

In addition to those IBM 3270 Information Display System products shown in Figure 3-5, the IBM 3270 uses type 0 sessions for Models 11 and 12 of the 3271 and 3275. All other 3270 products use type 1, type 2, or type 3 sessions, or a combination thereof.

Type 1 LU-LU Sessions

Type 1 sessions are used for data communications between application programs and single- or multiple-device terminals. The application programs are associated with the primary half-session; the terminals are associated with the secondary half-session.

The terminal's devices may include consoles, printers, diskettes, disks, and card units. The devices generally are not directly visible to the primary half-session because terminal implementations are allowed to provide microprogramming to manage devices and data locally (or permit user-written programs to do so). Terminals may be attended or unattended. The environment is either interactive or remote job entry.

A terminal's processing capability may range from minimal to an amount sufficient to provide for system and user programming. Depending on the capability of terminals, end-users have great flexibility with the way functions are distributed between them. As a terminal's computational capabilities increase, there is more opportunity to distribute function outboard of the "host" application program.

FM profile 3, TS profile 3, the 48-character graphic set, the new-line and forms-feed SNA character set (SCS) controls, and a console are the mandatory functions of type 1 sessions. FM headers, compression and compaction, the remaining SCS controls, and the other profiles shown in Figure 3-4 are optional.

The BIND Command: The BIND command is sent by the primary LU. If the secondary LU's send and receive capability is comparable, the secondary LU returns a positive response; otherwise, it must return a negative response.

The primary LU controls the session and the secondary LU conforms to the primary LU's requirements. The primary LU has error recovery responsibility for the session.

FM Headers: Type 1 sessions may use function management headers (FM headers or FMHs) to select a destination and then control the way the data is sent to the destination. FM headers also may be used to control the way the data is presented. Three types of FM headers are valid if the session partners choose to use them:

FMH-1 Selects a destination (medium on which data is presented). A BEGIN FMH-1 selects a destination. Data is then sent to that destination. When all data has been sent and received, the session partner that sent the BEGIN FMH-1 sends an END FMH-1.

If a priority message or job must interrupt the flow of data between the session partners, the session partner that sent the BEGIN FMH-1 can send a SUSPEND FMH-1 to suspend data traffic with its current destination, called the active destination. It then sends a BEGIN FMH-1 to the new destination, which becomes the active destination. When it completes its priority traffic, it sends an END FMH-1 to the active destination. (When the transaction can occur within one chain, the transaction may be completed with a BEGIN/END FMH-1 rather than the BEGIN-data-END sequence, which requires a minimum of two chains.)

To resume the suspended data traffic, the session partner sends a RESUME FMH-1 to the original destination, completes its data traffic, and sends an END FMH-1.

If a session partner is sending data to an active destination but wishes to change a parameter such as the record length, it sends a CONTINUE FMH-1. If a session partner wishes to terminate the destination immediately, it sends an END-ABORT FMH-1.

If a session partner sends data without previously sending an FMH-1, the data goes to the console.

FMH-2 Specifies the data management activities to be performed at the destination selected by the FMH-1. Their use is optional. The functions that can be performed by FMH-2s are shown in Figure 3-6.

As can be seen in Figure 3-6, more than one FMH-2 can be used to perform an operation. One header can define the operation to be performed, while another provides a password or tells where to perform the operation. The FMH-2s that define the operation are the root FMH-2s. The ones that provide the password and identify where the operation is to be performed are the extension FMH-2s.

FMH-3 Carries information that relates to all destinations of both session partners. FMH-3s can send a prime compression character or a compaction table, or send a query to obtain a compaction table. They also can send status information that helps correlate related FM header operations.

For more information on FM headers, see SNA Logical Unit Types.

Operation Type	FMH-1	Root FMH-2	Extension FMH-2
Destination Selection	Begin		
	Begin/End		
	Resume		
Volume Selection	Begin		Volume ID, Password
	Begin/End		Volume ID, Password
Data Set	(Note)	Create Data Set	Password
		Scratch Data Set	Password
		Scratch All Data Sets	Password
		Erase Data Set Records	Password
Record	(Note)	Add	Password
		Add Replicate	Password
		Replace	Record ID, Password
		Replace Replicate	Record ID, Password
		Erase	Record ID, Password
Compression and Compaction	(Note)	Compaction Table	
		Prime Compression Character	
Operation Information	(Note)	Peripheral Data Information Record	
Status	(Note)	Query for a Data Set	Password
		Note	
		Note Reply	
Scheduling	(Note)	Execute Program Offline	
Record Identification	(Note)		Record ID, Password

Note: A destination must be active; that is, a BEGIN, BEGIN/END, CONTINUE, or RESUME FMH-1 must have been the last FMH-1 issued. If no destination is active, the transmission goes to the console.

Figure 3-6. Data management is accomplished using function management headers (FMHs). FMH-1 is used to select a destination; optionally, FMH-2s are used to process data at that destination.

Type 2 LU-LU Sessions

Type 2 sessions are used for data communications between an application program and a single display device using the 3270 data stream. The application program is associated with the primary half-session and the display is associated with the secondary half-session. Type 2 sessions allow users to migrate existing customer-written 3270 application programs into an SNA network.

The logical device within the secondary half-session is visible to the primary half-session. The physical device that is represented by the logical device remains allocated for the duration of the unit of work, for example, a bracket or a session. The physical device also may be dedicated permanently.

FM profile 3, TS profile 3, and the SNA 3270 data stream are the mandatory functions of type 2 LU-LU sessions. FM headers and the compaction and decompaction of data are not allowed.

The following options are allowed:

1. Selector Pen.
2. Screen Size. A 480- or 1920-character screen size may be selected. When the screen is physically larger than the size specified in BIND, a wraparound occurs at the specified size. A wraparound is

the continuation of an operation from the maximum addressable location in storage to the first addressable location.

3. Magnetic Stripe Reader. The numeric and alphameric character sets are supported.

The SNA 3270 data stream may be extended. The following functions are options when implementations support an extended data stream:

1. Screen Size. Screen sizes other than 480 and 1920 characters are supported. When the screen is physically larger than the size specified in BIND, a wraparound occurs at the specified size. The secondary half-session may support screen sizes up to 65,536 characters.

2. Screen Size Switching. A secondary half-session may provide dynamic screen size switching. The screen size may be switched between two values (specified in BIND) by the Erase/Write and the Erase/Write Alternate commands. These commands are part of the SNA 3270 data stream. The Erase/Write command selects the default size while the Erase/Write Alternate command selects the alternate size. The screen size is initialized to the default size.

3. Partitions. The screen can be partitioned and a partition can be controlled separately from other partitions.

4. Scrolling. A partition can be scrolled left, right, up or down.

5. Color. A color can be selected for a character or field.

6. Programmed Symbols. In addition to using pre-defined characters, you may define and use other characters and symbols.

7. Highlighting. A character or field can be highlighted (emphasized) by intensifying, underlining, blinking, or reversing the image. Formerly, highlighting allowed only intensifying the image.

Since type 2 sessions are most commonly associated with a keyboard/display station, they have some characteristics that do not apply to other types of LU-LU session. These characteristics are described below.

Device Sharing: Device sharing is the use of a device (or medium) by more than one session. The SLU allows two sharing configurations:

Printer shared with direct-print sessions (type 1 or type 3 sessions) or other type 2 sessions

Display shared between the SSCP-SLU and PLU-SLU sessions

The contents of a screen can be copied to a printer. When a printer is shared between sessions, the allocation and scheduling of the printer are not SLU or PLU functions. When a display is shared between sessions, concurrent sessions can be held between the SLU and a PLU, and between the SLU and its SSCP. The scheduling of messages to the display is an SLU function. The sharing of the display device is not directly visible to the primary half-session.

Display Printout (Copy): The pre-SNA 3270 Copy command is not supported by the secondary half-session, but the primary half-session can initiate a printout of the screen at the secondary half-session by setting an indicator in the data stream. To provide a hard copy capability, the secondary half-session must have a printer device available in addition to the display device. The printer could be permanently allocated to one secondary half-session, or shared among the secondary half-sessions of several SLUs. The printer that is shared can also be used by secondary half-sessions of other session types.

The operator-initiated printout is handled by secondary half-session access to the shared printer logic. No interaction with the primary half-session is required or desired at the time of initiation.

<u>Half-Duplex</u> <u>Operations</u>: Type 2 LU-LU sessions use half-duplex flip-flop (HDX-FF) operation. By using HDX-FF, the primary half-session has control/awareness of the display screen.

Type 2 LU-LU sessions use the concept of a formatted screen to reduce the amount of data that must be transferred between the display station and the primary half-session. When the screen is formatted, only those fields defined as "modified" are transferred to the primary half-session in reply to a read-modified operation. For traffic to the display, the primary half-session may keep track of the format of the display and thus reduce the amount of data that must be transferred to the display to alter the image.

Type 3 LU-LU Sessions

Type 3 sessions are used for data communications between an application program and a single printer (without a keyboard) using the 3270 data stream. The application program is associated with the primary half-session and the printer is associated with the secondary half-session. Type 3 sessions allow users to migrate existing customer-written 3270 application programs into an SNA network.

The logical device within the secondary half-session is visible to the primary half-session. The physical device that is represented by the logical device remains allocated for the duration of the unit of work, for example, a bracket or a session. The physical device also may be dedicated permanently.

FM profile 3, TS profile 3, and the SNA 3270 data stream are the mandatory functions of type 3 LU-LU sessions. FM headers and the compaction and decompaction of data are not allowed.

One option is allowed with the 3270 data stream: the buffer size of the secondary half-session can be 480 or 1920 characters. When the buffer is physically larger than the size specified in BIND, a wraparound occurs at the specified size.

The SNA 3270 data stream may be extended. The following functions are options when implementations support an extended data stream:

1. Buffer Size. Buffer sizes other than 480 and 1920 characters are supported. When the buffer is physically larger than the size specified in BIND, a wraparound occurs at the specified size. The secondary half-session may support buffer sizes up to 65,536 characters.

2. Buffer Size Switching. A secondary half-session may provide dynamic buffer size switching. The buffer size may be switched between two values (specified in BIND) by the Erase/Write and the Erase/Write Alternate commands. These commands are part of the SNA 3270 data stream. The Erase/Write command selects the default size while the Erase/Write Alternate command selects the alternate size. The buffer size is initialized to the default size.

The profiles that define type 3 LU-LU sessions differ slightly from those that define session types 1 and 2. These differences are discussed below.

<u>Device</u> <u>Sharing</u>: In addition to providing a dedicated printer configuration, the secondary half-session allows sharing of the printer for local use, that is, for display printouts for type 2 LU-LU sessions as well as for dedicated type 1 and type 3 sessions. The printouts for type 2 sessions may be on a between-session basis or between-bracket basis.

The allocation and scheduling of the shared device(s) is not apparent to the primary half-session.

When a printer is configured for sharing between brackets, the primary half-session must assume that both the print buffer and print element positioning have changed between brackets. The primary half-session is not informed when the printer is used. The same applies to the between-session interval when sharing between sessions is implemented.

Chaining: The secondary half-session supports multiple element chains. It does not send FM data chains.

Only one 3270 command is allowed per chain. The command must be the first byte of the chain.

Half-Duplex Protocols: The secondary half-session operates only in half-duplex flip-flop (HDX-FF) mode. It returns the flow immediately whenever the primary half-session sends CD with any request.

Reading the Buffer: A type 3 printer accepts the same data stream commands as a type 2 display except that the Read, Read Modified, and Read Modified All commands are not accepted.

Type 4 LU-LU Sessions

Type 4 sessions are used for data communications between two terminals or between application programs and single- or multiple-device terminals. Type 4 sessions are similar to type 1 sessions. Many of the same data processing applications that execute using type 1 sessions can execute using type 4 sessions. In addition, word processing applications can execute using type 4 sessions.

Type 4 sessions are used to manage input/output devices such as printers, card readers, and various storage devices. An operator console is often present for further input/output in an interactive mode or for diagnostic messages. In addition, some type 4 implementations perform data management functions such as storing data for retrieval and modification by the sender. Application programs and host subsystems may also use type 4 protocols to communicate with similar components in other LUs in their network.

Type 4 sessions do not require a network with a separate SSCP if the session partners handle required SSCP functions. For example, each terminal in a two-terminal network could be capable of handling the minimal resource management requirements of the network.

In networks with SSCPs, the primary half-session must have an SSCP-PLU session and the secondary half-session must have an SSCP-SLU session.

FM profile 7 and TS profile 7 are the mandatory functions of type 4 sessions. If a console or printer is supported, the 48-character graphic set is required also. FM headers, SNA character set (SCS) controls, compression and compaction are optional.

Data Stream Profiles: Type 4 sessions may use the data processing or word processing SCS controls. Sets of these controls, called data stream profiles, are defined so that type 4 implementations can limit the number of SCS controls that must be handled by a half-session. Each half-session selects the data stream profile when the destination is selected (see FMH-1 under "FM Headers" below).

Relationship between Session Partners: LU type 4 sessions may have session partners that are equals or ones that use a PLU-SLU relationship. When partners are equals, the relationship between half-sessions is symmetrical; that is, the LU at either end of the session is of equal standing (a peer) with the LU at the other end. In PLU-SLU sessions, the relationship is asymmetrical, and the primary LU "controls" the session.

In a symmetrical relationship, the sender of data is responsible for recovery from errors in the data. In an asymmetrical relationship, the primary half-session is responsible for all error recovery.

When partners are equals, the terms primary half-session and secondary half-session no longer retain their traditional meaning. To preserve the terminology used by other session types, the half-session that sends

BIND is the primary and the half-session that receives BIND is the secondary. These terms are used only in the description of the BIND/UNBIND procedure because, other than during these procedures, either half-session may start any sequence of activity. In most discussions, the session partners are identified as the sender of BIND and the receiver of BIND.

BIND Command Parameters: There are two types of BIND -- non-negotiable and negotiable. With the non-negotiable BIND, the BIND sender establishes the session parameters without compromise on potentially negotiable characteristics such as maximum RU size.

With the negotiable BIND, the BIND sender proposes a set of session parameters. The BIND receiver adapts where possible and returns a set of response parameters, which establish the negotiated limits for exchanges in both directions. The sender of a negotiable BIND, upon receipt of the BIND response, checks the returned parameters and accepts the session as negotiated or sends UNBIND to the session partner.

Reviewing it another way, the BIND sender includes its receive capability and a proposal for its send protocol in the BIND request. The BIND receiver examines the BIND request and either accepts the session parameters by returning a positive response with the same session parameters, or "negotiates" the session parameters by returning a positive response with different session parameters. This process is illustrated as follows:

```
                           BIND
   BIND Sender    -----------------> BIND Receiver

                      +RSP, parameters
   BIND Response <------------------ BIND Response
   Receiver                          Sender
```

Although other capabilities apply (such as security and user information), the key to establishing a session is the exchange of receive capabilities. Once an LU-LU session is established, an LU must not exceed the receive capability of the session partner -- so the BIND process must ensure that there is a send-receive match.

FM Headers: Type 4 sessions may use function management headers (FM headers or FMHs) to select a destination and then control the way the data is sent to the destination. FM headers also may be used to control the way the data is presented. Three types of FMHs are valid if the session partners choose to use headers:

FMH-1	Selects a destination (medium on which data is presented) and a data stream profile, and delimits a unit of work. See "FM Headers" under "Type 1 LU-LU Sessions" above for a description of FMH-1.
FMH-2	Specifies the data management activities to be performed at the destination selected by the FMH-1. Their use is optional. The functions that can be performed by FMH-2s are shown in Figure 3-6.
FMH-3	Carries information that relates to all destinations of both session partners. FMH-3s can send a prime compression character or a compaction table, or send a query to obtain a compaction table. They also can send status information that helps correlate related FM header operations.

For more information on FM headers, see *SNA Logical Unit Types*.

Type 6 LU-LU Sessions

Type 6 sessions are used for data communication between application subsystems. Application subsystems create an environment wherein programs in each subsystem communicate to accomplish a function distributed across these subsystems. For example, two CICS subsystems, two IMS subsystems, or one CICS and one IMS subsystem could be in session; a program within either subsystem or an application program supported by either subsystem may communicate with another program. The subsystems are the session partners -- not the programs. The session partners, therefore, must support a multiplexing of logical flows (parallel sessions) from any of their programs to any of their session partner's programs.

Some subsystem implementations use the term transactions for this type of communication between programs. It is a meaningful term because it connotes a short-duration exchange; the longer batch and interactive sessions are not precluded, however.

Within type 6 sessions, many transactions may occur on the session between application subsystems. In this sense, a transaction is a unit of work initiated by a single request. Where the work is accomplished is determined by where the end-user placed the programs within the network.

Any transaction routing that must be accomplished is the subsystem's responsibility. With the subsystem doing the routing, the location of the data (local or remote) becomes transparent. Programs can obtain access to remote data without issuing a unique transaction to the target system to access the data.

The programs in the application subsystems provide the required end-user functions. Terminal operators or application programs request these end-user functions. These requests are converted to a standardized format and forwarded to the appropriate subsystem following the session's protocol rules.

FM profile 18, TS profile 4, the 48-character graphic set, and FM header type 5 (ATTACH) are the mandatory functions of type 6 LU-LU sessions.

One of the data streams defined for use on type 6 sessions is the SNA field-formatted data stream. It contains user data along with information concerning the structure and disposition of the data. The data stream is a linear string of fields of fixed or variable length. The fields are described as to their type (for example, binary or decimal), layout (for example, right- or left-justified), length, data characteristics, and other attributes. Because two programs are communicating, the data stream can contain a wider range of data and control capability than the data streams for session types 1 through 4.

BIND Command Parameters: Type 6 sessions use both non-negotiable and negotiable BIND. The negotiable BIND is used primarily when the session partner's capabilities or session requirements are not known or when an application subsystem wants to change the session parameters. The negotiation protocol is similar to that described for LU-LU session type 4 (see "BIND Command Parameters" under "Type 4 LU-LU Sessions" above).

FM Headers: Type 6 sessions use FMH-4 through FMH-8. Except for FMH-5, their use is optional:

FMH-4 is the header for a transmission record, which is the logical unit of data interchange between processes. (Processes are programs and are defined under "Processes" below.) A chain may contain only one transmission record.

FMH-5 is used by a process in one half-session to select (attach) a process in the session partner. An FMH-5 can be followed by an FMH-4, an FMH-6, data, or any combination of the three. The change direction (CD) and end bracket (EB) indicators can be sent with an FMH-5.

FMH-6 is used by currently active processes to send requests (commands) to other currently active processes.

FMH-7 is used by currently active processes to provide error information after certain negative responses.

FMH-8 is used by currently active processes to send commands, parameters, and other information about messages flowing between processes.

Processes: A process is a systematic sequence of operations to produce a specified result. Thought of another way, it is a program or a piece of a program (along with its associated resources) to accomplish that result. In a type 6 session, processes are the users of the half-session.

The half-session and the process are linked using an FMH-5. The link remains for the duration of the transaction.