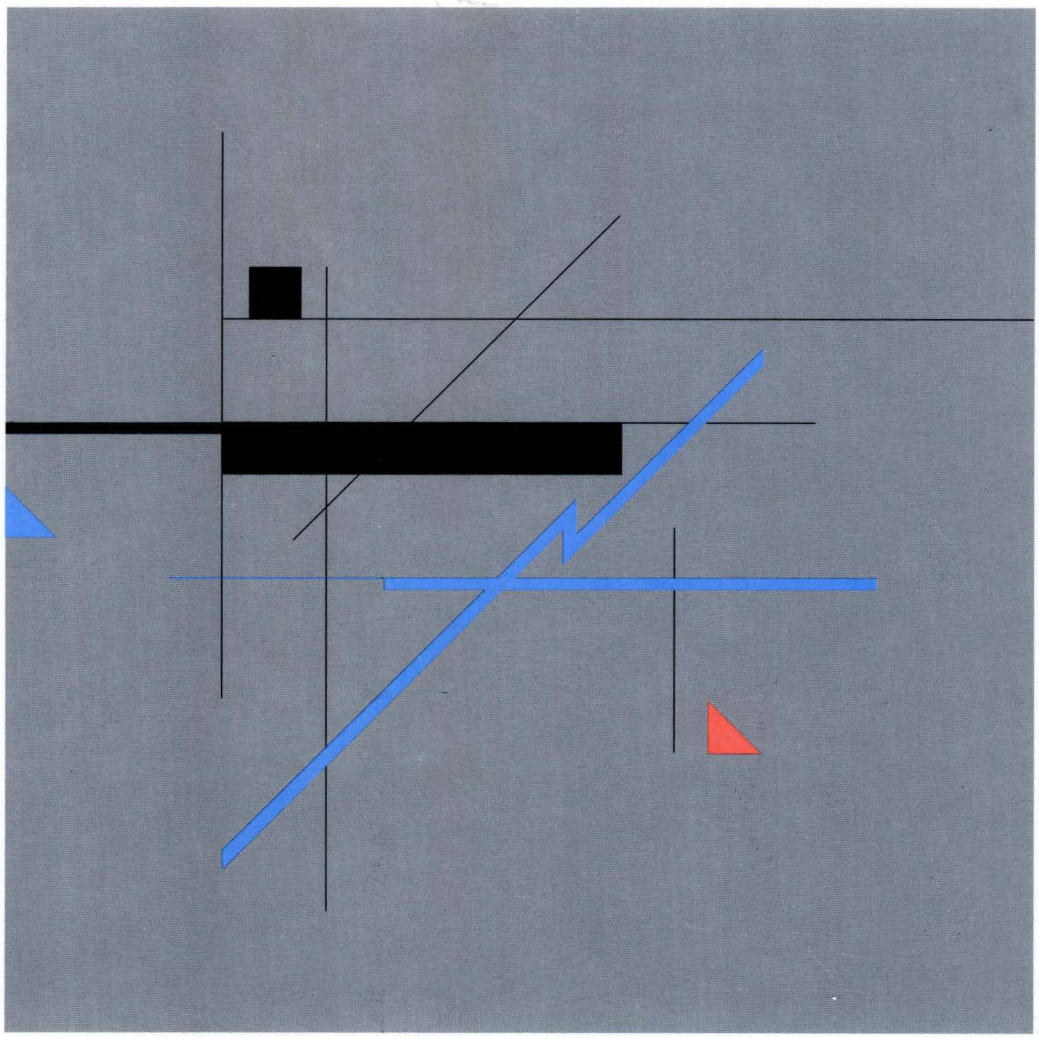
Systems Network Architecture
Technical Overview

GC30-3073-04



Systems Network Architecture
Technical Overview

GC30-3073-04

Note

Before using this document, read the general information under "Notices" on page xiii.

Fifth Edition (January 1994)

This major revision replaces GC30-3073-03, which is now obsolete.

Publications are not stocked at the address given below. If you want more IBM publications, ask your IBM representative or write to the IBM branch office serving your locality.

A form for your comments is provided at the back of this document. If the form has been removed, you may address comments to:

IBM Corporation
Department E15
P.O. Box 12195
Research Triangle Park, North Carolina 27709
U.S.A.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Contents

Figures

Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program or service. Evaluation and verification of operation in conjunction with other products, programs, or services, except those expressly designated by IBM, are the user's responsibility.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, 208 Harbor Drive, Stamford, Connecticut, United States 06904.

This document is not intended for production use and is furnished as is without any warranty of any kind, and all warranties are hereby disclaimed including the warranties of merchantability and fitness for a particular purpose.

Trademarks

The following terms, denoted by an asterisk (*) on first usage in the text of this publication, are trademarks of the IBM Corporation in the United States or other countries or both:

Advanced Peer-to-Peer Networking
AIX
Application System/400
APPN
AS/400
Bookmanager
DisplayWrite
Enterprise System/9370
Enterprise Systems Architecture/370
Enterprise Systems Connection
ESA/370
ES/9000
ES/9370
ESCON
IBM
IBM Library Reader
MVS/ESA
NetView
Operating System/2
Operating System/400
OS/2
OS/400
Personal System/2
PS/2
QMF
RISC System/6000
SAA
Series/1
System/88
Systems Application Architecture
VM/XA
VTAM

About This Book

This book provides a technical overview of IBM's Systems Network Architecture (SNA). IBM hardware and software products implement the SNA functions that enable network users to be independent of the network's characteristics and operation. This book explains what those SNA functions are and how they provide for communication between network users.

Who Should Read This Book

The audience for this book includes managerial and technical staff members whose responsibilities include selection, definition, or installation of SNA networks.

How This Book Is Organized

Chapter 1, "Introduction" provides an overview of SNA, including its architectural objectives, network components, data transport services, transaction services, and network management services.

Chapter 2, "Link and Node Components" provides information on link and node components, the major functions they perform, and the SNA layers in which they reside.

Chapter 3, "Data Formats" discusses message-unit formats, control fields, and data streams defined by SNA.

Chapter 4, "Defining Network Resources" discusses the requirements and methodology for the definition of network resources.

Chapter 5, "Activating the Network" explains network activation within SNA networks.

Chapter 6, "Establishing Routes Through the Network" explains how the transport network selects routes through the network in accordance with a requested class of service.

Chapter 7, "Controlling Congestion in the Network" explains the roles of transport network transmission protocols and pacing algorithms in moving data efficiently through the network.

Chapter 8, "Transporting Data Through the Network" discusses session protocols for controlling end-to-end data transport.

Chapter 9, "Transaction Services" discusses architectures within the SNA transaction services layer that aid in managing end-user data in a network.

Chapter 10, "Managing an SNA Network" discusses SNA management services, the functions of SNA components in management services, and the management services request units.

Appendix A, "Sequence Charts" presents sequence charts that illustrate typical request unit flows for activating, controlling, and deactivating network resources.

A Glossary of terms and abbreviations, a Bibliography, and an Index are at the end of this book.

Softcopy

This publication is available as a softcopy book. The softcopy book is on an electronic bookshelf and is part of both the *IBM Networking Systems Softcopy Collection Kit* (SK2T-6012) and *The Best of APPC, APPN, and CPI-C* (SK2T-2013) on compact disk read-only memory (CD-ROM).

You can view and search softcopy books by using BookManager* READ products or by using the IBM Library Reader* product included on each CD-ROM. For more information on CD-ROMs and softcopy books, see *IBM Online Libraries: Softcopy Collection Kit User's Guide* (GC28-1700) and BookManager READ documentation.

Where to Find More Information

No prerequisite reading exists for data processing personnel familiar with SNA concepts and terminology.

The following publications about SNA and related topics may be of interest:

- *Data Security Through Cryptography*, GC22-9062
- *IBM SDLC Concepts*, GA27-3093
- *IBM 3270 Data Stream Programmer's Reference*, GA23-0059
- *Networking Blueprint Executive Overview*, GC31-7057
- *Non-SNA Interconnection General Information Manual*, GC33-2023
- *Systems Network Architecture Format and Protocol Reference Manual: Architectural Logic*, SC30-3112
- *Systems Network Architecture Format and Protocol Reference Manual: Architecture Logic for LU Type 6.2*, SC30-3269
- *Systems Network Architecture LU 6.2 Reference—Peer Protocols*, SC31-6808
- *Systems Network Architecture Format and Protocol Reference Manual: SNA Network Interconnection*, SC30-3339
- *Systems Network Architecture Formats*, GA27-3136
- *Systems Network Architecture Management Services Reference*, SC30-3346
- *Systems Network Architecture—Sessions Between Logical Units*, GC20-1868
- *Systems Network Architecture Transaction Programmer's Reference Manual for LU Type 6.2*, GC30-3084
- *Systems Network Architecture APPN Architecture Reference*, SC30-3422
- *The X.25 Interface for Attaching Packet-Switched Data Networks General Information Manual*, GA27-3345
- *Token-Ring Network Architecture Reference*, SC30-3374

Refer to the following publications for information about SNA's transaction services:

- *Distributed Data Management Architecture: General Information Manual*, GC21-9527
- *Document Interchange Architecture: Technical Reference*, SC23-0781

- *Systems Network Architecture Distribution Services Reference*, SC30-3098
- *Systems Network Architecture File Services Reference*, SC31-6807

Note: This edition of the *SNA Technical Overview* includes information previously found in the *SNA Concepts and Products*. *SNA Concepts and Products* is now obsolete.

Summary of Changes

This edition of *Systems Network Architecture Technical Overview* includes information about the following subject areas:

- Advanced Peer-to-Peer Networking enhancements:
  - Peripheral and extended border nodes
  - VTAM extensions to APPN such as:
    - Interchange node
    - Central directory server (CDS)
  - Central resource registration (CRR)
  - Dependent LU server/dependent LU requester
  - APPN Topology and Accounting Manager (APPNTAM).
- High-performance routing:

  High-performance routing (HPR) is an evolutionary extension to APPN that enhances data routing performance and session reliability. Described in detail are the following concepts of HPR:

  - Rapid-transport protocol (RTP)
  - Automatic network routing (ANR)
  - Nondisruptive path switch
  - Adaptive rate-based (ARB) flow/congestion control.

Chapter 1. Introduction

This chapter provides an overview of SNA, including its architectural objectives, network components, data transport services, transaction services, and network management services.

The Architecture

A data communication network is a collection of hardware and software components that enable end users to exchange data. To send or receive data through a network, end users interact with communication devices such as telephones, terminals, or computers. This publication uses the term ***end user*** to identify both (1) individuals who interact with the network through a workstation and (2) application programs. The architecture views end users as the ultimate sources and destinations of information that flows through a network.

Data communication requires that network components agree on both the layouts of the messages they exchange and the actions they take based on the kinds of data they receive. The layouts are referred to as ***formats***, and the actions taken are referred to as ***protocols***. Formats and protocols together constitute an architecture.

Systems Network Architecture (SNA) is a data communication architecture established by IBM to specify common conventions for communication among the wide array of IBM hardware and software data communication products. The manner in which products internally implement these common conventions can differ from one product to another. But because the external *interface* of each implementation is compatible, different products can communicate without the need to distinguish among the many possible product implementations.

SNA Layers

SNA functions are divided into a hierarchical structure that consists of seven well-defined layers. Each layer in the architecture performs a specific set of functions. Figure 1 identifies SNA's seven layers and their major functions.

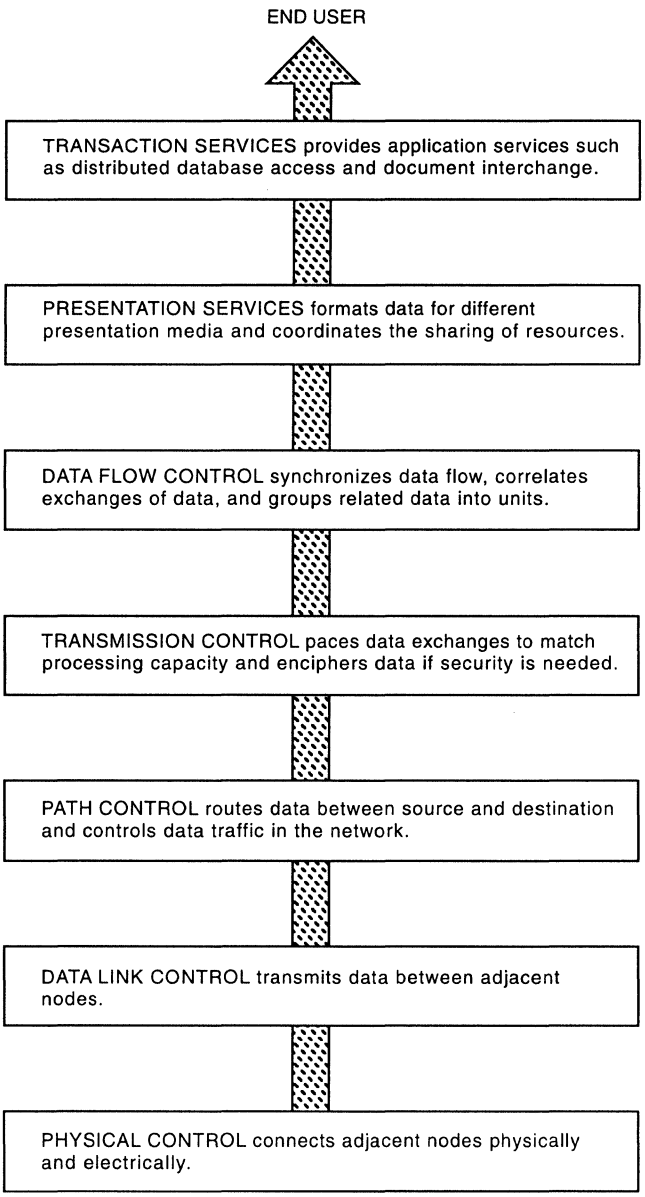


Figure 1. A Layered Architecture

SNA defines formats and protocols between layers that permit equivalent layers (layers at the same level within the hierarchy) to communicate with one another. Each layer performs services for the next higher layer, requests services from the next lower layer, and communicates with equivalent layers. To illustrate this concept, consider end-user data that requires encryption. Figure 2 illustrates how the two transmission control layers communicate.

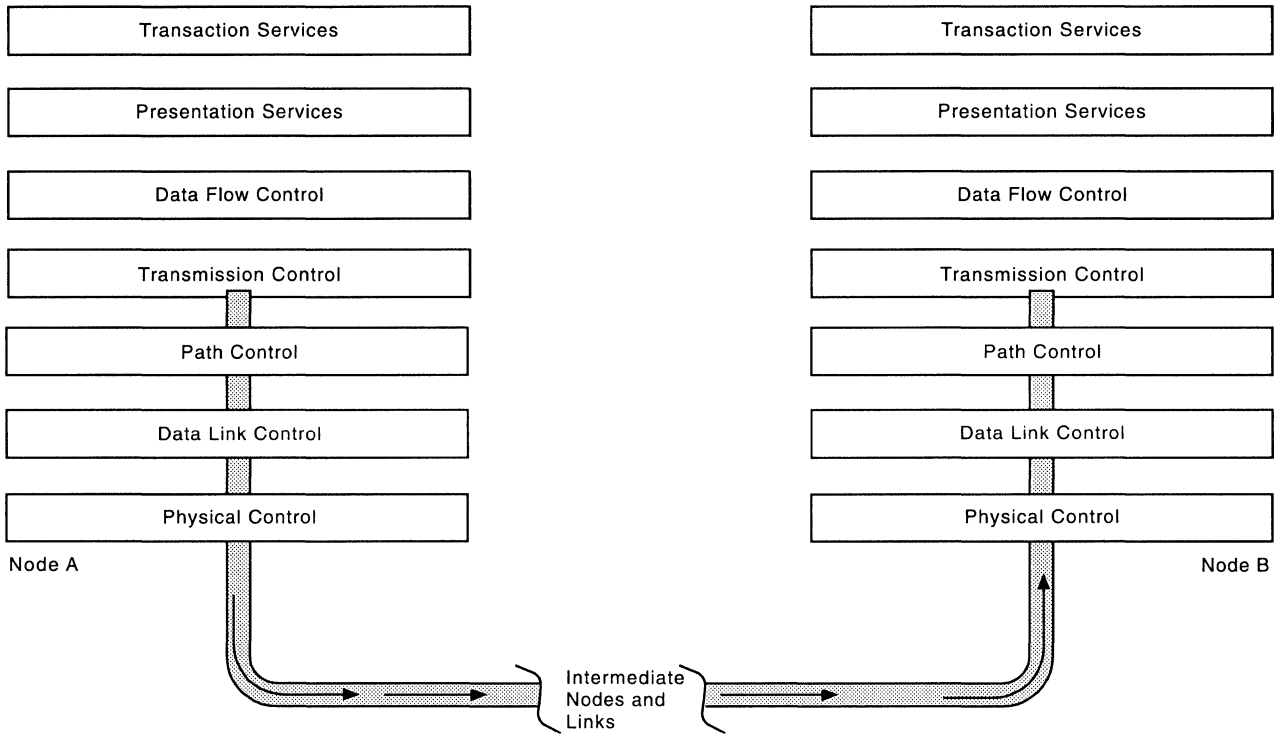


Figure 2. Communication Between Two Transmission Control Layers

The two transmission control layers encipher and decipher data independently of the functions of any other layer. The transmission control layer in the originating node enciphers the data it receives from the data flow control layer. It then requests that the path control layer route the enciphered data to the destination node. The transmission control layer in the destination node deciphers the data that the path control layer delivered. It then requests that the data flow control layer give the deciphered data to the destination end user.

Hardware and software components implement the functions of the seven architectural layers. Hardware components include:

- Processors such as the ES/9000* family
- Distributed processors such as the Application System/400*
- Communication controllers such as the 372X and 374X series
- Cluster controllers
- Workstations
- Printers.

The software components that implement SNA functions include:

- Operating systems such as Multiple Virtual Storage/Enterprise Systems Architecture (MVS/ESA*), and Operating System/400* (OS/400*).
- Telecommunication access methods such as the Virtual Telecommunications Access Method (VTAM*) and Communications Manager/2 (CM/2*)
- Application subsystems such as Customer Information Control System (CICS)
- Network control programs such as the Advanced Communication Function for Network Control Program (NCP).

Figure 3 illustrates one possible network configuration of these hardware and software components.

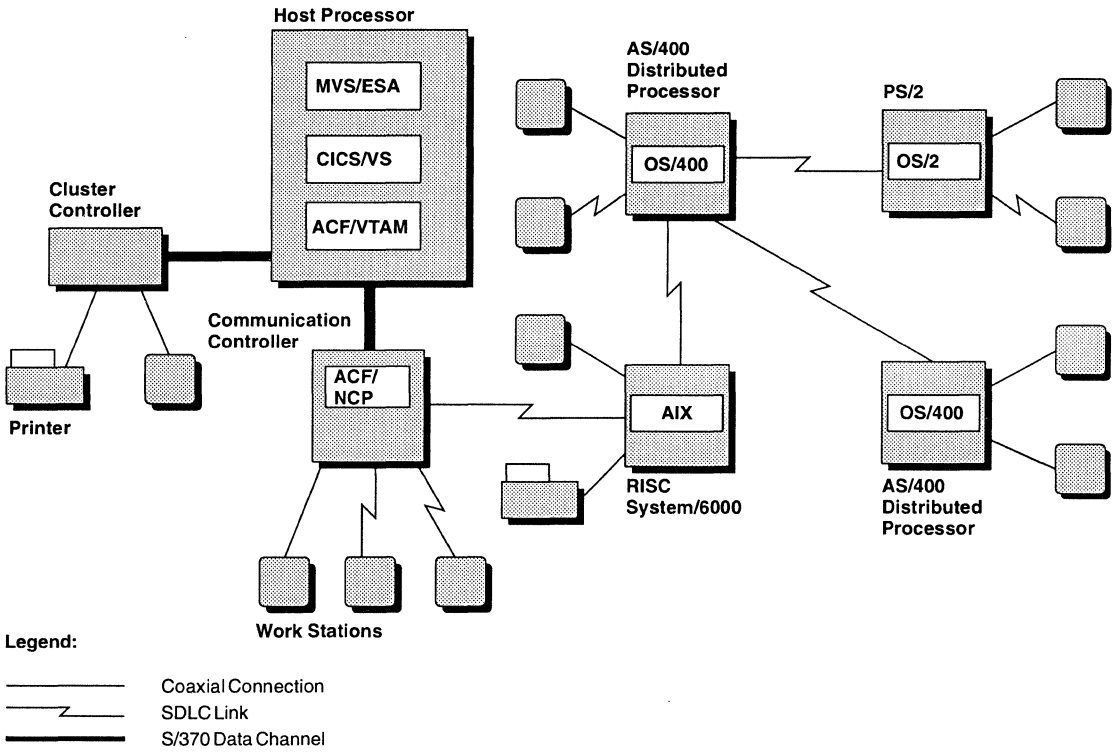


Figure 3. Hardware and Software Components of an SNA Network

SNA and the Networking Blueprint

IBM's networking design strategy employs a "Networking Blueprint" that provides an open, highly modular framework for structuring networks using industry-wide standards. The blueprint is designed to facilitate the creation of an evolutionary and flexible plan that an organization can specifically tailor to meet its needs; this is much needed considering today's fast moving technologies and the diversity of networks. Figure 4 is a picture of the modules used to describe the Networking Blueprint.

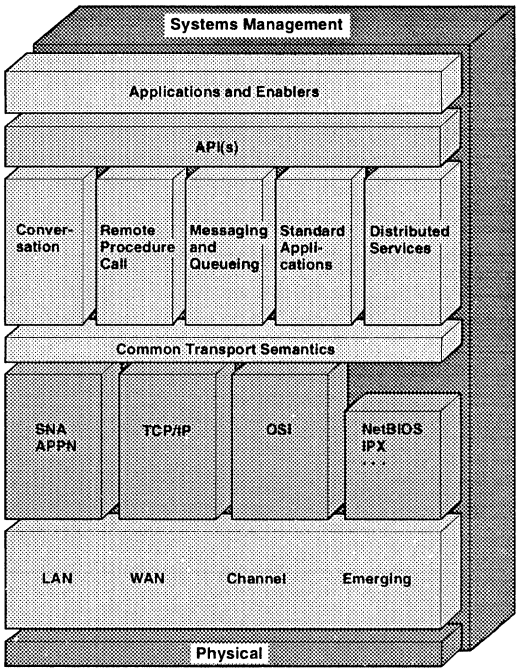


Figure 4. The Networking Blueprint

The Networking Blueprint supports the implementation of multiple protocols, as well as integrating these protocols into a cohesive, modular structure. The Networking Blueprint defines layers of functions, plus a systems management backplane. SNA Advanced Peer-to-Peer Networking* (APPN*), is part of the transport layer, and it is one of the protocols that can be used in this layer. For more information on the Networking Blueprint, refer to the *Networking Blueprint Executive Overview*.

Advanced Peer-to-Peer Networking (APPN)

Today's complex networks require a new approach to networking—an intelligent technology that ties together diverse platforms, topologies, and applications into a single network. That approach is ***Advanced Peer-to-Peer Networking (APPN)***. APPN's any-to-any connectivity makes it possible for large and small networks alike to communicate over local- and wide-area networks, across slow and fast links.

APPN provides two basic functions: keeping track of the location of resources in the network, and selecting the best path to route data between resources. APPN nodes dynamically exchange information about each other; therefore, customers may never have to deal with complicated system and path definitions. APPN nodes limit the information they exchange, enabling more efficient use of network resources.

High-performance routing (HPR) is a small but powerful evolutionary extension to APPN. It enhances data routing performance via decreasing intermediate node processing. HPR also increases session reliability via "nondisruptive path switch," described later. The two main components of HPR are ***rapid-transport protocol (RTP)*** and ***automatic network routing (ANR)***, described later.

Architectural Objectives

Systems Network Architecture was designed with certain objectives in mind. These objectives address common concerns of data communication network users.

Promote Reliability

One such concern is the reliability of the network. Recoverable data communication errors must be handled transparently to the user. At the data link layer of the architecture, error checking protocols ensure that error detection and message retries are performed automatically. When errors are not recoverable, communicating partners must be able to achieve a mutual understanding as to the results of attempted message transfers. Protocols at the presentation services layer provide positive and negative responses and establish ***synchronization points***, which enable communicating partners to ensure the consistency of their resources.

For some applications, consistent, fast performance is needed to handle updates immediately; for other applications, it is important to send data as inexpensively as possible. Advanced Peer-to-Peer Networking (APPN) includes several features that can handle this type of traffic mix efficiently. APPN's priority services ensure that important data move through the network quickly. Similarly, by using intelligent class-of-service routing, APPN nodes consider factors such as security, cost, delay, and throughput to select the best route for different types of data. Unlike other protocols that react to network bottlenecks by dropping and resending packets, APPN avoids network congestion by using adaptive pacing, which ensures a higher, more consistent traffic volume.

High-performance routing (HPR) improves on APPN's reliability by providing greater session reliability in case of link and node failure via nondisruptive path switch. It is accomplished transparent to both the sessions and end-users.

Enhance Network Dependability

An SNA network is dependable because SNA products recognize and recover from loss of data during transmission, use flow control procedures to prevent data overrun and avoid network congestion, identify failures quickly, and recover from many errors with minimal involvement of network users. SNA products also increase network availability through options such as the extended recovery facility, backup host, alternate routing capability, and maintenance and recovery procedures integrated into workstations, modems, and controllers.

Certain SNA peer-to-peer networking protocols reduce the negative impact of a failure of an individual component in the system. They do this by decentralizing control of the network and selecting routes based only on the current status of the network topology, not on predefined static information.

Promote Efficiency

Another concern is the efficiency with which data is transferred. Components within SNA act to promote efficiency both by selecting options that maximize data transmission throughput and by taking action to reduce network congestion when it is detected. Route selection services, for example, can select optimal routes for data transmission.

When too much data is introduced into the network, congestion can occur. Severe congestion blocks the flow of messages and can result in the loss of data. Proto-

cols at the transmission control and path control layers prevent overload and deadlocks by controlling the pace at which messages flow through the network.

Advanced Peer-to-Peer Networking (APPN) features help eliminate unnecessary network control traffic, thus providing more bandwidth for moving data through the network. APPN nodes never broadcast changes to all machines in the network. Instead, only network nodes exchange topology information, and they exchange this information only when changes occur in the backbone of the network. Other protocols broadcast routing information at frequent regular intervals, even if nothing changes in the network. Other APPN features, like directory caching and central directory servers, limit searches for other resources in the network, and, as a result, improve network performance. In summary, APPN permits more bandwidth for real work for applications, without the need to invest in new equipment.

Promote Ease of Use

An ongoing objective of SNA is ease of use, both for end-users and network personnel. Because SNA products have compatible interfaces, they can connect and communicate with one another. ***Application program interfaces*** at the presentation services layer, for example, shield end-users from concern with the underlying details of communication protocols. When hardware and software upgrades are required, the functional independence of the SNA layers enables any one layer to be enhanced or modified without disrupting the functions of any other layer. To link independent networks, network interconnection protocols are provided that enable the networks to communicate without redefining network identifiers. An example of this is Common Programming Interface for Communications (CPI-C), which can protect an organization's investment in application programming. Finally, many of SNA's more recent enhancements ease the task of system definition. ***Dynamic definition of independent and dependent LUs***, for example, eliminates the need to specify independent and dependent logical units in system definition statements. Also, the services provided by ***Advanced Peer-to-Peer Networking*** (APPN) include network topology updates and automatic route selection, which make it unnecessary to predefine node locations and routes between nodes. Thus, network operators can add new components to the network without affecting the network availability for existing users.

Improve End-User Productivity

With peer-to-peer communication protocols, end-users benefit through more timely access to their applications because they are less dependent on the system programming staff to code network definitions. As workstations and applications first become available in the network, or are moved, APPN directory functions locate them dynamically without requiring coordinated definitions to be coded by the system programmer. This improves network availability and end-user productivity.

Allow for Resource Sharing

Resource sharing addresses the concern for fair and effective resource utilization. The sharing of access to storage devices, output devices, and data communication lines is paramount for containing network costs. Resource sharing is addressed at many levels within SNA, from the multiplexing of data links to the sharing of sessions by individual end-user transactions.

At the same time other mechanisms control the equitability with which services are provided to network users. ***Transmission priorities*** and ***classes of service*** enable equal service to be given to sessions of equal priority, or preference given to sessions of higher priority.

Provide for Network Security

The security of data is an increasing need for today's networks, which have become the vehicles for such sensitive data as banking transactions. Protocols for data encryption at the transmission control layer, and password verification at the transaction services layer and the presentation services layer serve the data security objective.

Provide for Resource Management

Tools for resource management provide the ability to identify errors, help in problem determination, and maintain accounting data on network resource usage. Capabilities exist within all layers of SNA for monitoring and reporting errors relating to the functions for which they are responsible. Usage statistics facilitate fair charging of network users, performance tuning, capacity planning, and capital budgeting.

Protect Network Investment

IBM offers communication products that conform to SNA specifications. Because of IBM's ongoing development of products compatible with this architecture, organizations often can substitute one type of SNA product for another as their needs change. In an SNA network, newer devices with improved capabilities can coexist with older ones. As an organization's SNA network evolves with the addition of new workstations, processors, communication facilities, and applications, it can continue using the applications already in place.

Subarea Networks: Networks that use peer-to-peer communication protocols can be integrated with subarea networks, allowing application sessions between peer-to-peer nodes and hierarchical nodes.

Dependent LU Support: Users of APPN can carry their investment in dependent LUs into the future of dynamic Advanced Peer-To-Peer (APPN) networking. Dependent LUs can continue to access VTAM applications in the same VTAM domain, or in different domains, using subarea protocols. In addition, dependent LUs in one domain can access VTAM applications in a different domain utilizing APPN protocols between domains. This allows the user to begin implementing APPN instead of subarea protocols between VTAM domains while continuing to provide access to VTAM applications from dependent LUs in different domains.

Applications: Existing applications continue to be supported for dependent LUs and independent LUs. There is no change to APIs.

Ease of Migration: Migrating to a client/server environment does not have to be complex, time-consuming, and expensive. Advanced Peer-to-Peer Networking (APPN) provides a better solution for migrating both networks and applications. With APPN, a network can be migrated to the client/server environment at any pace. The new version of VTAM, for example, allows traditional SNA networks to be integrated with APPN networks. APPN also enables running both existing mainframe applications and new client/server applications at the same time.

Simplify Problem Determination

SNA helps find and resolve problems that can occur in a network. SNA management services functions in each network component to assist in problem determination. Also, the NetView* family of products, which is network management software that runs on a host processor, has components that monitor, collect, and store network data. Other system management functions built into the NetView program allow rapid problem determination, increase network availability, and minimize the number of the personnel needed to operate and maintain the SNA network.

Accommodate New Facilities and Technologies

IBM continues to maintain an open-ended architecture while developing new products to meet an organization's information systems needs. An SNA network accommodates such facilities and technologies as:

- Digital networks
- Digitized voice
- Distributed systems
- Fiber optics
- Graphics
- Public packet-switching data networks
- Satellites
- Token-ring networks
- Videotex
- Client/server
- Ethernet
- Frame relay
- Security.

Let Independent Networks Communicate

The SNA Network Interconnection (SNI) facility helps exchange information with another independent organization or merge separately administered subarea networks into one.

The SNA Network Interconnection facility allows users in one SNA network to access information and application programs in other SNA networks. Each interconnected network can maintain its existing management procedures and controls. A "gateway" between two or more SNA networks connects them operationally while isolating their administrative characteristics from one another. Network users are not aware of network boundaries.

Two APPN subnets may communicate with each other via a "border node." The border node allows sessions between users in different APPN subnets while isolating the subnets' topology information from one another.

Architectural Components of an SNA Network

A data communication network can be described as a configuration of nodes and links. Nodes are the network components that send data over, and receive data from, the network. Node implementations include processors, controllers, and workstations. Links are the network components that connect adjacent nodes. Nodes and links work together in transferring data through a network.

Nodes

A ***node*** is a set of hardware and associated software components that implement the functions of the seven architectural layers. Although all seven layers are implemented within a given node, nodes can differ based on their architectural components and the sets of functional capabilities they implement. Nodes with different architectural components represent different ***node types***. Four types of nodes exist: type 5 (T5), type 4 (T4), type 2.0 (T2.0), and type 2.1 (T2.1). Their architectural components are described in Chapter 2, "Link and Node Components."

Nodes that perform different network functions are said to act in different ***network roles***. It is possible for a given node type to act in multiple network roles. A T4 node, for example, can perform an interconnection role between nodes at different levels of the subarea network hierarchy, or between nodes in different subarea networks. The functions performed in these two roles are referred to as ***boundary function*** and ***gateway function***, respectively. T2.1 and T5 nodes can also act in several different network roles. Node roles fall into two broad categories: ***hierarchical roles*** and ***peer-oriented roles***.

Hierarchical Roles

Hierarchical roles are those in which certain nodes have a controlling or *mediating* function with respect to the actions of other nodes. Hierarchical networks are characterized by nodes of all four types acting in hierarchical roles. Within such networks, nodes are categorized as either ***subarea nodes*** (SNs) or ***peripheral nodes*** (PNs). Subarea nodes provide services for and control over peripheral nodes. Networks consisting of subarea and peripheral nodes are referred to as ***subarea networks***.

Subarea Nodes: Type 5 (T5) and type 4 (T4) nodes can act as subarea nodes. T5 subarea nodes provide the SNA functions that control network resources, support transaction programs, support network operators, and provide end-user services. Because these functions are provided by host processors, T5 nodes are also referred to as ***host nodes***. T4 subarea nodes provide the SNA functions that route and control the flow of data in a subarea network. Because these functions are provided by communication controllers, T4 nodes are also referred to as ***communication controller nodes***.

Peripheral Nodes: Type 2.0 (T2.0) and type 2.1 (T2.1) nodes can act as peripheral nodes attached to either T4 or T5 subarea nodes. Peripheral nodes are typically devices such as distributed processors, cluster controllers, or workstations. A T2.1 node differs from a T2.0 node by the T2.1 node's ability to support peer-oriented protocols as well as the hierarchical protocols of a simple T2.0 node. A T2.0 node requires the mediation of a T5 node in order to communicate with any other node. Subarea nodes to which peripheral nodes are attached perform a ***boundary function*** and act as subarea ***boundary nodes***.

Although Figure 3 represents nodes as particular classes of hardware, there is no architectural association between node type, or node role, and the kind of hardware that implements it. To avoid associating node types and roles with hardware implementations, network architecture diagrams use symbols to represent node types and roles. Figure 5 uses symbols to illustrate a subarea network containing the four node types acting as subarea and peripheral nodes. The network contains two type 5 subarea nodes, three type 4 subarea nodes, and seven peripheral nodes.

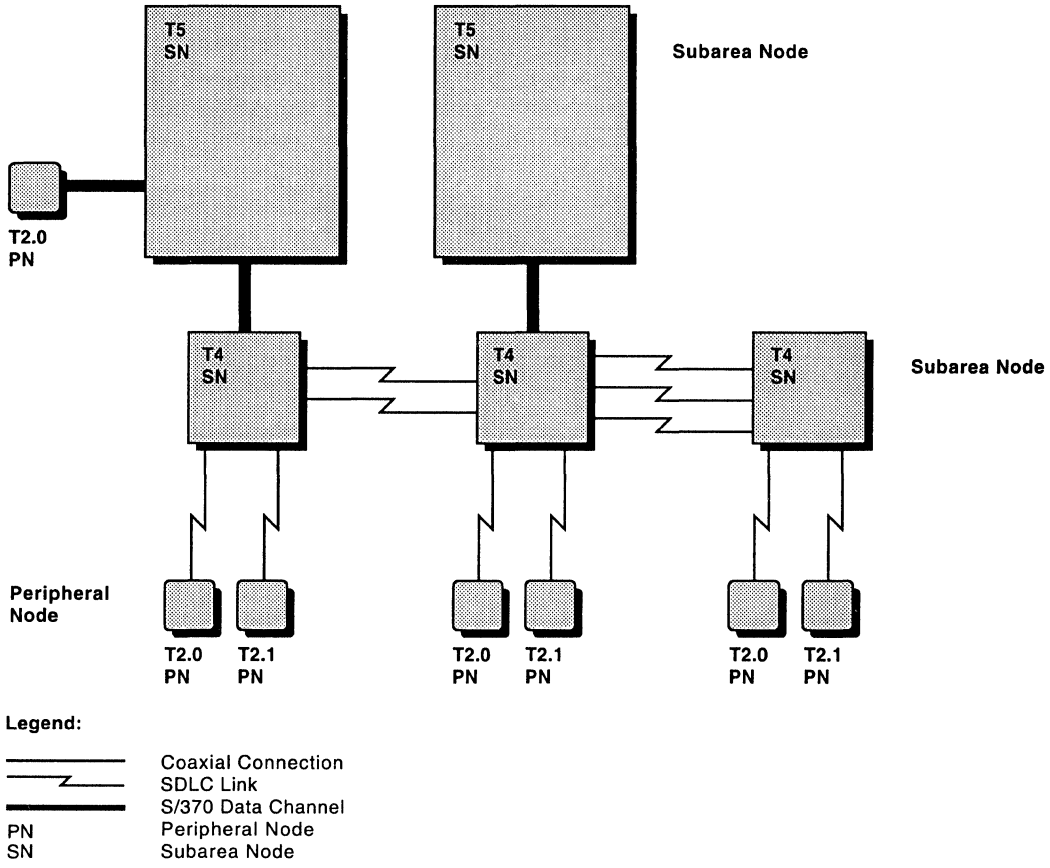


Figure 5. A Subarea Network

Peer-Oriented Roles

The Advanced Peer-to-Peer Networking (APPN) extensions allow greater distribution of network control by enhancing the dynamic capabilities of the node. Nodes with these extensions are referred to as ***APPN nodes***, and a network of APPN nodes makes up an ***APPN network***. A low-entry networking (LEN) node can also attach to an APPN network. An APPN node can dynamically find the location of a partner node, place the location information in directories, compute potential routes to the partner, and select the best route from among those computed. These dynamic capabilities relieve network personnel from having to predefine those locations, directory entries, and routes. APPN nodes can include processors of varying sizes such as the Application System/400 (AS/400), the Enterprise System/9370* (ES/9370*) running under Distributed Processing Program Executive/370 (DPPX/370), the Personal System/2* (PS/2*) running under Operating System/2* (OS/2*), and VTAM running under Multiple Virtual Storage/Enterprise Systems Architecture (MVS/ESA).

Peer-oriented protocols enable nodes to communicate without requiring mediation by a T5 node, giving them increased connection flexibility. APPN defines two possible roles for a node in an APPN network, that of an end node and that of a network node. (Network nodes provide additional options that can further distinguish them.)

T2.1 nodes can act as either APPN or LEN nodes. T5 nodes can also act as APPN or LEN nodes, but have additional capability to interconnect subarea and

APPN networks by interchanging protocols between them. (In this capacity, they are called *interchange nodes*, discussed later.) Together with its subordinate T4 nodes, a T5 node can also form a *composite LEN node* or a *composite network node*. As composite nodes, they appear as single LEN or network nodes to other LEN or APPN nodes to which they are interconnected.

If two network nodes do not support the border node option (discussed later) and are located in two separate net-ID subnetworks, CP-CP sessions cannot be established between them. For the two network nodes to communicate, a LEN connection may be established between the two. This allows the two net-ID subnetworks to communicate, but does not support any APPN function. If NN1 in subnet A is to establish a session with NN2 in subnet B, all LUs in subnet A must be predefined to NN2 in subnet B. The network nodes in both subnets are APPN nodes, but since they communicate across net-ID subnetwork boundaries, they are defined to each other via LEN links.

End Nodes: Are located on the periphery of an APPN network. An end node obtains full access to the APPN network through one of the network nodes to which it is directly attached—its ***network node server***. The two kinds of end nodes are APPN end nodes and LEN end nodes. An ***APPN end node*** supports APPN protocols through explicit interactions with a network node server. Such protocols support dynamic searching for resources and provide resource information for the calculation of routes by network nodes. A ***LEN end node*** is a LEN node attached to a network node. Although LEN nodes lack the APPN extensions, they are able to be supported in APPN networks using the services provided them by network nodes. In an APPN network, when a LEN node is connected to another LEN node, or to an APPN end node, it is referred to simply as a *LEN node*. When connected to an APPN network node, however, it is referred to as a *LEN end node*.

Network Nodes: Together with the links interconnecting them, network nodes form the ***intermediate routing network*** of an APPN network. Network nodes connect end nodes to the network and provide resource location and route selection services for them. Routes used to interconnect network users are selected based on network topology information that can change dynamically.

Figure 6 represents one possible APPN network configuration and contains LEN end nodes as well as APPN nodes.

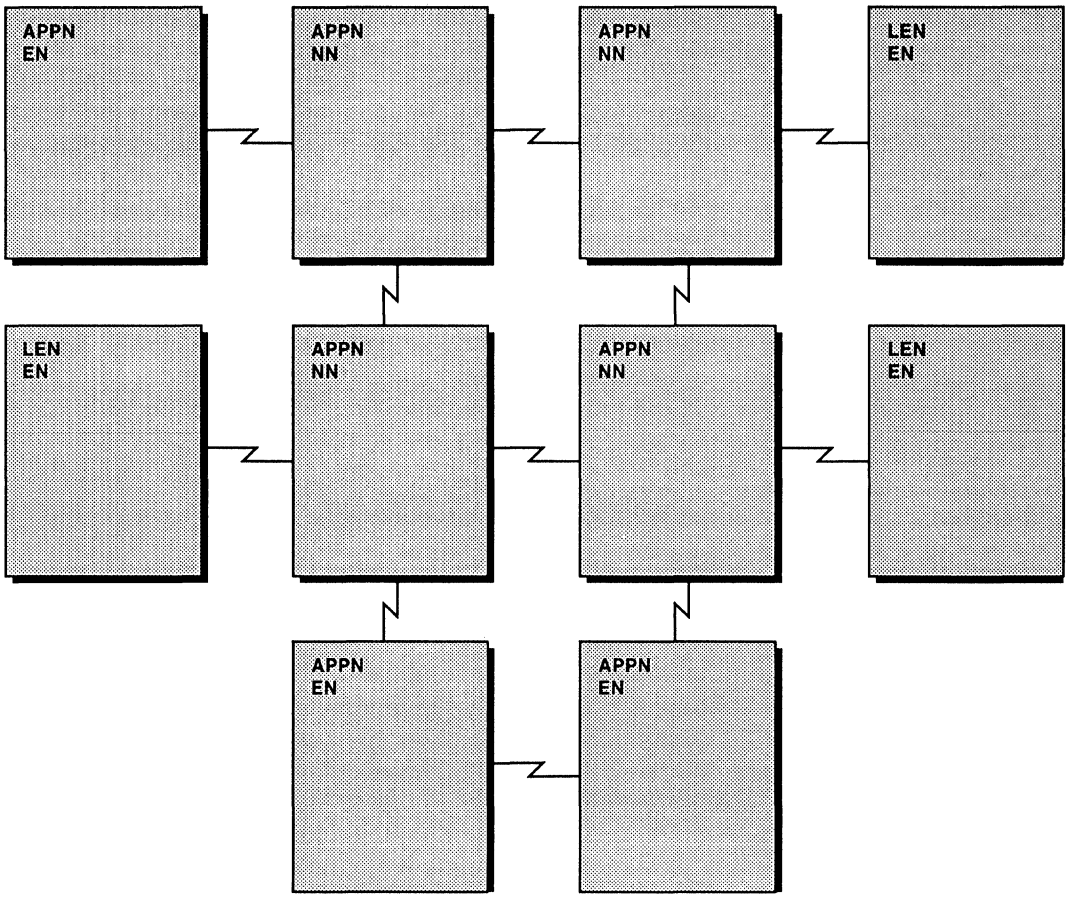


Figure 6. An APPN Network

Network Node Server: A network node that provides resource location and route selection services to the LUs it serves. These LUs can be in the network node, itself, or in the client end nodes. A network node server uses CP-CP sessions to provide network information for session setup in order to support the LUs on served APPN end nodes. In addition, LEN end nodes can also take advantage of the services of the network node server. A LEN end node, unlike an APPN end node, must be predefined by the network operator as a client end node for which the network node acts as server. Any network node can be a network node server for end nodes that are attached to it. The served end nodes are defined as being in that network node server's domain.

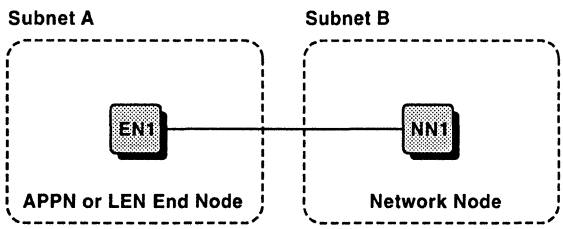


Figure 7. An End Node Attached Nonnatively to Its Network Node Server

Figure 7 illustrates an end node connecting to a network node server with a different net ID from its own. The capability allows the end node to dial into different APPN networks. This type of attachment is called a nonnative attachment.

Central Directory Server (CDS): A network node that builds and maintains a directory of resources from the network. The purpose of a CDS is to reduce the number of network broadcast searches to a maximum of one per resource. Network nodes and APPN end nodes can register their resources with a CDS, which acts as a focal point for resource location information.

A CDS can be involved in APPN end node resource registration. An APPN end node registers its resources to improve network search performance. When an APPN end node registers its resources with its network node server, it can request that its network node server also register them with a CDS. Entries in a directory database can be registered, defined, or dynamic.

When a network node receives a search request for a resource that has no location information, the network node first sends a directed search request to a CDS if there is one. The CDS searches in its directory for information about the location of the resource. If it does not find the location of the resource, the CDS searches end nodes in its domain, other CDSs, and, if necessary, the entire network (via a "broadcast search"). If the resource is still not found, the CDS notifies the network node that originally requested the search that the search is unsuccessful. A ***central directory client*** is a network node that forwards directory searches to a CDS. For more information on CDS, see "Central Directory Server (CDS)" in Chapter 5, "Activating the Network" and "Use of a Central Directory Server in a Search" in Chapter 6, "Establishing Routes Through the Network."

Subnetworks and Border Nodes: A **subnetwork** consists of a group of interconnected nodes, within a larger, composite network, that have some common attribute or characteristic, such as the same network ID, or that share a common topology database, or that implement a common protocol. A *net-ID subnetwork* refers to the set of nodes having the same network ID. A *topology subnetwork* consists of all the network nodes exchanging and maintaining the same topology information. A *high-performance routing (HPR) subnetwork* consists of all the interconnected APPN nodes implementing the HPR function, discussed later. The terms *subnetwork* and *network*, as used in this book, relate to one another analogously to *subset* and *set* in general usage. As indicated by the above examples, the adjective preceding *subnetwork* identifies the subsetting attribute.

Within an APPN network, it is often desirable to partition the network topology database to reduce its size at each network node and lower the overall topology data interchange traffic. Topology database replication and interchange is confined to a topology subnetwork, with each subnetwork acting independently. The presence of **border nodes** enables the topology subnetworks to be tied together into a larger composite network with the same freedom of LU-LU sessions as if the topology partitioning did not exist. Border nodes are network nodes having additional function and are of two types: *peripheral* and *extended*.

Peripheral border nodes permit APPN networks with different net IDs to interconnect, allowing session setup across net-ID subnetwork boundaries. Extended border nodes can also interconnect distinct net-ID subnetworks, but in addition allow the partitioning of the topology database among network nodes with the same net ID; this is known as ***clustering*** and the network partitions are also known as ***clusters***. Each cluster is a distinct topology subnetwork.

Each individual APPN network maintains its own topology database, which is not known by other APPN networks. Therefore, topology database updates (TDUs),

indicating topology changes and flowing within subnets, are prevented from crossing subnet boundaries. An immediate benefit of using border nodes is that the number of TDUs flowing through topology subnets will be lower, thus reducing the storage requirements for the network topology database in network nodes in each of the subnets. This reduces network flows and allows network nodes with limited resources to participate in APPN networking more efficiently.

A peripheral border node provides directory, session setup, and route-selection services across the boundary between adjacent networks with different net IDs—each such network being referred to as a distinct net-ID subnet in the composite network. The peripheral border node isolates each subnet from the other subnet's topology information; the border node receives TDU messages only from network nodes within its native subnet. It allows session establishment between LUs located in adjacent subnets. A network node, LEN end node, APPN end node, or another peripheral border node can attach natively or nonnatively to a peripheral border node. A peripheral border node can act as a basic network node. Peripheral border nodes do not support intermediate network routing; they must be located in peripheral subnets in order to have the border node function. Figure 8 illustrates two subnets, subnet A and subnet B, interconnected by a peripheral border node. Subnet A will not receive TDU messages from subnet B and vice versa.

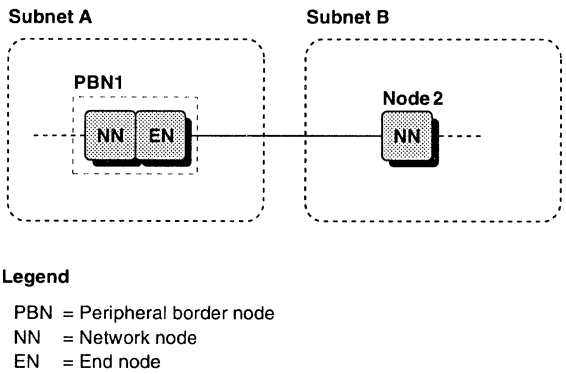


Figure 8. Peripheral Border Node Connections

Figure 9 illustrates the dual nature of a border node. XID exchange determines which peripheral border node presents the network node image and which one presents the end node image.

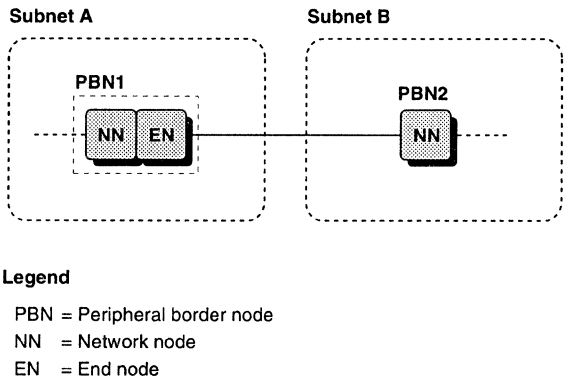
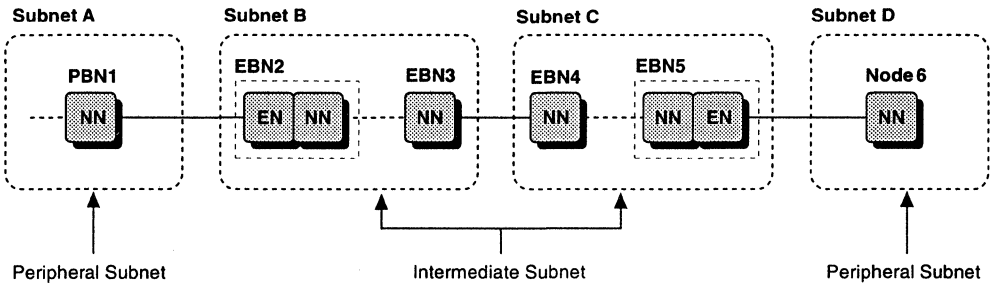


Figure 9. Peripheral Border Node to Peripheral Border Node Connection

For more information on peripheral border nodes, see "Defining APPN Subnets Containing Peripheral Border Nodes" in Chapter 4, "Defining Network Resources."

An extended border node provides directory, session setup, and route selection services across the boundary between paired or cascaded nonnative net-ID subnets, while isolating each subnet from the topology information of the other networks. An extended border node can also partition a single net-ID subnet into two or more isolated topology domains, or clusters. An advantage that extended border nodes have over peripheral border nodes is that sessions can be established between LUs located in nonadjacent subnets. Extended border nodes, unlike peripheral border nodes, support intermediate network routing with the option of limiting the number of subnets traversed. A network node, LEN end node, APPN end node, peripheral border node, or an extended border node can attach natively or nonnatively to an extended border node. Figure 10 illustrates an APPN configuration with extended border nodes.



Legend

PBN = Peripheral border node
EBN = Extended border node
NN = Network Node
EN = End Node

Figure 10. Extended Border Node Support

In Figure 10, subnet B and subnet C are the intermediate subnets. Subnet A and subnet D are the peripheral subnets; a peripheral subnet can contain an endpoint of a multisubnet session, but may never act as an intermediate subnet to connect two different subnets. At the boundary of a peripheral subnet is either a peripheral border node, an extended border node, or a basic network node.

The border node roles are as follows:

- PBN1 in subnet A acts as a network node.
- From subnet A's point of view, EBN2 in subnet B acts as an APPN end node.
- From subnet B's point of view, EBN2 in subnet B acts as a network node.
- EBN3 and EBN4 act as network nodes.
- From subnet C's point of view, EBN5 in subnet C acts as a network node.
- From subnet D's point of view, EBN5 in subnet C acts as an APPN end node.
- Node6 in subnet D acts as a network node.

An extended border node may appear as a network node to one subnet and at the same time, may appear as an end node to a second subnet.

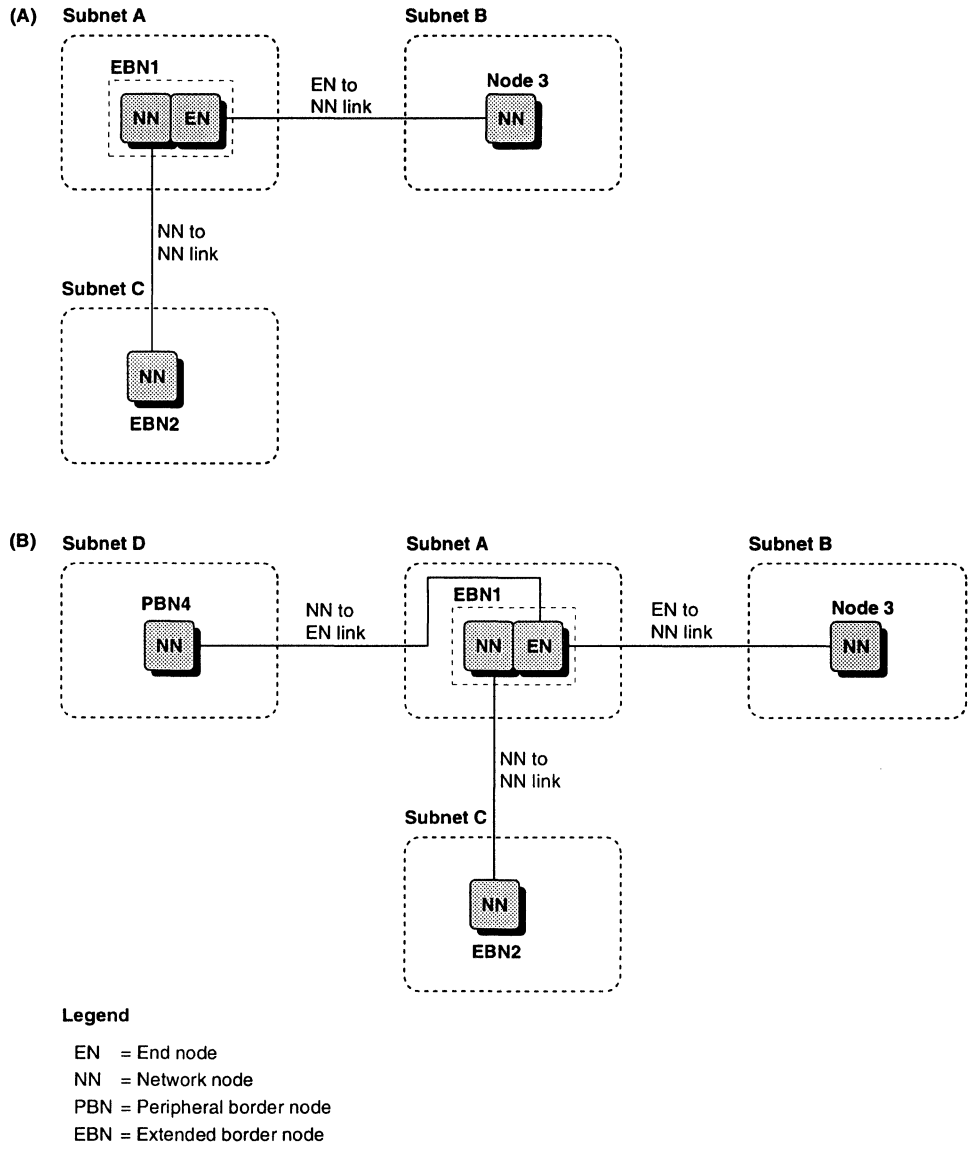


Figure 11. Extended Border Node with Peripheral Border Node Function

Figure 11 illustrates an extended border node that uses both extended border node and peripheral border node functions. In Figure 11 (A), EBN1 in subnet A may have CP-CP sessions with Node3 which is a network node in subnet B. At the same time, EBN1 may have CP-CP sessions with EBN2 in subnet C. The advantage of having an extended border node in subnet A is that if the network grows, the extended border node may be connected to an extended border node, a peripheral border node, or a basic network node in another subnet for multisubnet connectivity. In Figure 11 (B), subnet D is added to the network configuration. EBN1 may establish sessions with PBN4 in subnet D and therefore be able to act as an intermediate subnet between subnet B and subnet D.

For more information on extended border nodes, see "Defining APPN Subnets Containing Peripheral Border Nodes" in Chapter 4, "Defining Network Resources."

Nodes with both Hierarchical and Peer-Oriented Function

Interchange Node: The ***interchange node*** is a VTAM product feature merging APPN and subarea capabilities in one node. It enables the integration of APPN networks and subarea networks. An interchange node receives network search requests from APPN nodes and transfers them into subarea network searches, without exposing the subarea aspects to the APPN part of the network. At the same time, the interchange node can receive search requests from a subarea network and transfer them into APPN search requests, without exposing the APPN aspects to the subarea network. The interchange node maintains its subarea appearance to other subarea nodes and maintains its APPN appearance to other APPN nodes.

The interchange node supports SSCP-SSCP sessions with other VTAM nodes as well as CP-CP sessions with adjacent APPN network nodes and end nodes. This support allows the interchange node to use both APPN and subarea data flows to locate LUs and to provide the best route between nodes. APPN session setup protocols, which employ CP-CP sessions, are converted to the corresponding subarea protocols for use on SSCP-SSCP sessions and vice versa. Figure 12 illustrates an intermediate stage of migrating a subarea network to an APPN network. An interchange node permits migration of an existing subarea network to APPN on a node-by-node, link-by-link basis without a need for network-wide coordination. Also, an interchange node permits session establishment across any combination of APPN and subarea networks, including SNA network interconnection (SNI) gateways.

For detailed interchange node flows, see Appendix A, "Sequence Charts."

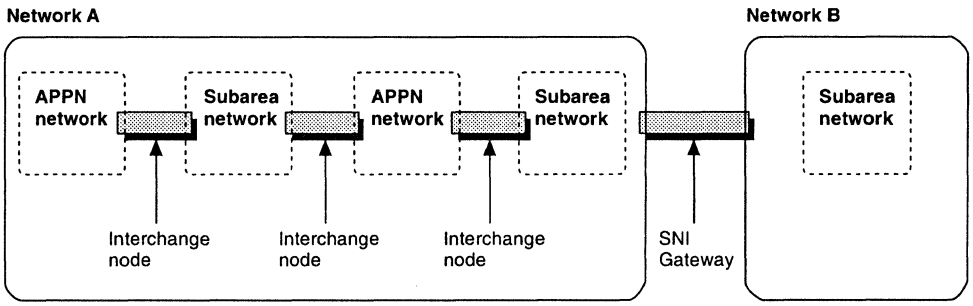


Figure 12. Interchange Nodes Interconnecting APPN and Subarea Networks

Figure 13 illustrates the protocol conversion of a Locate GDS variable between an APPN network and a subarea network.

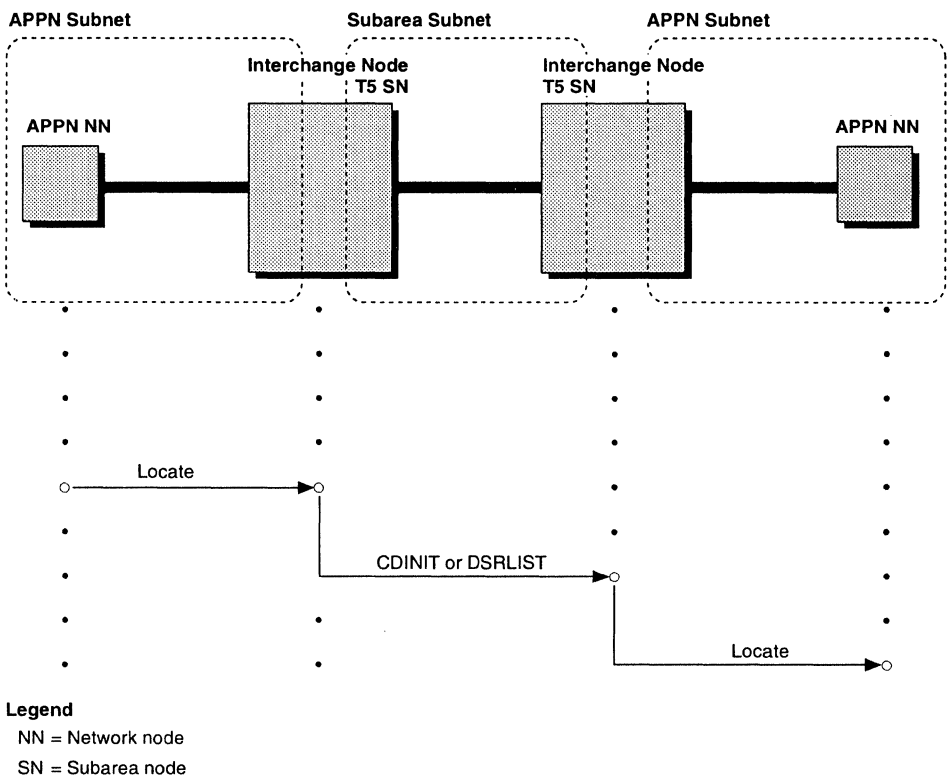


Figure 13. Simple APPN/Subarea/APPN Protocol Conversion

A subarea network can enable session establishment between two disjoint APPN networks by appearing as an APPN network node to each APPN network. The subarea network acts as a ***surrogate network node server*** in this case. In Figure 14, assume the interchange nodes providing the boundary function, Node B and Node C, are two separate T5 nodes (VTAMs), that have SSCP-SSCP connectivity with each other, but no CP-CP connectivity. Each of the nodes in one APPN network appears to its endpoint partner in the disjoint APPN network as an APPN end node that connects to the surrogate network node server.

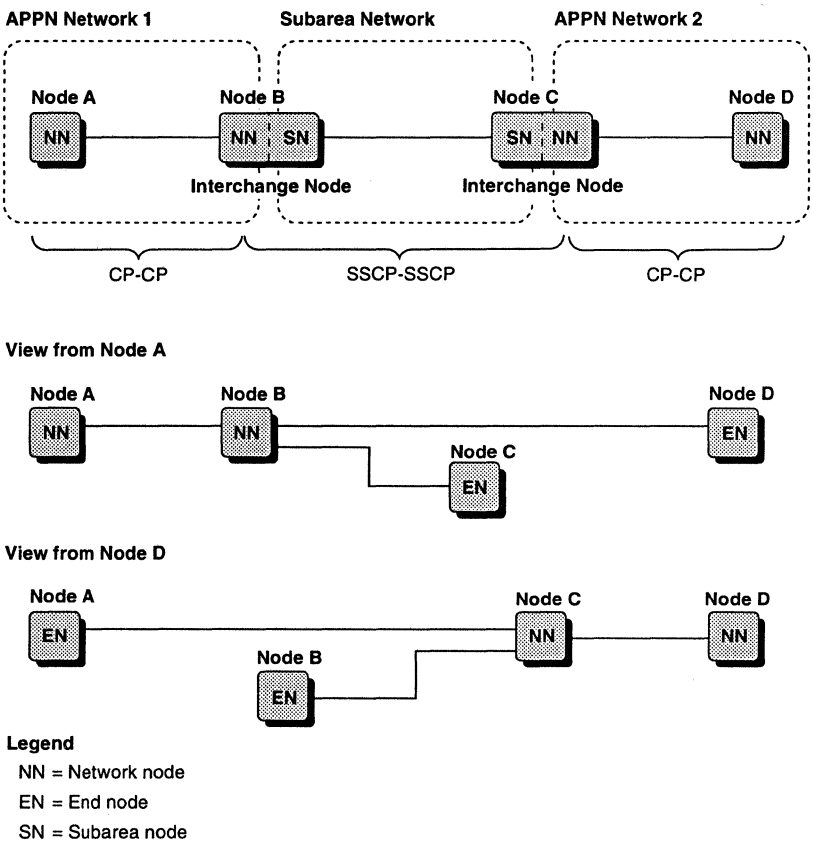


Figure 14. Surrogate Network Node Servers (Node B and Node C)

In Figure 14, the node images are as follows:

- From Node A's viewpoint, Node C and Node D appear as end nodes connected to the surrogate network node server Node B
- From Node D's viewpoint, Node A and Node B appear as end nodes connected to the surrogate network node server Node C.

Composite Network Node (CNN): A composite network node (CNN) is composed of a single T5 (VTAM) node and all the T4 (NCP) nodes that the T5 node owns, which work together to provide the appearance of a single network node to other LEN and APPN nodes to which the CNN is interconnected. The CNN can have only APPN function or have both APPN and subarea function. Existing subarea protocols are used within the CNN for communication between the T5 node and its T4 nodes. APPN protocols are used to communicate with other APPN network nodes and end nodes. The T4 node provides boundary functions for attaching other APPN nodes. The T5 node provides the CP functions and can also provide boundary function support. Figure 15 illustrates a composite network node that is part of both a subarea network and an APPN network. This composite network node can act as an interchange node. However, interchange nodes do not have to be within composite network nodes.

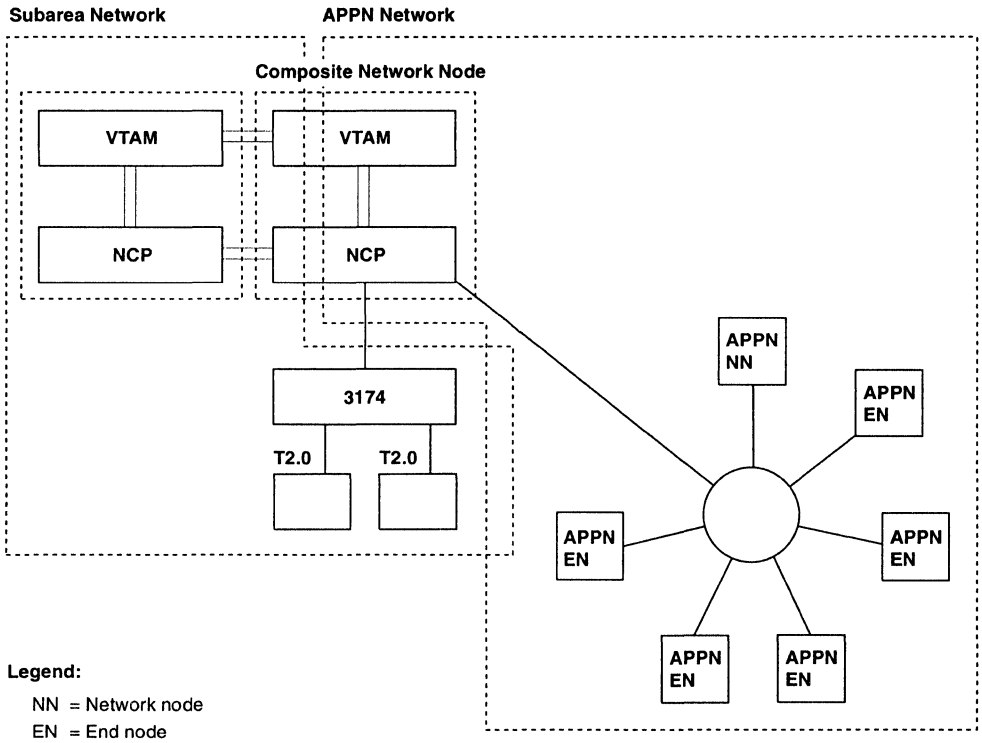


Figure 15. Composite (Interchange) Network Node with Subarea and APPN Function

Composite LEN Node: A composite LEN node is composed of a single T5 (VTAM) node and all the T4 (NCP) nodes that it owns, which work together to provide the appearance of a single LEN node to other LEN and APPN nodes to which it is interconnected. Like the CNN, it uses subarea protocols internally; it uses LEN protocols in its interactions with attached LEN or APPN nodes.

Migration Data Host: A ***migration data host*** resides on the border of an APPN network and a subarea network. It combines the function of an APPN end node with the function and role of a subarea data host. A subarea data host is a host that predominantly owns applications. The subarea data host processes the applications and spends little time on session establishment and routing. A migration data host, like a subarea data host in a subarea network, is dedicated to processing application programs and controls only local network resources. The migration data host communicates network control data by using SSCP-SSCP sessions with subarea nodes and CP-CP sessions with APPN nodes. Migration data hosts do not provide intermediate session routing, and do not interchange APPN and subarea protocols.

Links and Transmission Groups (TGs)

Adjacent nodes in a network are connected to one another by one or more links. A ***link*** includes both the link stations within the two nodes it connects and the link connection between the nodes. A ***link station*** is the hardware or software within a node that enables the node to attach to, and provide control over, a link connection. It exchanges information and control signals with its partner link station in the adjacent node. Link stations use data link control protocols to transmit data over a link connection. A ***link connection*** is the physical medium over which data is transmitted. Examples of transmission media include telephone wires, microwave

beams, fiber-optic cables, and satellite circuits. Multiple links between the same two nodes are referred to as ***parallel links***.

A ***transmission group*** (TG) may consist of one or more links between two nodes. A TG comprising two or more parallel links is called a ***multilink TG***. Multilink TGs may be defined between two type 4 subarea nodes using SDLC links. In APPN, multilink TGs are not defined, so the terms *TG* and *link* are used generally interchangeably in an APPN context. In both subarea networks and APPN networks, multiple (or parallel) TGs may connect two adjacent nodes. Data traffic is distributed dynamically over the links of a multilink TG. Multilink TGs allow a session to remain active when one link in the multilink TG fails. Parallel TGs do not generally offer this advantage, except in high-performance routing (HPR) subnetworks (discussed later), which provide nondisruptive path switching between HPR nodes, allowing sessions to remain unaffected by a TG outage.

Also, in APPN two VTAM hosts may be connected to each other via a high-performance channel, called a host-to-host channel. Host-to-host channels use a multipath channel interface; that is, multiple logical paths may be defined over a single TG.

Figure 16 shows parallel TGs connecting adjacent T4 nodes via TG2 and TG3, as well as a multilink TG connecting adjacent T4 nodes via TG1. The data traffic flowing between a pair of adjacent T4 nodes is distributed among the parallel TGs. If one of the TGs fails, the session is broken; that is, session traffic is not automatically rerouted over the other TG. On the other hand, if at least one link in a multilink TG is still operational, session traffic is not disrupted over the TG in the case of link failure, except that throughput may suffer. Similarly, a link may be restored to operation in a TG without disrupting existing sessions.

Link components and protocols are discussed in Chapter 2, "Link and Node Components."

SNA Network Configurations

SNA defines the following network configurations:

- A hierarchical network consisting of subarea nodes and peripheral nodes
- A peer-oriented network consisting of APPN and LEN nodes
- A mixed network that combines one or more hierarchical subnets with one or more peer-oriented subnets.

Hierarchical Network Configurations

The organization of a hierarchical network structure is determined by the way control of network services is maintained. Host nodes containing SSCPs are responsible for overall control of communication in the hierarchical network.

A hierarchical network might include LEN or APPN nodes that are attached as peripheral nodes. These nodes can communicate with each other through a subarea network if the boundary nodes to which they are attached support the basic SSCP-independent LU-LU protocols needed for such peer interactions.

Subareas

A ***subarea*** consists of one subarea node and the peripheral nodes that are attached to that subarea node. The concept of a subarea applies only to subarea networks and composite networks. The network configuration in Figure 16 contains five subarea nodes and seven peripheral nodes. Because each subarea node and its attached peripheral nodes constitute a subarea, this configuration contains five subareas.

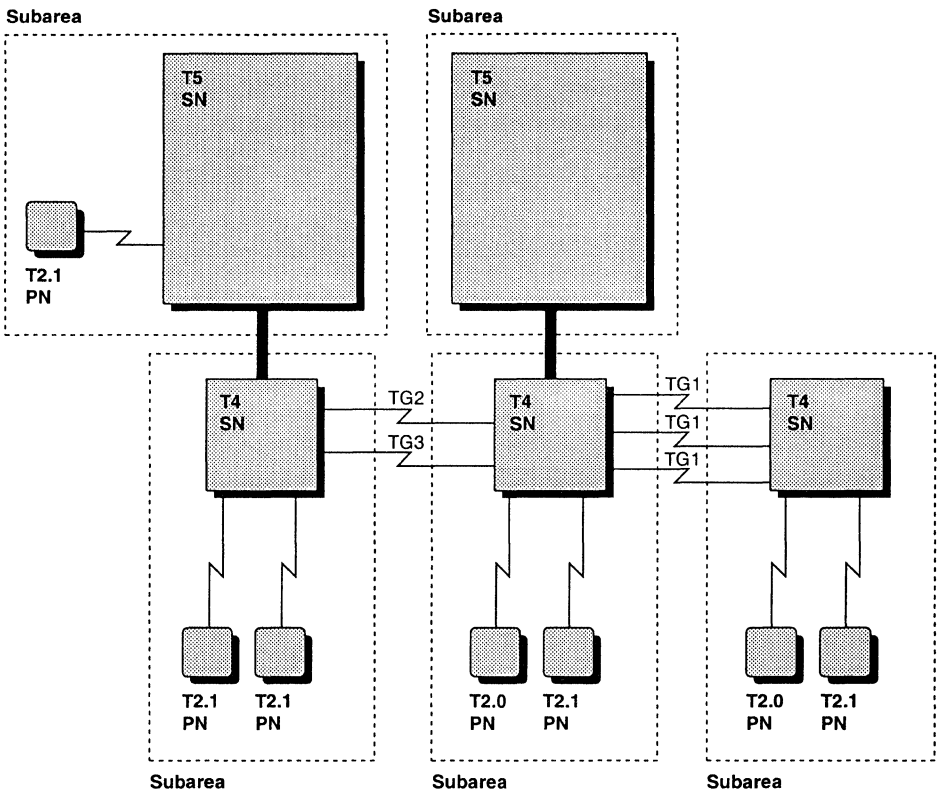


Figure 16. Subareas

Domains in a Subarea Network

A ***domain*** is an area of control. The concept of a domain within a subarea network differs from that within an APPN network. Within a subarea network, a domain is that portion of the network managed by the control point in a T5 subarea node. The control point in a T5 subarea node is called a ***system services control point*** (SSCP).

When a subarea network has only one T5 node, that node must manage all of the network resources. A subarea network that contains only one T5 node is a ***single-domain subarea network***. When there are multiple T5 nodes in the network, each T5 node may control a portion of the network resources. A subarea network that contains more than one T5 node is a ***multiple-domain subarea network***. In a multiple-domain subarea network, the control of some resources can be shared between SSCPs. Some resources can be shared serially and some concurrently. For more information on resource sharing in subarea networks, see "Defining Shared Control of Resources in Subarea Networks" in Chapter 4, "Defining Network Resources." Figure 17 illustrates two domains, **A** and **B**, joined by direct-attached T4 nodes to form a multiple-domain subarea network.

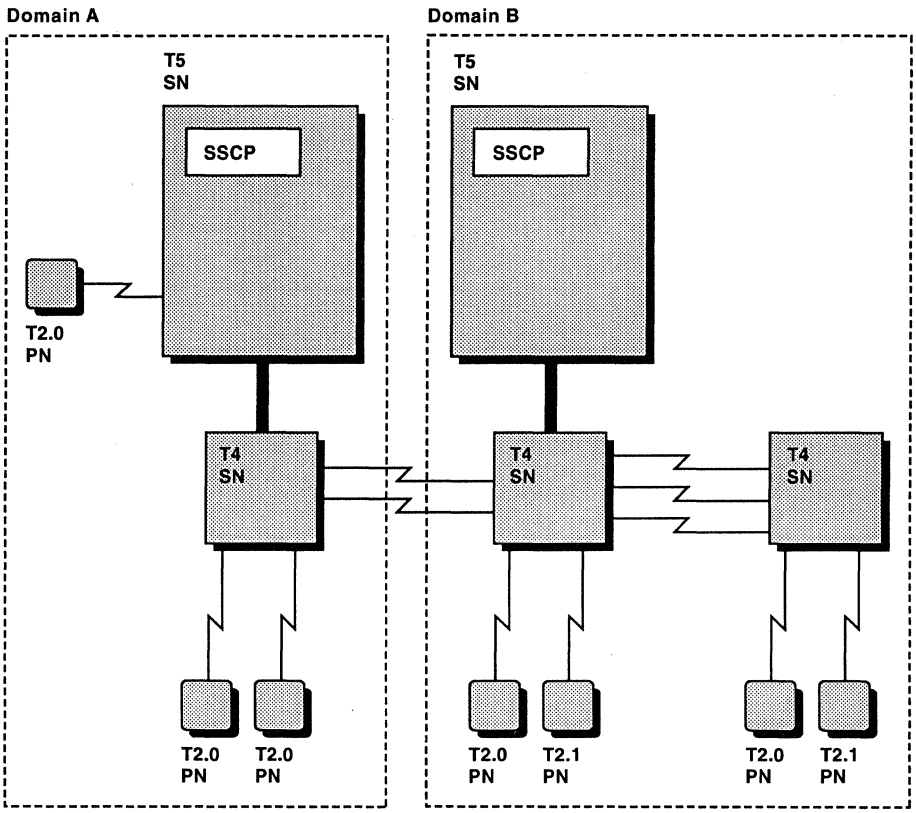


Figure 17. Domains in a Subarea Network

Peer-Oriented Network Configurations

An APPN network constitutes a peer-oriented SNA network. All APPN nodes are considered to be peers and do not rely on other nodes to control communication in the network the way a subarea node controls communication between peripheral nodes. There is, however, a measure of hierarchical control because a network node server provides certain network services to its attached end nodes. The difference is that, in APPN networks, hierarchical control is not determined by product or processor type as it is in subarea networks, where only large host processors contain SSCPs and node types generally reflect product types.

Domains in an APPN Network

The domain of a node in an APPN network is that portion of the network served by the control point in the node. The control point in an APPN node is called simply a ***control point*** (CP). An end node control point's domain consists solely of its local resources. It is included within the domain of its network node server. A network node control point's domain includes the resources in the network node and in any *client* end nodes (nodes for which the network node is acting as the network node server) attached directly to the network node.

APPN networks are by definition multiple-domain networks. Figure 18 illustrates an APPN network containing four network node domains, **A**, **B**, **C**, and **D**. The domains of the end nodes are included within the domains of their respective network node servers.

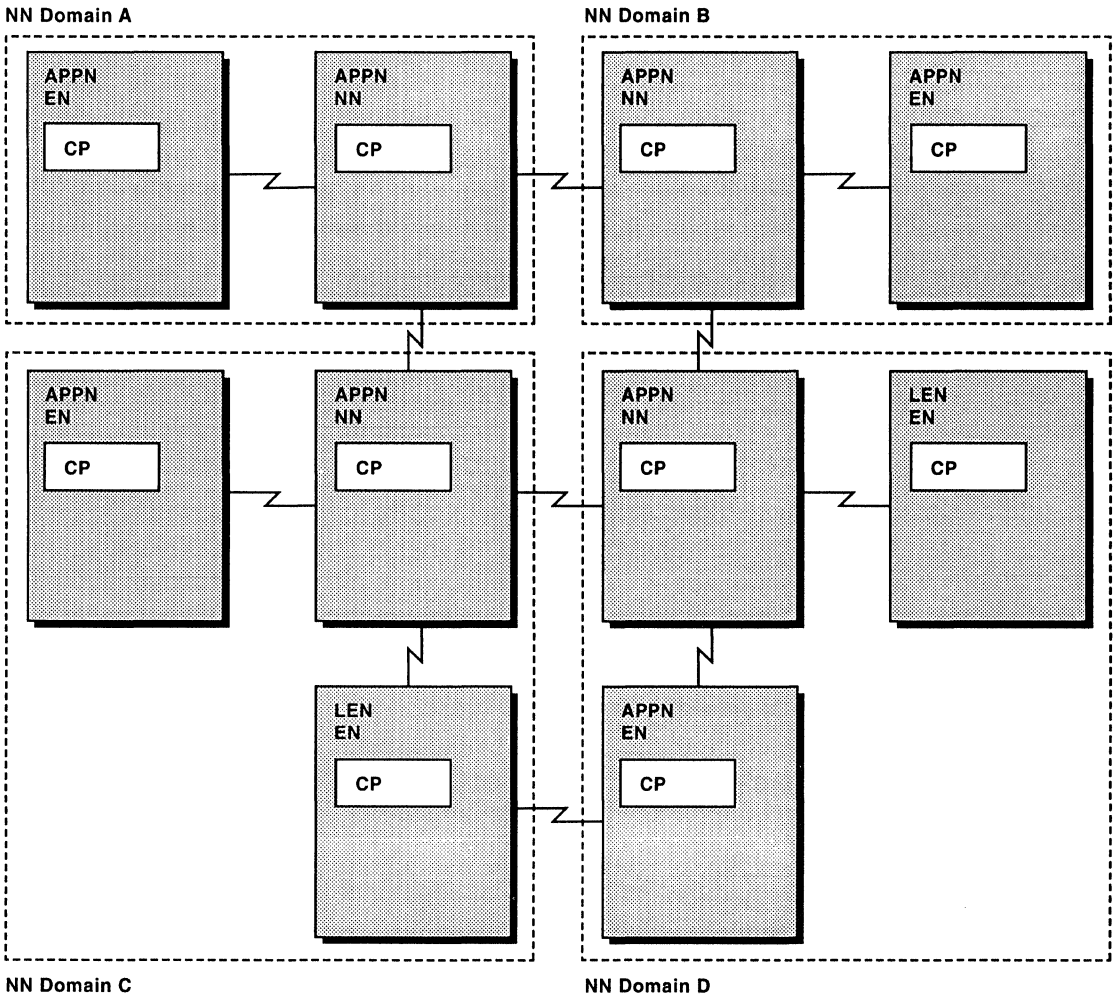


Figure 18. Network Node Domains in an APPN Network

The Transport Network and Network Accessible Units

Node and link components exhibit SNA layering through the functions they perform. The lower three architectural layers comprise the transport network, and certain components within the upper four layers are referred to as network accessible units.

The Transport Network

Node and link components within the lower three architectural layers—the physical control, data link control, and path control layers—are collectively referred to as the ***transport network***. The physical control layer includes both the data communication equipment within nodes and the physical link connections between them. Node components within the data link control layer activate and deactivate links on command from their control points, and they manage link-level data flow. Node components within the path control layer perform routing and congestion control. Together, the distributed components of these three lower layers transport data through the network on behalf of network accessible units.

Network Accessible Units

Certain components within the upper four architectural layers of a node use transport network services to establish temporary, logical connections with one another called ***sessions***. Sessions can be established between two components residing in different nodes, or between components within the same node. Components that can establish sessions are referred to as ***network accessible units*** (NAUs).[1] NAUs in session with one another are referred to as ***session partners***.

A set of physical connections consisting of the links and intermediate nodes between session partners constitutes a ***route*** for the session. During session initiation procedures, a route is selected for the session. Once a session is initiated, the session traffic flows over the same route for the duration of the session.

The three fundamental kinds of network accessible units are physical units, logical units, and control points. ***Physical units*** (PUs) perform local node functions such as activating and deactivating links to adjacent nodes. PUs exist only in nodes within subarea networks. (In APPN networks, these functions are performed by control points.) To perform its functions, a PU must exchange ***control data*** with its controlling system services control point (SSCP) over an ***SSCP-PU session*** initiated by the SSCP.

Logical units (LUs) provide network access for end-users by helping the end-users send and receive data over the network. Nodes in both subarea and APPN networks contain LUs. LUs send and receive control data and ***end-user data*** over the ***LU-LU sessions*** established between them.

In a subarea network, an LU residing in a peripheral node is classified as either SSCP-dependent or SSCP-independent, depending on the protocols it uses for LU-LU session initiation. An ***SSCP-dependent LU***, or simply ***dependent LU***, sends a session-initiation request to its controlling SSCP over an ***SSCP-LU session***. The LU is dependent on the SSCP to mediate the session initiation with the partner LU and requires an SSCP-LU session for that mediation. The session is activated when the LU receives a session-activation request from the partner LU. Dependent LUs reside in both T2.0 and T2.1 nodes.

An ***SSCP-independent LU***, or simply ***independent LU***, sends a session-activation request directly to the partner LU. It does not interact with an SSCP to mediate an LU-LU session and does not require an SSCP-LU session. An independent LU may reside in a T2.1 node, but not in a T2.0 node. Independent LU protocols are also referred to as ***peer-session protocols***.

In APPN networks, independent LUs send session-activation requests directly to their partners, and all nodes support peer-session protocols. Two LEN nodes directly attached to one another also support peer-session protocols.

Control points (CPs) provide network control functions that include managing the resources in their domains and monitoring and reporting on the status of those resources. The functions of node CPs in subarea networks differ greatly from those in APPN networks. In subarea networks, SSCPs control the PUs and

---

[1] NAUs are also known as "network *addressable* units" in earlier SNA literature. In the APPN routing context, unlike the subarea context, NAUs are identified by *name* rather than by *address*. The terminology has therefore been refined to reflect this nuance but to preserve the familiar acronym.

dependent LUs in their domains by exchanging control data with them over SSCP-PU and SSCP-LU sessions respectively. They may also initiate ***SSCP-SSCP sessions*** for the control of *cross-domain* sessions (sessions that cross domain boundaries). In APPN networks, a CP does not initiate sessions with the LUs in its domain. It does, however, initiate sessions with adjacent CPs, called ***CP-CP sessions***, in order to exchange control data for its routing and directory services.

Link and node functions and the components that perform them are discussed in Chapter 2, "Link and Node Components."

Interconnecting Nodes

The collection of interconnected links and nodes in a network is the network's ***configuration***. Network configurations must grow in response to increases in demand for network services. The roles that a network's nodes and links are capable of determine the variety of possible network configurations. Nodes that are capable only of hierarchical roles must be configured into hierarchical networks (see Figure 17 on page 26). T2.0 nodes, for example, must appear as peripheral nodes attached to subarea (T4 and T5) nodes acting as boundary nodes. As discussed earlier, T2.1 and T5 nodes can take on a much larger set of network roles.

A network of APPN nodes, such as that shown in Figure 18 on page 27, can have a virtually arbitrary topology. An APPN end node can be configured in the same way as a LEN node, but it includes support for CP-CP sessions with a network node server. In this way, they can dynamically register their LUs with the APPN network via their server, provide local connectivity information to their server, and obtain the location of, and best route to, a session partner. An APPN network node supplies routing and directory services to its client LEN and APPN end nodes through its cooperation with other APPN network nodes. These services enable APPN networks to support LU-LU sessions and transport data without reliance on the centralized services provided by large host processors. Configuration flexibility, together with the implementation of APPN on small and midrange processors, and also on mainframes (VTAM), allows a wide range of size and cost options for APPN networks.

Interconnecting Networks

Independent SNA networks might need to be interconnected for any of several possible reasons:

- Two enterprises have merged, creating the need for terminals in one SNA network to access the application programs in another.
- Within a large enterprise, multiple SNA networks have evolved, each with different characteristics. Some, for example, might be subarea networks, and some APPN networks. These may be merged into a single logical network while maintaining the autonomy of each.
- A token-ring local area network (LAN) has been installed to offload an existing host-based network, and an interconnection is needed between them.
- The need to share resources between geographically separate token-ring LANs requires that they be interconnected.

- An interconnection is needed to a network that uses X.25 interface protocols, such as a public X.25 packet-switched data network.

Two subarea networks can be interconnected through ***SNA Network Interconnection*** (SNI). SNI is an SNA-defined architecture that enables independent subarea networks to be interconnected through a ***gateway***. SNI enables partner nodes to communicate through the gateway without either one knowing that the other resides in a different subarea network. This communication is possible even though each network retains its own independent configuration, definition, and management. An SNI gateway consists of a gateway node and one or more gateway SSCPs. A ***gateway node*** is a T4 node containing a ***gateway function*** that performs cross-network address translation for the gateway. A ***gateway SSCP*** exists in a T5 node in one of the interconnected networks. It interacts with the gateway node and other gateway SSCPs in the same gateway to help initiate cross-network LU-LU sessions through the gateway. (For more information on subarea network interconnection, see "Defining Gateways in Subarea Networks" in Chapter 4, "Defining Network Resources.")

Two APPN networks can be interconnected by a border node, either extended or peripheral, which allows networks with different topology subnets to establish CP-CP sessions with each other. The topology information of each subnet is not shared. A peripheral border node can appear only in the endpoint subnet of a network configuration. Depending on what it is attached to in the other subnet, the peripheral border node presents either an APPN end node image or a network node image to the attached subnet. Topology information is not exchanged between peripheral border nodes. For more information on peripheral border nodes, see "Defining APPN Subnets Containing Peripheral Border Nodes" in Chapter 4, "Defining Network Resources." An extended border node can appear in either an endpoint subnet or an intermediate subnet of a network configuration. Two extended border nodes residing in two different topology subnets are seen as network nodes to each other. Therefore, they must specify during XID exchange that topology information is not to be exchanged. Extended border nodes appear to the peripheral subnets as APPN end nodes. For more information on extended border nodes, see "Defining Subnets Containing Extended Border Nodes" in Chapter 4, "Defining Network Resources."

(For more information on APPN network interconnection, see "Calculating Routes Across Interconnected APPN Networks Containing Peripheral Border Nodes" in Chapter 6, "Establishing Routes Through the Network.")

An APPN network can be connected to a subarea network through a boundary node in different ways depending on the protocol capabilities within the subarea network. A ***boundary node*** is a T4 or T5 (VTAM) node containing a boundary function. An APPN network node can attach directly to a boundary node using either LEN or APPN protocols depending on the support level of the T5 node owning the boundary node. All of the APPN network's resources appear to the LEN boundary node to reside in a LEN peripheral node. In this case, the APPN network node supports the LEN boundary node in the same manner as it supports a LEN end node. All of the subarea network's resources appear to the APPN network node to reside in the LEN boundary node.

Using APPN protocols, an APPN network can be connected to a subarea network through an interchange node. The interchange node participates as a network node in the APPN network. The APPN network attached to an interchange node

appears to the subarea network as a LEN peripheral node. The subarea network interconnected to the interchange node appears to the APPN network as APPN end nodes served by the interchange node. The interchange node itself maintains its subarea appearance to other subarea nodes and maintains its APPN appearance to other APPN nodes.

Interconnecting Session Stages

Messages transmitted on a session carry *addresses* in their headers. The addresses enable ***intermediate routing nodes*** (the nodes situated along a session route between the two partner nodes) to correlate a message with its appropriate session and to route the message accordingly.

As session data traverses a route, not only the endpoints (called ***half-sessions***) but also certain components along the route called ***session connectors*** provide flow control and address translation functions for the session. The parts of a session that are delimited by half-sessions and/or session connectors are called ***session stages***. The three points in an SNA network where session connection occurs are at the boundary function, the gateway function, and the intermediate session routing function.

- The ***boundary function*** resides in a T4 or T5 node acting as a boundary node. It translates between the network addresses used by a subarea node and the local addresses used by a peripheral node. (For more information on the boundary function, see "The Boundary Function Component" in Chapter 6, "Establishing Routes Through the Network.")
- The ***gateway function*** resides in a T4 node acting as a gateway node. It translates between the network addresses used by one network and the network addresses used by another. (For more information on the gateway function, see "Defining Gateways in Subarea Networks" in Chapter 4, "Defining Network Resources.")
- The ***intermediate session routing function*** resides in an APPN network node acting as an intermediate session routing node. It translates between the session address used by one session stage and the session address used by another. (For more information on the intermediate session routing function, see "The Intermediate Session Routing Component" in Chapter 6, "Establishing Routes Through the Network.")

The boundary function, the gateway function, and the intermediate session routing function all have the same basic structure (see Figure 19). Each has a ***session connector manager***, which creates and destroys session connectors, and multiple ***session connectors***, each of which performs address translation for an individual session. In the boundary function and intermediate session routing function, the session connector also provides session flow control. The gateway function session connector need not provide such session flow control, because it simply joins together the two ***virtual routes*** (discussed later) that terminate in its node (one going to each network) and relies on the comparable flow control supplied by its underlying path control components.

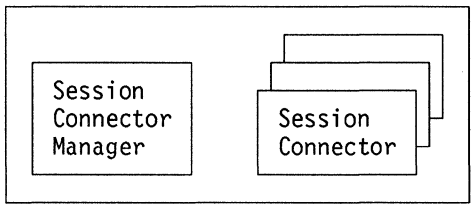
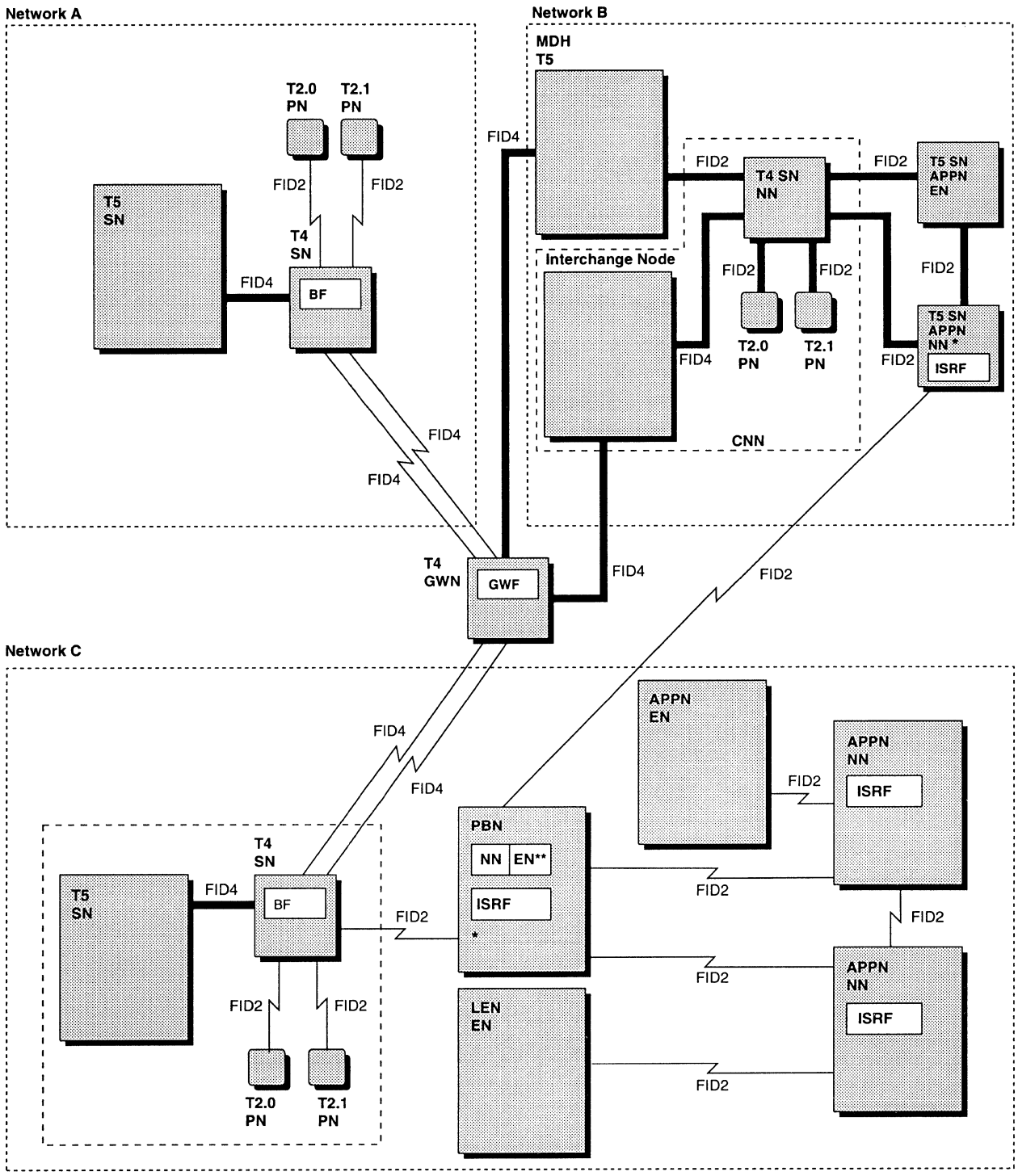


Figure 19. Session-Interconnection Function Structure

The flexibility provided by SNA in node, network, and session interconnection allows a wide array of node and network configuration options. One possible configuration of interconnecting networks is shown in Figure 20.



Legend:

BF = Boundary Function
CNN = Composite Network Node
GWF = Gateway Function
GWN = Gateway Node
FID2 = Format Identifier 2
FID4 = Format Identifier 4
ISRF = Intermediate Session Routing Function
MDH = Migration Data Host
PBN = Peripheral Boder Node
* Viewed as a T2.1 PN by the T4 SN
** Viewed as an APPN EN by the partner NN in Network B

Figure 20. Interconnecting Networks

Combined HPR/APPN Network Operation

HPR nodes use the basic APPN CP-CP sessions unchanged. HPR network nodes and basic APPN network nodes share a common topology database. Nodes in the APPN network see the nodes in the HPR portion of the network as basic APPN nodes. Nodes in the HPR portion of the network can distinguish between the APPN and HPR TGs and nodes. For more information on the topology database, see "Topology Database" in Chapter 5, "Activating the Network."

Figure 21 illustrates an APPN network which contains nodes with the HPR function (comprising the HPR portion of the APPN network). The endpoint of a session can be either in the basic APPN subnet or in the HPR portion of the APPN subnet. For example, NN2 may establish an LU-LU session with NN6; or NN2 may establish an LU-LU session with NN4 in the HPR portion of the APPN subnet.

The HPR portion of the APPN subnet may use the high-performance routing and nondisruptive path switch features that comprise HPR. These two features are accomplished via rapid-transport protocol (RTP) connections, automatic network routing (ANR), and adaptive rate-based (ARB) flow/congestion control, all described later.

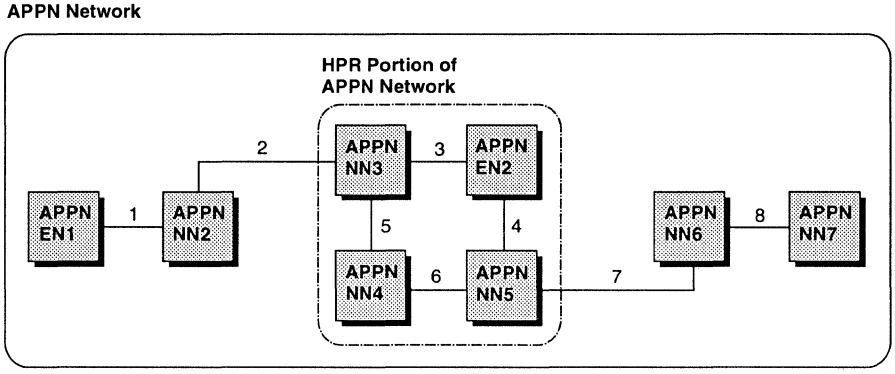


Figure 21. HPR/APPN Network

Requirements for Data Transport

Formatting Data

The ***request/response unit*** (RU) is the unit of data upon which other data formats are built. An RU can contain either end-user data or control data. As an RU flows through the network on its way from the sending NAU to the receiving NAU, it is handled by node components within different SNA layers. To enable peer communication between layers, RUs are encapsulated between data fields called *headers* and *trailers.* Network components use headers and trailers for such purposes as addressing, error detection, and protocol control. Regardless of the number of headers and trailers attached to the RU, the string of data transmitted as a unit is referred to generically as a ***message unit.***

Within the RU, data can be formatted further by either node components or by the end-user. Data is often transmitted in SNA-defined data streams. ***Data streams*** consist of end-user data formatted for presentation to the destination through the use of control characters and control fields contained in the data.

The formats of message units and control fields, and of data streams, are discussed in Chapter 3, "Data Formats."

Defining Network Resources

Before a network can be activated, its resources must be defined to the nodes within it. The nature of the definition process depends on the type of node to which the resources are being defined. Resources are defined to subarea (T5 and T4) nodes through system generations and network operator commands. System generations define resources in the domains of T5 nodes and in the subareas of T4 nodes. Not all resources, however, need be defined initially to T4 and T5 nodes. Some resources can be defined dynamically by network operators through ***dynamic reconfiguration***. Resource definition for APPN or LEN nodes is entirely dynamic. Node operators can define resources during node start-up or as needed. In addition, APPN network nodes can dynamically learn of network resources through ***resource registration*** by client end nodes and through ***search protocols***.

For more information on defining network resources to subarea and APPN nodes, see Chapter 4, "Defining Network Resources."

Activating a Network

Network activation proceeds differently in subarea and APPN networks. The activation of subarea networks involves activating groups of nodes and links within a domain by operator command. The process is hierarchical, proceeding outward from the T5 nodes to the T4 and peripheral nodes. Predefined directories established during system generation control the activation of subarea networks. Not all nodes, however, need to be activated at once. Peripheral nodes on switched connections, for example, can join a subarea network after the network has been activated.

APPN network activation is more distributed in nature. Each node activates its resources and the links to its adjacent nodes independently. A network node dynamically learns of the addition of a new network node to the network through ***topology database update*** (TDU) messages received from an adjacent network node. The receiving network node then sends that information on to its other adjacent network nodes. The network topology information is thus propagated throughout the APPN network.

Both subarea and APPN network activation are discussed in Chapter 5, "Activating the Network."

Establishing Routes

Like network activation, the process of establishing session routes differs between subarea and APPN networks. Routes within subarea networks are statically predefined and stored within subarea routing tables. Subarea routes include both explicit routes and virtual routes. An ***explicit route*** defines the nodes and links along a physical path between two subareas. Different explicit routes can, therefore, have different reliability and performance characteristics. A ***virtual route*** is a logical connection between endpoints that is built on two explicit routes (one for each direc-

tion). Virtual routes incorporate the characteristic of ***transmission priority***. Multiple virtual routes (with different priorities) can share the same explicit route. A virtual route takes on the physical characteristics of its underlying explicit route.

In a subarea network, the combination of the explicit and virtual route characteristics constitutes a ***class of service***. When an LU within a subarea network requests a session, it requests a particular class of service. The T5 node that is servicing the request then selects a list of virtual routes that provide the desired class of service and returns the list to the path control layer in the requesting LU. Path control then uses the list to select the first virtual route whose underlying explicit route is active.

APPN networks, like subarea networks, use the concept of class of service when determining routes. As in a subarea network, the requesting LU-LU session partner in an APPN network requests a particular class of service for the session. The requesting partner's network node server then determines a route for the session. Unlike subarea networks, however, APPN networks do not include the concepts of explicit and virtual routes. Because the locations of APPN nodes are not predefined, neither are the routes between them. Both the locations of APPN nodes and the routes between them are dynamically determined at the time of session initiation.

The process of establishing a route in an APPN network requires the preliminary step of locating the session partner. This is one of the services of the requesting partner's network node server. The network node may already have the partner's location recorded within its local directory. If so, the network node sends a ***directed search*** message to the target partner. If, however, the network node cannot determine the partner's location from its local directory, and there is a CDS located in the subnet, a request is sent to the CDS to determine whether the partner's location is registered or cached in the CDS. If the partner's location is not in the CDS, the CDS queries the alternate CDSs (if any). If they do not have the requested information, the first CDS initiates a broadcast search. Only if no CDS exists does the basic network-node-initiated ***broadcast search*** occur. The requesting session partner is eventually notified as to whether or not the target partner can be located. Once the target partner has been located, the route for the session is determined by the requesting partner's network node server.

Both subarea and APPN route establishment are discussed in Chapter 6, "Establishing Routes Through the Network."

Controlling Congestion

When messages enter a network faster than they can be transmitted, congestion may result. Many factors can impede the transport of data through a network. Examples include insufficient link bandwidth, high link error rates, and limited buffer space within session partners or intermediate nodes. To optimize network throughput, SNA employs two categories of performance-related protocols: message repackaging and message pacing. ***Message repackaging*** is the combining of messages into larger units, or the subdividing of messages into smaller units, for transmission. It prevents congestion by improving the efficiency of nodes and links in processing messages. ***Message pacing*** is a protocol by which a receiving node controls the rate at which it receives message units. It prevents congestion by controlling the overall number of message units in the network. In subarea networks, pacing is performed on both the session level and the virtual

route level. In APPN networks, pacing is performed between adjacent nodes on each session stage.

Message repackaging and message pacing are discussed in Chapter 7, "Controlling Congestion in the Network."

Session Protocols

Network accessible units establish sessions for transporting control or end-user data. The two partner NAUs can reside in the same node, in the same domain, in different domains, or in different networks connected by an SNI gateway or an APPN border node. One NAU activates a session by sending a session activation request to the other. When the partner NAUs are logical units, the session activation request is referred to as a ***Bind Session*** or ***BIND*** request.

To ensure that partner NAUs use the same protocols during a session, the session-activation request specifies the protocol options to be observed. Such protocols determine response requirements on the part of the receiving NAU, message size requirements, and the permissibility of having multiple requests in transit simultaneously between NAUs.

SNA supports several LU types and many protocols to support LU-LU sessions. The growth of distributed processing has created, however, increased demand for peer-oriented LU-to-LU communication. One LU type, type 6.2, provides a set of protocols that answer this need. ***LU 6.2***, also referred to as ***Advanced Program-to-Program Communication*** (APPC), provides a set of functions required for all kinds of connectivity requirements, including program-to-program, program-to-device, and device-to-device communication using processors of varying power and generality. LU 6.2 functions are invoked by application programs through a well-defined ***protocol boundary*** that provides a universal foundation for product implementations. CPI-C allows easy application development without having to learn the details of the LU 6.2 API for each product platform that is used.

Session protocols, LU-LU session activation, and LU 6.2 protocols are discussed in Chapter 8, "Transporting Data Through the Network."

Transaction Services

The top layer of SNA, the transaction services layer, maintains the architectural definitions for service transaction programs. A ***transaction program*** processes transactions in an SNA network. A ***transaction*** involves an exchange of messages, with a specified format that accomplishes a specific task or job. The entry of a customer's deposit that updates the customer's balance is one example of a transaction. Another example is the process of verifying a check and returning an authorization to accept the check as legal tender.

There are two kinds of transaction programs: application transaction programs and service transaction programs. An ***application transaction program*** is an end-user of an SNA network. It accesses the network through an LU and processes end-user transactions. A ***service transaction program*** is an IBM-supplied transaction program that the architecture defines. It exists in an LU or CP and provides common network services to other network components or to end-users.

The growth of distributed systems has created a demand for specific transaction services for distributed system support. Several transaction services architectures are defined by IBM in response to this demand. They include Distributed Data Management (DDM), SNA/Distribution Services (SNA/DS), SNA/File Services (SNA/FS), and Document Interchange Architecture (DIA). These architectures define services for the interchange of user data among distributed systems. DDM, SNA/DS, SNA/FS, and DIA are discussed in Chapter 9, "Transaction Services."

A higher-level architecture, ***Systems Application Architecture*** (SAA), provides a foundation for structuring application software development. SAA defines software interfaces, conventions, and protocols to provide a framework for the development of consistent applications by IBM, IBM customers, or third-party vendors. ***Consistent applications*** are those with an appearance and behavior that remain constant across IBM product lines. Their value lies in the ease with which they can be developed by programming personnel, learned and used by end-users, and ported across multiple systems in distributed processing environments.

SAA builds on IBM's approach for structuring product and software management. The software foundation used consistently in all SAA environments includes application enablers, communications, and system control programs. ***Application enablers*** are programming products used to write applications. They include languages, application generators, query and report writers, database managers, and presentation and dialog managers. ***Communications*** are functions that connect applications, devices, systems, and networks. ***System control programs*** are products providing data and storage management, job entry and security, utilities, and system services. They create the execution environments for applications.

For additional information on SAA, refer to *Systems Application Architecture: An Overview.*

Network Management

Network management is the planning, organizing, monitoring, and controlling of a communication network. The architecture provided to assist in network management of SNA networks is called ***SNA/Management Services*** (SNA/MS). SNA/MS is implemented as a set of functions and services within the node. It is designed to capture and use information needed for effective management.

SNA/MS functions are divided into four categories: problem management, performance and accounting management, configuration management, and change management. Each category has a number of functions, but each node is not required to implement the full set of functions in all categories. Each node has a mandatory set of functions that it must implement and an optional set that it can implement, if needed. The node's role in the network determines which functions are mandatory and which functions are optional.

Two MS roles are defined for SNA nodes: entry points and focal points. An ***entry point*** is a node that provides distributed network management support. A ***focal point*** is an entry point that provides centralized network management for itself and other entry points. Entry points send MS data to focal points for centralized reporting and control.

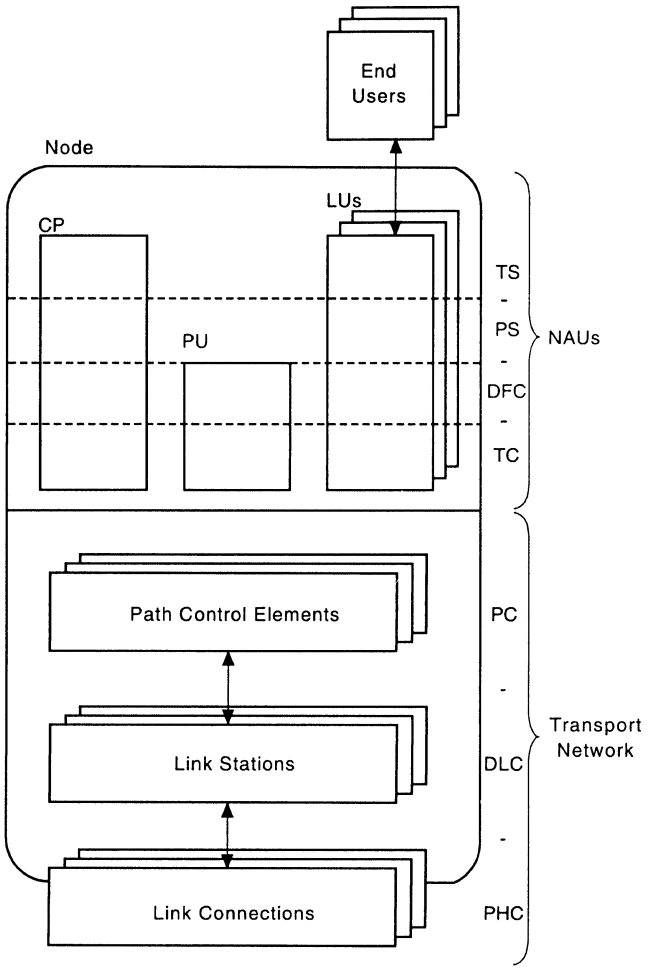
SNA/Management Services is discussed in Chapter 10, "Managing an SNA Network."

Chapter 2. Link and Node Components

This chapter provides information on link and node components, the major functions they perform, and the SNA layers in which they reside.

SNA Components in Links and Nodes

SNA link and node components implement the functions of the seven architectural layers. Figure 22 illustrates the major components in a generic node and the layers in which they reside.



Legend:

CP = Control Point
DFC = Data Flow Control
DLC = Data Link Control
LU = Logical Unit
NAU = Network Accessible Unit
PC = Path Control
PHC = Physical Control
PS = Presentation Services
PU = Physical Unit
TC = Transmission Control
TS = Transaction Services

Figure 22. SNA Components in Links and Nodes

Link and node components fall into two broad categories: the transport network and network accessible units (NAUs).

The Transport Network

The ***transport network*** consists of the lower three layers of the architecture: the physical control, data link control, and path control layers. Together, these three layers control the links and routes over which data is transported in the network.

Physical Control

Recall that a ***link*** includes the link stations within two adjacent nodes, plus the link connection between the nodes. A link thus includes both software and hardware components. Figure 23 illustrates the components of a link.

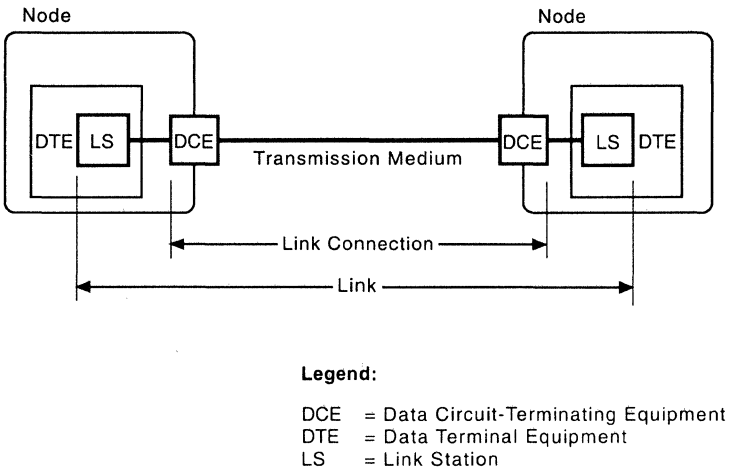


Figure 23. Components of a Link

At the physical control layer, links employ hardware components that use a particular medium to transmit data. ***Data terminal equipment*** (DTE) is hardware that communicates over links and resides in processors, controllers, or terminals. ***Data circuit-terminating equipment*** (DCE) is hardware that establishes, maintains, and terminates connections. Depending on the implementation, the DCE might or might not be part of the same physical package as the DTE. The ***link connection*** is the part of the link that includes the DCEs and the transmission medium between them, but not the link stations.

Data Link Control

Components within the data link control layer are called data link control instances. A ***data link control instance*** (DLC instance) transmits data across a link between adjacent nodes. Every node contains one DLC instance for each link attached to that node.

Each DLC instance consists of a DLC manager and a DLC element. The ***DLC manager*** performs functions not related to session traffic routing, such as exchanging management services data with the control point, while the ***DLC element*** transmits message units that flow on sessions.

The DLC manager performs the following functions:

- Activates and deactivates the DLC element
- Activates and deactivates links
- Notifies the control point whenever a station becomes operative or inoperative.

The DLC element performs the following functions:

- Transfers data to and from the physical medium
- Exchanges data traffic with adjacent DLC elements.

SNA identifies a data link control instance as a ***link station***. Different pairs of link stations can share a single link connection, or transmission medium. Each pair, together with their link connection, constitutes a link. Therefore, multiple links can be defined on one underlying link connection. Within a node, each link is identified with a specific adjacent link station to which it is attached over the link connection. The control points in two adjacent nodes activate, control, and deactivate a link through the link stations associated with that link. Figure 24 highlights the link stations in a configuration of nodes.

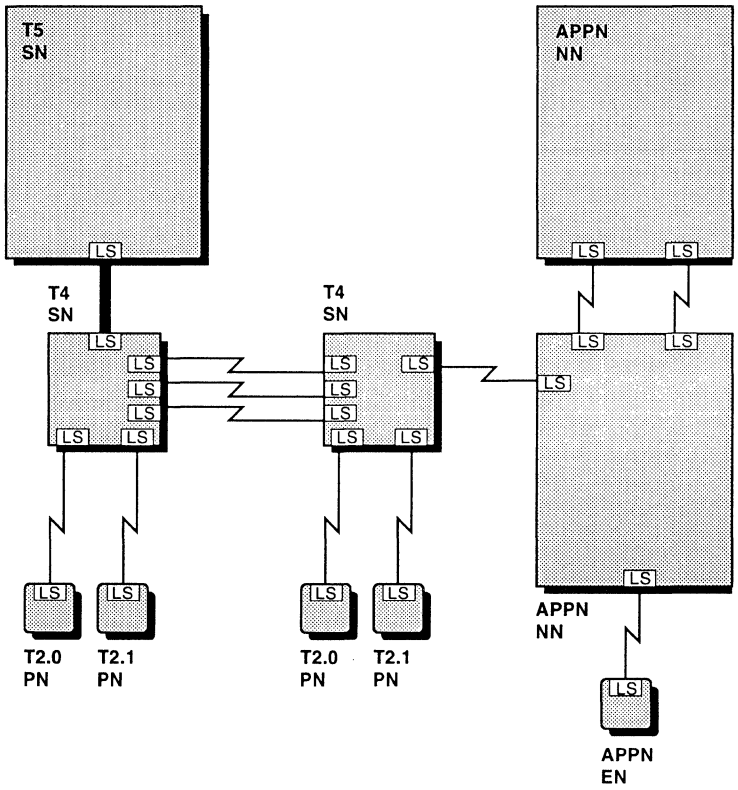


Figure 24. Link Stations

A link station performs the following functions:

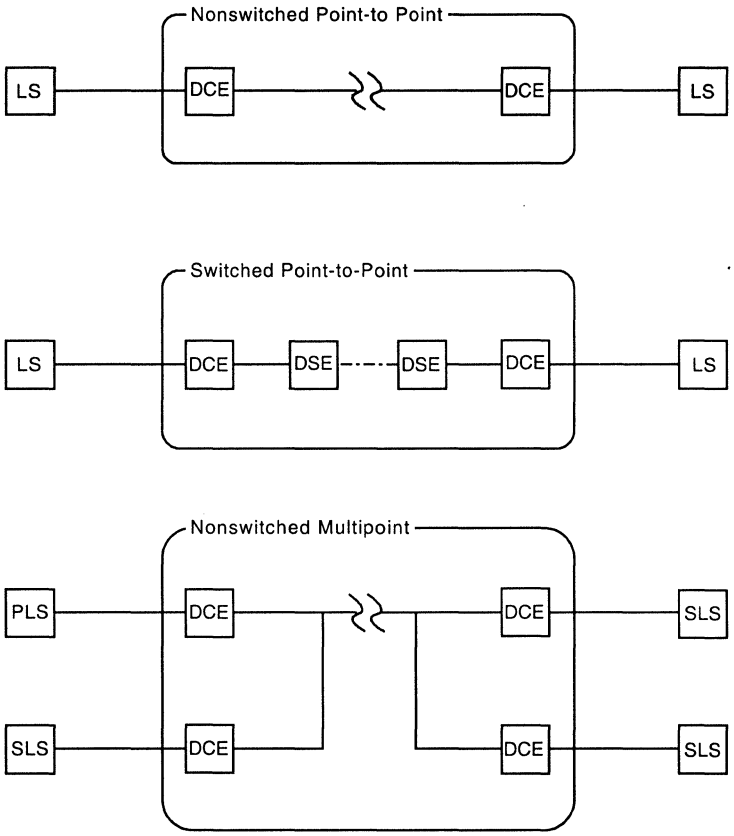
- Acts upon requests received from its control point to activate and deactivate a link
- Notifies its control point whenever its adjacent link station or link connection becomes operative or inoperative
- Exchanges data traffic with its adjacent link station across the physical medium
- Manages link-level flow
- Manages error recovery procedures for transmission errors.

SNA networks support several different kinds of links. Different link types are distinguished by the hardware they support and the data link control protocols they use to transmit data over the link connection. The primary link types supported by

SNA include synchronous data link control links, System/370 data channels, token-ring link connections, and X.25 links.

Synchronous Data Link Control Links

Synchronous data link control (SDLC) links transmit data in a code-transparent, serial-by-bit fashion. Within a transmitted information packet, control information is contained in positional bit sequences rather than in control bytes. For this reason, SDLC is referred to as a *bit-oriented* protocol. An SDLC link can have one of the basic configurations shown in Figure 25.



Legend:

DCE	= Data Circuit-Terminating Equipment
DSE	= Data Switching Exchange
LS	= Link Station
PLS	= Primary Link Station
SLS	= Secondary Link Station

Figure 25. SDLC Link Connection Configurations

In a ***nonswitched configuration***, the link connection exists permanently, whether or not it is being used to transmit data. In a ***switched configuration***, the link connection is activated when needed, then disconnected. A special SDLC mode, based on X.21 ***short-hold mode*** (SHM) circuits, enables a switched connection to be dropped when no data activity has occurred for a given interval, and quickly re-established when it is needed again. SHM is used by certain European circuit-switched networks. Such networks provide fast call set-up and a short charging interval. SHM offers, therefore, potential cost savings for applications that transmit short bursts of data traffic over a switched network.

A ***point-to-point configuration*** has two link stations sharing a link connection; a ***multipoint configuration*** has three or more link stations sharing a link connection. In both point-to-point and multipoint configurations one link station has the role of primary link station. The ***primary link station*** controls the use of the link connection by all of the link stations attached to it. The remaining link stations are called ***secondary link stations***. In a multipoint configuration, the secondary link stations communicate only with the primary link station, and only when contacted by the primary link station; secondary link stations never communicate directly with each other.

Asynchronous SDLC is a form of SDLC in which the synchronous-clocking component of SDLC is replaced by an asynchronous-clocking component, thus allowing standard SDLC frames to be transported over asynchronous link connections. Asynchronous SDLC function may be implemented in programmable workstations. The function of the asynchronous SDLC component of a station is to provide the code and data transparency and the framing needed to transfer data between the SDLC elements of procedure component and the asynchronous support component of the station. The SDLC elements of procedure remain unchanged. Asynchronous SDLC combines the benefits of SDLC, including its widespread use within a layered architecture (SNA), with the benefits of low-cost, asynchronous hardware and line adapters.

System/370 Data Channels

A ***System/370 data channel*** (S/370 data channel) provides high-speed local attachment for nodes situated within relatively close proximity of one another. Unlike other data link control protocols, which transmit data serially, bit by bit, S/370 data channels transmit bits in parallel on multi-wire cables. Synchronizing parallel transmission requires distance limitations on channel-attached nodes. Placing nodes at greater than recommended distances can result in unacceptably high error rates.

A channel can be used in several configurations. A channel supporting devices such as communication controllers, terminal controllers, and printers, has a multipoint appearance. In this configuration, a single primary station selects the other stations, which are secondaries. Unlike an SDLC link, the channel allows a secondary station to send an attention signal to the primary station to request servicing. The channel also supports a point-to-point configuration for interconnecting mainframe processors.

IBM's ***Enterprise Systems Connection**** (ESCON*) product extends the range and data rates of the data channel. The fiber optic technology of the ES Connection allows devices such as display units and printers to be situated miles away from their host processors, and to communicate at speeds twice that of prior channel speeds. In addition, the ES Connection architecture enhances the connectivity of channel connections through its switched point-to-point topology capability. The ***switched point-to-point topology*** feature enables multiple channels to share link connections, channels and control units to be added without affecting current users, and alternate paths to be used when failures occur.

The ***APPN host-to-host channel*** function allows a VTAM node to attach to a second VTAM node with a high-performance direct channel using APPN protocols. The function enables two adjacent APPN mainframes to be connected by a channel interface. APPN host-to-host channel function uses the multipath channel interface to provide channel connectivity.

Local-Area Network (LAN) Connections

A ***local-area network*** (LAN) is a general-purpose link-level network confined to a small geographic area such as a building.

Token-Ring Link Connections: IBM's Token-Ring Network is a LAN consisting of ***ring stations*** connected in a logically circular fashion. A sample ring configuration of eight ring stations is shown in Figure 26.

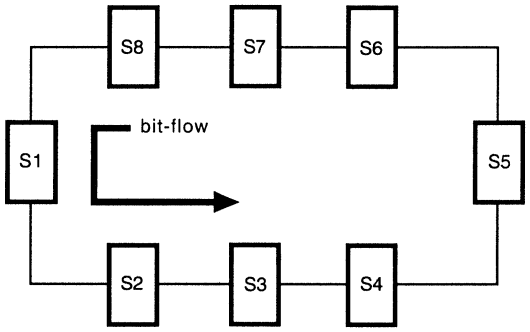


Figure 26. Sample Ring Configuration

Each ring station can serve one or more attached devices such as terminals or printers. Using ring-access protocols, ring stations allow their attached devices to communicate with other attached devices on the ring. A token-ring LAN contains no primary or secondary token-ring stations. The token-ring communication technique ensures that all stations share equal access to the ring.

The communication technique uses a small data packet called a ***token*** that circulates around the ring. When all stations are idle, the token is called a ***free token***. Before a ring station can send data to another ring station, it must wait until it receives a free token. The station changes the token to a ***busy token*** by changing the bit pattern, then transfers the token and the data to the ring. While the token is busy, no other station can transmit data on the ring.

The data travels along the path indicated by the arrow in Figure 26. Each ring station copies the data and places it back on the ring. The destination station also copies the data into a local buffer. When the data returns to the originating ring station, it provides acknowledgment that the data was transmitted successfully. The originating station then removes the data from the ring and inserts a new free token on the ring.

The protocol employed between token-ring stations is defined by the Institute of Electrical and Electronics Engineers (IEEE). It involves two sublayers: the medium access control sublayer and the logical link control sublayer. The ***medium access control*** (MAC) sublayer controls the routing of information between the physical layer and the logical link control sublayer. It conforms to IEEE standard 802.5 for token-rings. The ***logical link control*** (LLC) sublayer provides error recovery, sequencing, and flow control. It conforms to IEEE standard 802.2 for LANs. The token-ring link stations are defined in the LLC sublayer.

The token-ring LAN is an example of a shared-access transport facility on which multiple link stations are defined. A ***shared-access transport facility*** (SATF) is a transmission facility on which multiple pairs of nodes can form concurrently active links. For information on defining shared-access transport facilities in APPN net-

works, see "Virtual Routing Nodes and Connection Networks" in Chapter 4, "Defining Network Resources."

Fiber Distributed Data Interface (FDDI) is similar to the IEEE 802.5 token-ring. FDDI is defined as a dual counter-rotating ring, which operates at a defined rate of 100 Mbps. One ring is used for data transmission; the other is a backup in case of link or station failure. It involves four sublayers: the MAC sublayer, the LLC sublayer, the physical layer protocol (PHY), and the physical media dependent layer (PMD). The latter two previously comprised the physical layer. The PMD layer, which is the lower sublayer of the physical layer, provides the point-to-point communication between the FDDI stations. The PHY layer, which is the upper sublayer of the physical layer, provides the connection between the PMD and the MAC, which is the lower layer of the data link layer.

FDDI is a token-passing protocol; each station has the chance to transmit data when a token passes. A station can decide how many frames it will transmit using an algorithm that permits "bandwidth" allocation and thus may transmit many frames before releasing the token.

Ethernet is the IEEE version of LAN technology that uses carrier-sense, multiple-access, with collision detection (CSMA/CD), which is a bus-oriented LAN technology. In Ethernet, each station contains a transceiver interface that provides recognition of the destination address and performs error detection by a cyclic redundancy check. Each station also contains a network processor board that establishes and terminates circuit connections, and builds packets for transmission.

In a bus using collision detection, any station may transmit at any time. This results in collisions, which causes the system load to increase even more as data is retransmitting following collisions. To remedy this, the CSMA/CD protocol requires that a station transmit only when it discovers that the medium is quiet. Furthermore, CSMA/CD systems enable a station to stop a transmission it initiated if a collision is detected during transmission.

X.25 Links

The ***X.25 interface*** is an international standard recommended by the International Telegraph and Telephone Consultative Committee (CCITT). It was designed primarily as an interface to a public packet-switched data network. In a ***packet-switched data network*** (PSDN), users (DTEs) do not have exclusive right to a specific physical circuit. Instead, many network users share the same circuits for transmitting their messages. Messages are divided into segments called ***packets***. Packets relating to a common message are transmitted independently through the network until they reach their destination node. The destination node reassembles the message by reordering the packets into the same sequence in which they were transmitted.

One of the primary requirements for the X.25 link is user isolation. Users cannot interfere with the internal operation of the network, or affect the operation of other network users. A PSDN is another example of a shared-access transport facility. As with token-ring LANs, no primary or secondary station roles are defined on the network.

Frame Relay (FR)

Frame relay (FR) is an interface for high-performance routing of variable-length frames and is similar to the X.25 version of packet switching. It is a connection-oriented protocol for high-speed transfer of data over local- or wide-area networks. While it is possible for networks to use frame-relay techniques for transmission between network nodes, there will be many networks that do not work this way. A frame-relay network allows a user to attach through a frame-relay interface and provides the services and facilities necessary to support communication between frame-relay interfaces. In frame relay, communicating devices interact with one another across the network transparently. That is, a network is interposed between devices communicating on a link, but the devices are not aware of the interposed network.

Layer 2 is used in frame relay for the virtual circuit identifier (which was implemented in X.25 layer 3). Processing-intensive functions such as level-3 addressing, packet sequencing, and error correction are moved to the end-systems connected to the network. Packets in error are dropped by the network. Frame relay is simplistic, yet also has potential for use as the interface to fast-packet networks.

Other Data Link Control Protocols

SNA networks also support binary synchronous communication (BSC) and start-stop data link protocols. ***Binary synchronous communication*** and ***start-stop*** (also known as ***asynchronous***) data link control protocols allow SNA networks to transmit data to, and receive data from, non-SNA workstations. For additional information, refer to *Non-SNA Interconnection General Information Manual*.

Path Control

Components within the path control layer called ***path control instances*** route session traffic in the node. Path control instances route session traffic in two ways: between partner NAUs located in the same node, or between partner NAUs located in different nodes. Routing between session partners located in the same node is referred to as ***intranode routing***; routing between partners located in different nodes is referred to as ***internode routing***.

One internode path control instance exists in a node for each active transmission group connected to the node. A ***transmission group*** (TG) is a link or group of links between adjacent nodes that is treated as a single link by a path control instance. Between adjacent subarea nodes, a TG can consist of multiple parallel SDLC links. Between adjacent APPN nodes, a TG is synonymous with a link; each link between adjacent APPN nodes is used independently.

A path control instance consists of a path control manager and one or more path control elements. The ***path control manager*** initializes the path control instance and performs functions not related to session traffic routing such as communicating with management services. The ***path control element*** routes message units that flow on sessions between network accessible units (NAUs).

Transmission groups and the routing functions of path control are discussed in Chapter 6, "Establishing Routes Through the Network."

The path control manager performs the following functions:

- Initializes the path control instance according to parameters provided by the control point
- Connects and disconnects half-sessions to a path control instance
- Routes nonsession traffic
- Communicates with management services when errors are detected.

The path control element performs the following functions:

- Routes intranode or internode session traffic
- Segments message units as needed
- Controls transmission priority
- Controls virtual routes, including virtual-route pacing, in subarea nodes
- Controls explicit routes in subarea nodes.

Network Accessible Units

Network accessible units implement the upper four layers of the node: the transmission control, data flow control, presentation services, and transaction services layers. NAUs establish and control sessions in order to deliver control data and end-user data across the transport network. As shown in Figure 22 on page 41, nodes contain three fundamental kinds of NAUs: physical units, logical units and control points.

Physical Units

Physical units (PUs) exist in subarea and type 2.0 nodes. (In type 2.1 peripheral nodes, the control point performs the functions of a PU.) The PU supports sessions with control points in type 5 nodes and also interacts with the control point in its own node.

A physical unit provides the following functions:

- Receives and acts upon requests from SSCPs, such as activating and deactivating links to adjacent nodes
- Manages links and link stations, while accounting for the unique aspects of different link types
- Sets up virtual and explicit routes in T4 and T5 subarea nodes.

Logical Units

All node types can contain logical units (LUs). (T4 nodes, however, typically do not contain LUs except for protocol conversion for non-SNA terminals.) The LU supports sessions with control points in type 5 nodes and with LUs in other nodes. A type 6.2 LU using SSCP-independent protocols, however, does not engage in sessions with an SSCP.

End users access SNA networks through logical units. A logical unit manages the exchange of data between end users, acting as an intermediary between the end user and the network. A one-to-one relationship is not required between end users and LUs. The number of end users that can access a network through the same LU is an implementation design option.

Before end users can communicate with one another, their respective LUs must be connected in a session. In some cases, multiple, concurrent sessions between the same two logical units are possible. When such sessions are activated, they are called ***parallel sessions***.

SNA defines different kinds of logical units called ***LU types***. LU types identify sets of SNA functions that support end-user communication. Since SNA was announced, IBM has developed a number of LU types to handle the communication requirements of a variety of end users.

LU-LU sessions can exist only between logical units of the same LU type. For example, an LU type 2 can communicate only with another LU type 2; it cannot communicate with an LU type 3.

The LU types that SNA currently defines, the kind of configuration or application that each type represents, and the hardware or software products that typically use each type of logical unit are listed below.

LU type 1

LU type 1 is for application programs and single- or multiple-device data processing workstations communicating in an interactive, batch data transfer, or distributed data processing environment. The data streams used by LU type 1 conform to the SNA character string or Document Content Architecture (DCA). (For information on data streams, see Chapter 3, "Data Formats.") An example of the use of LU type 1 is an application program running under IMS/VS and communicating with an IBM 8100 Information System at which the workstation operator is correcting a database that the application program maintains.

LU type 2

LU type 2 is for application programs and display workstations communicating in an interactive environment using the SNA 3270 data stream. Type 2 LUs also use the SNA 3270 data stream for file transfer. An example of the use of LU type 2 is an application program running under IMS/VS and communicating with an IBM 3179 Display Station at which the 3179 operator is creating and sending data to the application program.

LU type 3

LU type 3 is for application programs and printers using the SNA 3270 data stream. An example of the use of LU type 3 is an application program running under CICS/VS and sending data to an IBM 3262 Printer attached to an IBM 3174 Establishment Controller.

LU type 4

LU type 4 is for:

1. Application programs and single- or multiple-device data processing or word processing workstations communicating in interactive, batch data transfer, or distributed data processing environments. An example of this use of LU type 4 is an application program running under CICS/VS and communicating with an IBM 6670 Information Distributor.
2. Peripheral nodes that communicate with each other. An example of this use of LU type 4 is two 6670s communicating with each other.

The data streams used by LU type 4 are the SNA character string (SCS) for data processing environments, and Office Information Interchange (OII) Level 2 (a precursor of DCA) for word-processing environments.

LU type 6.1
LU type 6.1 is for application subsystems communicating in a distributed data processing environment. An example of the use of LU type 6.1 is an application program running under CICS/VS and communicating with an application program running under IMS/VS.

LU type 6.2
LU type 6.2 is for transaction programs communicating in a distributed data processing environment. The type 6.2 LU supports multiple concurrent sessions. The LU 6.2 data stream is either an SNA general data stream (GDS), which is a structured-field data stream, or a user-defined data stream. LU 6.2 can be used for communication between two type 5 nodes, a type 5 node and a type 2.1 node, or two type 2.1 nodes. Examples of the use of LU type 6.2 are:

1. An application program running under CICS/VS communicating with another application program running under CICS/VS
2. An application program in an AS/400 communicating with a Personal System/2* (PS/2*).

Control Points

Every node defined by SNA contains a control point. In general, a control point manages the network resources within a domain. Domain resources include PUs, LUs, and link stations. Management activities include resource activation, deactivation, and status monitoring. A control point's domain and the range of its capabilities depend on the type of node in which it resides. Regardless of the node type, a control point performs the following common functions:

- Manages domain resources in accordance with the commands that operators issue
- Monitors and reports on the status of resources within its domain.

System Services Control Points

A type 5 subarea node contains a ***system services control point*** (SSCP). A system services control point activates, controls, and deactivates network resources in a subarea network. In order to control and provide services for its subordinate nodes, an SSCP establishes sessions with NAUs in those nodes. For example, using a directory of network resources, an SSCP can use an SSCP-LU session to assist an LU in locating a partner LU and establishing an LU-LU session.

An SSCP provides the following functions:

- Manages resources on a subarea network level in accordance with the commands that network operators issue
- Coordinates the initiation and termination of sessions between LUs in separate nodes within its domain, or across domains in cooperation with other SSCPs
- Contacts nodes as needed, for example, by causing an autodial modem to dial a workstation over a switched-link connection
- Coordinates the testing and status monitoring of resources within its domain.

Physical Unit Control Points

Type 4 nodes and type 2.0 nodes contain a ***physical unit control point*** (PUCP). The PUCP performs a subset of the functions of the SSCP, but only for the node in which it resides. The PUCP interacts with its local PU in order to manage node resources. A PUCP shares control of its resources with one or multiple SSCPs, enabling local activation of node resources. For example, a PUCP can independently activate its locally attached links through interaction with its local PU.

A PUCP performs the following functions:

- Manages the resources in its node in accordance with commands from network or local operators
- Monitors and reports to its local operator (or to a log file) on the status of the resources in its own node.

Control Points in LEN and APPN Nodes

The control point in a type 2.1 node or a type 5 node is called simply a ***control point*** (CP). Like other control points, it can activate locally attached links, interact with a local operator, and manage local resources. It can also provide network services, such as partner location and route selection, for local LUs. Its other functions vary according to the role its node assumes in a network.[2],[3]

A LEN node CP does not communicate with a control point in another node. It communicates only with other components in its own node for controlling local resources and aiding local LUs in establishing LU-LU sessions.

An APPN end node CP participates in CP-CP sessions with the CP in an adjacent network node server. Two parallel sessions using LU 6.2 protocols are established between the partner CPs. The APPN end node does not establish CP-CP sessions, however, with any adjacent LEN or APPN end nodes. If it is attached to multiple APPN network nodes, the APPN end node chooses only one of them to be its active server; it does not establish CP-CP sessions with more than one network node at a time. (It can, however, route LU-LU sessions through any adjacent network node as determined by route selection criteria.)

In addition to the functions of a LEN node, an APPN end node performs the following functions:

- Manages the resources in the node in accordance with commands from local operators
- Makes its local resources (LUs) known to the APPN network, in response to operator commands, by registering them with its network node server and deleting them when they are no longer available for use
- Supports directory search requests sent by its network node server in order to locate, or verify the location of, a local LU

---

2 A T5 node acting as an interchange node contains both an SSCP and an APPN CP.

3 When its node is attached as a peripheral node in a subarea network, a T2.1 node CP uses both SSCP-dependent and SSCP-independent protocols. It uses SSCP-dependent protocols when acting as a T2.0 PU, receiving SSCP commands and exchanging management services data with an SSCP over SSCP-PU sessions. It uses SSCP-independent protocols when aiding local independent LUs in establishing LU-LU sessions.

- Supports route selection for LU-LU sessions by reporting its locally attached transmission groups (TGs) to its network node server and by reporting route information computed by its network node server to a requesting local LU (which subsequently appends the information to its BIND, used when requesting session activation)
- Monitors and reports on the status of its local resources through management services exchanges with its network node server, and hence possibly with a nonadjacent focal point.

An APPN network node CP provides network services for attached LEN and APPN end nodes, such as LU-LU session partner location and route computation. It performs these functions transparently for an attached LEN end node, because no CP-CP sessions are established with the LEN end node.

In addition to the functions of an APPN end node, an APPN network node also performs the following functions:

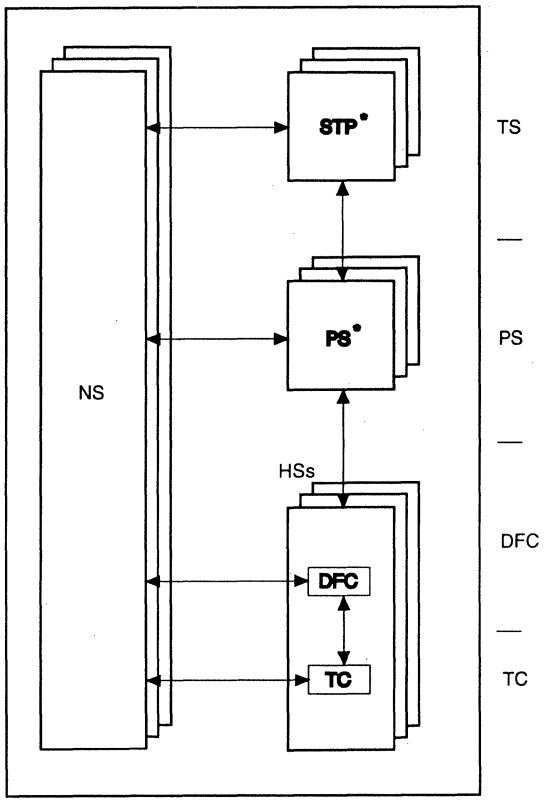
- Participates in directed and broadcast directory searches, involving other network nodes, in order to locate a session partner for an LU-LU session requested by an LU in its domain; retains such information in its cache memory to speed up future searches.
- Calculates a route for an LU-LU session using TG information received from the origin LU on a search request, and from the destination LU on the search response.
- Exchanges network topology information with other network nodes so that changes in network configuration can be communicated to all network nodes in the APPN network.

Intermediate Session Routing Component

The ***intermediate session routing*** (ISR) component exists only in an APPN network node. The ISR performs the intermediate session routing function. It routes and paces session traffic, one session stage at a time, through the intermediate routing network of an APPN network. This pacing is referred to as ***hop-by-hop pacing***. For information on APPN routing, see "Routing in APPN Networks" in Chapter 6, "Establishing Routes Through the Network." For information on pacing, see Chapter 7, "Controlling Congestion in the Network."

NAU Components

Network accessible unit components implement the upper four architectural layers of SNA. NAUs contain half-sessions, presentation services, and service transaction programs, plus a common set of NAU services used by all components. Figure 27 illustrates the major components of a generic NAU and the layers in which they reside.



* Do not exist in PUs

Legend:

DFC = Data Flow Control
HS = Half-Session
NS = NAU Services
PS = Presentation Services
STP = Service Transaction Program
TC = Transmission Control
TS = Transaction Services

Figure 27. Components of a Generic NAU

As Figure 27 indicates, not all NAUs require all four SNA layers. The physical unit (PU), for example, does not require the presentation services or transaction services layers.

Half-Sessions

A ***half-session*** consists of the resources allocated by a network accessible unit to support a session. For each session that an NAU supports, a half-session must be allocated. A NAU typically supports more than one session at a time. A physical unit in a T4 node, for example, might concurrently support sessions with multiple SSCPs. A logical unit can concurrently support a session with an SSCP and multiple sessions with LUs in other nodes.

Each half-session contains a transmission control component and a data flow control component residing within the transmission control and data flow control layers, respectively. The ***transmission control*** (TC) component performs lower-level session-related functions such as sequence number verification and message pacing. The ***data flow control*** (DFC) component performs higher-level functions such as assigning sequence numbers and correlating requests with responses.

When activating sessions, NAUs can elect to use any of several possible data transport protocols. To facilitate agreement on data transport protocols between communicating NAUs, SNA defines data fields called *profiles* which are carried in the session-activation request. ***Transmission services profiles*** (TS profiles) define protocol options relevant to the transmission control layer, such as pacing window counts. ***Function management profiles*** (FM profiles) define protocol options relevant to the data flow control layer, such as message sequencing and message response requirements. For more information on data transport protocols, see Chapter 8, "Transporting Data Through the Network."

The transmission control component performs the following functions:

- Verifies received sequence numbers
- Enciphers and deciphers data
- Manages session-level pacing
- Reassembles BIUs from BIU segments
- Provides boundary function support for peripheral nodes
- Enforces protocols specified in TS profiles.

The data flow control component performs the following functions:

- Builds request and response headers
- Assigns sequence numbers
- Correlates requests and responses
- Enforces protocols specified by FM profiles.

Presentation Services

Presentation services (PS) components exist in logical units and in the control points of APPN nodes. The primary function of the PS component is to provide support for the transaction program interface with an LU or CP. Such support varies by LU type and may include compressing session data for more efficient data transmission and adding column headings for display. The transaction program can be an application transaction program or a service transaction program. Application transaction programs communicate with LUs, and service transaction programs reside in (and thus communicate with) both LUs and CPs.

In an LU or CP, one PS component exists for each half-session that requires presentation services. As an example of a transaction program interface function, a PS component can format data for a particular presentation medium.

The presentation services component performs the following functions:

- Loads and invokes transaction programs
- Enforces transaction program interface rules
- Creates request units from transaction program data
- Generates function management headers.

Transaction Services

The transaction services layer of the node contains service transaction programs. ***Service transaction programs*** (STPs) are IBM-supplied transaction programs that the architecture defines. They exist in LUs and CPs and provide network services to other node components and end-users. SNA/DS and SNA/FS are examples of STPs created to provide end-user services. These STPs provide data distribution and file services. SNA/DS and SNA/FS are also used by SNA/MS for exchanging network management bulk data between nodes. The

RECEIVE_NETWORK_SEARCH and SEND_NETWORK_SEARCH programs are examples of STPs created to provide directory services in APPN nodes. These programs reside in the APPN node control point and communicate across CP-CP sessions to locate resources in an APPN network.

NAU Services

NAU services are various services components that reside in network accessible units. They provide network and nodal services such as initializing and controlling layer components within the node. To perform their functions, NAU services components employ the services of other node components. For example, configuration services sends commands to data link control instances in order to activate and deactivate links. NAU services include configuration services, session services, management services, directory services, the address space manager, topology and routing services, the session manager, and the resources manager.

Configuration services (CS) manages the physical links attached to the node. It is CS, for example, that controls the activation of a switched link connection. CS exists in control points and physical units, but not in logical units. For more information on configuration services, see Chapter 5, "Activating the Network."

Session services (SS) assists in initiating and terminating CP-CP and LU-LU sessions. SS exists only in control points. It supplies route and, optionally, class-of-service information to the session manager activating the session. For more information on session services, see Chapter 5, "Activating the Network" and Chapter 8, "Transporting Data Through the Network."

Management services (MS) provides facilities for network management including reporting resource status, testing various network resources, and tracing network activity at the link level. MS facilities exist in all network accessible units. For more information on management services, see Chapter 10, "Managing an SNA Network."

Directory services (DS) components maintain a distributed directory of logical units throughout an APPN network, and assist LUs in APPN nodes in locating session partners for LU-LU sessions. DS exists in the control points of LEN and APPN nodes. In a LEN or APPN end node, DS assists LUs that are local to the node; in a network node, DS assists all LUs within the control point domain. DS updates directory database entries as the result of system definition, run-time registration, or active network searches. The directory database enables DS in a node to cooperate with DS components elsewhere in the APPN network to locate resources. For more information on directory services, see Chapter 5, "Activating the Network."

The ***address space manager*** (ASM) manages the address spaces being used by the path control instances within a node. The ASM exists in the control points of LEN and APPN nodes. The ASM assigns a session address that identifies a session initiated from its node. The session address identifies the session both locally and to the adjacent node as well. For more information on the address space manager, see "Routing in APPN Networks" in Chapter 6, "Establishing Routes Through the Network."

Topology and routing services (TRS) calculates routes in an APPN network. It exists in the control point of an APPN node. TRS has three components: the topology database manager, the class-of-service manager, and route selection ser-

vices. The topology database and class-of-service managers maintain databases that are used in route calculation. Route selection services (RSS) calculates routes for sessions in APPN networks. In an APPN end node, RSS simply selects the appropriate transmission group to an adjacent node in cases where its network node server has not selected the route for it (such as to the network node server itself). In a network node, RSS selects routes throughout the network for LU-LU sessions to use. For more information on topology and routing services, see Chapter 6, "Establishing Routes Through the Network."

The ***session manager*** (SM) manages session initiation and termination in coordination with a control point. SM exists in all NAUs that support sessions. During session initiation, SM creates half-sessions and connects them to path control instances. It also supplies session-initiation message parameters and handles session-initiation message exchanges between session partners. In the system services control point, SM also manages keys for session cryptography. For more information on the session manager, see Chapter 8, "Transporting Data Through the Network."

The ***resources manager*** (RM) manages session access for transaction programs. RM exists in the control points of APPN nodes and in the type 6.2 LU. It requests SM to activate sessions for the support of conversations, creates presentation services components and connects them to half-sessions, and chooses the session to be used by a conversation. In the type 6.2 LU, RM also performs LU-LU verification and conversation-level security checks. For more information on the resources manager, see Chapter 8, "Transporting Data Through the Network."

Chapter 3. Data Formats

This chapter discusses message-unit formats, control fields, and data streams defined by SNA.

Requests and Responses

Message units flowing through the network contain either a request or a response. ***Requests*** are message units that contain (1) end-user data or (2) network commands. ***Responses*** are message units that acknowledge the receipt of a request.

Requests that contain end-user data are ***data requests***. Examples of end-user data include payroll data, personnel data, insurance policy data, and inventory data. Requests that contain network commands are ***command requests***. Network commands initiate and terminate sessions and control communication between network accessible units.

Responses are either positive or negative. ***Positive responses*** indicate that a request was received and is acceptable. ***Negative responses*** indicate that a request was received, but is unacceptable. Negative responses contain error codes that explain why the request is unacceptable.

Message Unit Formats

Network accessible units, path control elements, and data link control elements all use different message-unit formats to exchange information with other network accessible units, path control elements, and data link control elements in the network. This section explains the different message-unit formats that network resources use to exchange information and the type of information that each message unit contains.

SNA defines the following message-unit formats that NAUs, path control elements, and data link control elements use:

- Network accessible units use basic information units (BIUs)
- Path control elements use path information units (PIUs)
- Data link control elements use basic link units (BLUs).

Basic Information Unit

Network accessible units use ***basic information units*** (BIUs) to exchange requests and responses with other network accessible units. Figure 28 shows the format of a BIU.

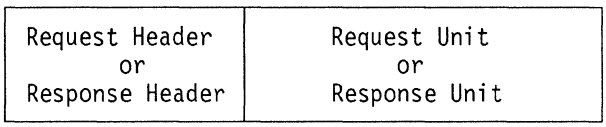


Figure 28. Basic Information Unit (BIU) Format

Basic information units that carry requests contain both a request header and a request unit. Basic information units that carry responses consist of (1) both a response header and a response unit or (2) only a response header.

Request Header

Each request that an NAU sends begins with a ***request header*** (RH). A request header is a 3-byte field that identifies the type of data in the associated request unit. The request header also provides information about the format of the data and specifies protocols for the session. Only NAUs use request header information.

Request Unit

Each request that an NAU sends also contains a ***request unit*** (RU). A request unit is a field of variable length that contains either end-user data (data RUs) or an SNA command (command RUs). Data RUs contain information that is exchanged between end users. Command RUs control the operation of the network.

Response Header

Each response that an NAU sends includes a ***response header*** (RH). Like a request header, a response header is a 3-byte field that identifies the type of data in the associated response unit. A bit called the ***request/response indicator*** (RRI) distinguishes a response header from a request header.

The receiving NAU indicates whether the response being returned to the request sender is positive or negative by setting a single bit, called the ***response type indicator*** (RTI), in the response header.

Response Unit

A ***response unit*** (RU) contains information about the request. Positive responses to command requests generally contain a 1–3 byte response unit that identifies the command request. Positive responses to data requests contain response headers, but no response unit. Negative response units are 4–7 bytes long and are always returned with a negative response. Response units are identified as response RUs.

The receiving NAU returns a negative response to the request sender if:

- The sender violates an SNA protocol
- The receiver does not understand the transmission
- An unusual condition, such as a path outage, occurs.

The receiving NAU returns a 4–7 byte negative response unit to the request sender. The first 4 bytes of the response unit contain sense data explaining why the request is unacceptable. The receiving NAU sends up to three additional bytes that identify the rejected request.

Note: Request units and response units are often referred to generically as *request/response units* when it is not necessary to distinguish between the two.

For additional information on basic information units, refer to *SNA Formats*.

Path Information Unit

The message-unit format used by path control elements is a ***path information unit*** (PIU). Path control elements form a PIU by adding a transmission header to a basic information unit. Figure 29 shows the format of a PIU.

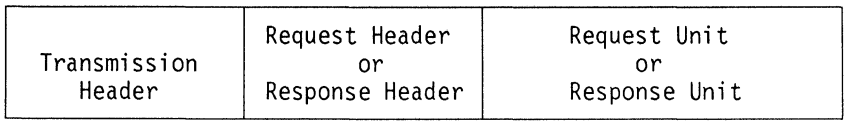


Figure 29. Path Information Unit (PIU) Format

Path control uses the ***transmission header*** (TH) to route message units through the network. The transmission header contains routing information for the transport network.

SNA defines different transmission header formats and identifies the different formats by a ***format identification*** (FID) type. Transmission headers vary in length according to their FID type. Path control uses the different FID types to route data between different types of nodes.[4]

FID 0: Path control uses this format to route data between adjacent subarea nodes for non-SNA devices. Few networks still use the FID 0; now a bit is set in the FID 4 transmission header to indicate whether the device is an SNA device or a non-SNA device.

FID 1: Path control uses this format to route data between adjacent subarea nodes if one or both of the subarea nodes do not support explicit and virtual route protocols.

FID 2: Path control uses this format to route data between a subarea boundary node and an adjacent peripheral node, or between adjacent APPN or LEN nodes.

FID 4: Path control uses this format to route data between subarea nodes if the subarea nodes support explicit and virtual route protocols.

FID 5: Path control uses this format to route data over an RTP connection. The FID 5 is very similar to the FID 2.

For additional information on path information units and transmission headers, refer to *SNA Formats*.

[4] SNA defines two additional FID types: FID 3 and FID F. Before the type 1 node became obsolete, path control used FID 3 to route data between a subarea node and a type 1 node. For information about FID F, refer to *SNA Format and Protocol Reference Manual: Architectural Logic*.

Basic Link Unit

Data link control uses a message-unit format called a ***basic link unit*** (BLU) to transmit data across a link. Data link control forms a BLU by adding a ***link header*** (LH) and a ***link trailer*** (LT) to a PIU. Link headers and link trailers contain link control information that manages the transmission of a message unit across a link. Only data link control elements use link header and link trailer information. For further explanation of SDLC link headers and link trailers, refer to *IBM SDLC Concepts*. Figure 30 shows the format of a BLU.

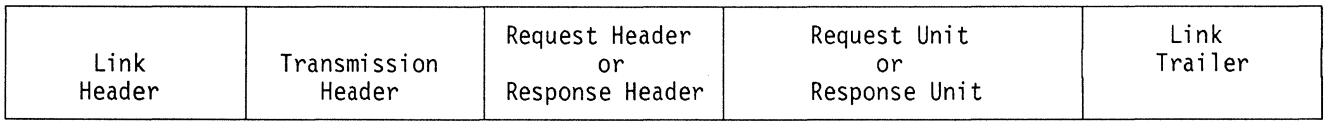


Figure 30. Basic Link Unit (BLU) Format

Network Layer Packet (NLP)

A ***network layer packet (NLP)*** is used to transport data through an HPR network. It is composed of the network layer header (NHDR), transport header (THDR) and the data. The length of a packet must not exceed the maximum packet size of any link over which the packet will flow. The maximum packet size of the links is obtained during the route setup protocol, discussed later.

All of the fields in the NLP are of variable lengths. Figure 31 illustrates the format of an NLP.

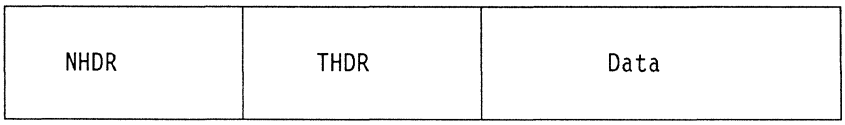
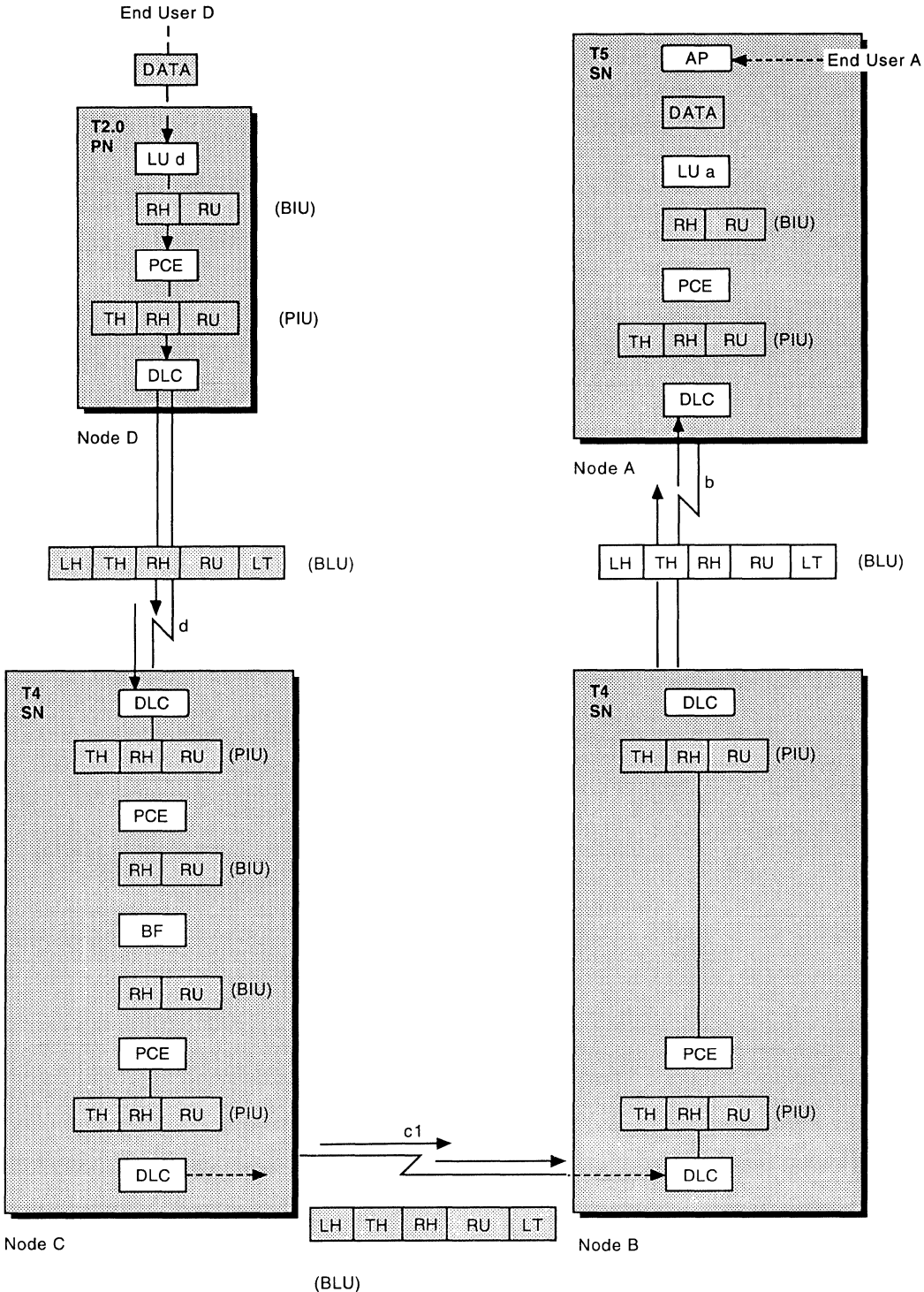


Figure 31. Network Layer Packet (NLP) Format

Using Data Formats in Data Transport

Figure 32 on page 65 reviews the format of the various message units and shows how NAUs, path control, and data link control use the formats to route a request from end user D to end user A.



Legend:

AP	=	Application Program	LU	=	Logical Unit
BF	=	Boundary Function	PCE	=	Path Control Element
BIU	=	Basic Information Unit	PIU	=	Path Information Unit
BLU	=	Basic Link Unit	RH	=	Request Header
DLC	=	Data Link Control	RU	=	Request Unit
LH	=	Link Header	TH	=	Transmission Header
LT	=	Link Trailer			

Figure 32. Use of Data Formats

The following section describes how the message formats shown in Figure 32 are used.

Node D

1. End user D gives end-user data to LU *d*.
2. LU *d* creates a request unit and affixes a request header to it; this forms a basic information unit (BIU). (Multiple BIUs may be needed to accommodate a large message.)
3. LU *d* gives the BIU to a path control element.
4. Path control affixes a transmission header to the BIU; this forms a path information unit (PIU).
5. Path control gives the PIU to a data link control element.
6. Data link control affixes both a link header and a link trailer to the PIU; this forms a basic link unit (BLU).
7. Data link control transmits the BLU over link *d* to node C.

Node C

1. The data link control element that receives the BLU removes the link header and link trailer, and then gives the PIU to a peripheral path control element.
2. This path control element routes the RH and RU (BIU) to the boundary function (BF). The BF then passes the BIU to a subarea path control element, which determines where to route the message unit text. Path control then gives the PIU to another data link control element.
3. Data link control affixes both a link header and a link trailer to the PIU.
4. Data link control transmits the BLU over link *c1* to node B.

Node B

1. The data link control element that receives the BLU removes the link header and link trailer and then gives the PIU to path control.
2. Path control uses the information in the transmission header to determine where to route the message unit next. Path control then gives the PIU to another data link control element.
3. Data link control affixes both a link header and a link trailer to the PIU.
4. Data link control transmits the BLU over link *b* to node A.

Node A

1. The data link control element that receives the BLU removes the link header and link trailer and then gives the PIU to path control.
2. Path control uses information in the transmission header to determine that node A is the destination subarea node. Path control removes the transmission header and then delivers the BIU to the destination NAU.
3. LU *a* removes the request header and gives the RU to end user A.

Formats Within the Request Unit

Within the RU, data can be further formatted either by node components or by the end user. RUs often contain SNA-defined control fields and data streams.

Data Streams

Data streams consist of end-user data, formatted for presentation to the destination end user by means of application control fields contained in the data.

SNA Character String

SNA character string control codes are EBCDIC control codes that define a data stream. Their primary function is to format a visual presentation medium such as a printed page or an alphanumeric display screen. These control codes can also set modes of device operation or define data to be used in a unique fashion. For example, they can be used for communication between a device operator and an application program where the specific function associated with the code is defined in a protocol established between the program and the operator.

An ***SNA character string data stream*** consists of a sequential string of SNA character string control codes and data characters. Control codes can be intermixed with graphic data characters. SNA character string control codes are in the range X'00' through X'3F' plus X'FF'. Graphic codes are in the range X'40' through X'FE'. Data streams can include other data types (such as binary and packed decimal), but only if accompanied by certain specific SNA character string control codes. Some codes also accompany 1-byte parameters specifying functions or binary values.

SNA character string functions do not include data flow control functions, even though they may be available to a keyboard operator through keys on the keyboard. Cancel, for example, is a data flow control request that can be initiated by a key on the keyboard.

An SNA character string control and parameter sequence can be contained entirely within a single RU or it can span two or more RUs; however, it must be entirely contained within one RU chain. An ***RU chain*** is a unidirectional sequence of BIUs that is treated, during error recovery, as a single unit. For information on chaining protocols, see Chapter 8, "Transporting Data Through the Network."

Examples of SNA character string controls are: backspace, carriage return, form feed, horizontal tab, indent tab, presentation position, select left platen, set horizontal format, superscript, and word underscore.

LU types 1, 4, and 6.2 can use SNA character string controls.

SNA 3270 Data Stream

The ***SNA 3270 data stream*** consists of user-provided data and commands that logical units transmit over an LU-LU session. Logical units also transmit control information that governs the way the receiver handles and formats the data.

The SNA 3270 data stream is the only data stream that LU types 2 and 3 use. It is an optional data stream for LU types 6.1 and 6.2. The data stream supports file-to-file transfer, display applications, and printer applications.

An application program communicates with a display station operator using one of two methods. In the first method, the application program leaves the display surface unformatted, and the operator uses it in a free-form manner. In the second method, the application program completely or partially formats the display surface (that is, organizes or arranges it into fields) and the operator enters data into the fields.

The SNA 3270 data stream enables the application programmer to divide the display surface into one active area and, optionally, one or more reference areas. Each area is called a *partition*. The *active* partition contains a cursor and is the only partition in which the operator can enter data or requests.

Specific information on the 3270 functions that can be specified in an SNA 3270 data stream appears in *IBM 3270 Data Stream Programmer's Reference*.

Intelligent Printer Data Stream

The ***Intelligent Printer Data Stream*** (IPDS) is a host-to-printer data stream that describes the layout of pages to be printed on an all-points-addressable printer. The printer can mix text, raster image graphics, vector graphics, and bar codes on a single page. IPDS merges the different types of data, creating an integrated, mixed data page.

General Data Stream Variable

The ***general data stream variable*** (GDS variable) consists of transaction program data preceded by a 2-byte binary logical length (LL) field and a 2-byte binary identification (ID) field. Figure 33 shows the format of a GDS variable.

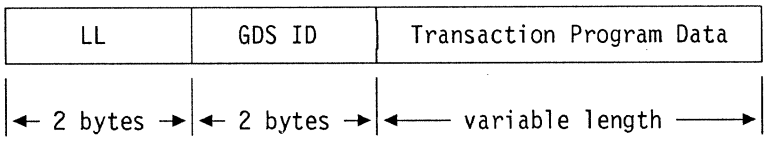


Figure 33. General Data Stream (GDS) Variable Format

The LL field is a descriptive prefix that indicates the length of the general data stream variable, including the LL field itself. The identification field determines the formats of the fields that follow (character and binary formats).

General data stream variables enable sending and receiving transaction programs to interpret the records they exchange in the same way. IBM service transaction programs (STPs) use general data stream variables to transfer data to one another. GDS variables pertaining to different STPs are distinguished by the contents of the ID field.

For additional information on the general data stream, refer to *SNA LU 6.2 Reference—Peer Protocols* and *SNA Formats*.

Control Headers

Certain control fields called function management headers and presentation services headers can be placed at the beginning of a request unit. Unlike request headers, which are required with every request unit, function management and presentation services headers are transmitted only when needed.

Function Management Headers

A ***function management header*** (FM header) carries control information for logical units and control points that support LU 6.2 protocols. A format indicator (FI) in the request header identifies whether the request unit contains any FM headers. Figure 34 shows the format of an FM header.

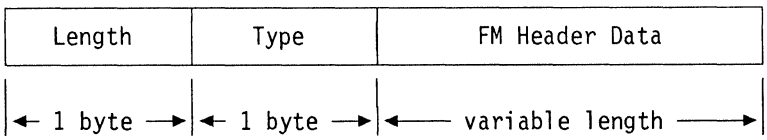


Figure 34. Function Management (FM) Header Format

Logical units differ in the kinds of FM headers they use. Type 6.2 logical units use FM headers for such functions as specifying the name of a target transaction program and carrying LU-LU session password verification data. Other types of logical units use FM headers to tell the receiving logical unit how to format a data RU for presentation to the end user.

LUs transmit FM headers only after session activation. Parameters in the BIND command indicate whether FM headers are to be used during the session and if there are limitations on their use. BIND parameters specify (1) whether a logical unit has full capability to use headers, (2) whether a logical unit has limited capability to use headers, or (3) whether a logical unit cannot use headers. The option selected depends on whether the LU type supports FM headers and on the amount of header capability needed to format end-user data.

For additional information about the use of FM headers by type 6.2 logical units, see Chapter 8, "Transporting Data Through the Network." For additional information about the use of FM headers by other types of logical units, refer to *SNA—Sessions Between Logical Units* and *SNA Formats*.

Presentation Services Headers

Presentation services headers contain control data for sync point managers in type 6.2 logical units. A ***sync point manager*** is the part of a presentation services component of the type 6.2 LU that provides the sync point services. Sync point services enable communicating transaction programs to designate when resource changes relating to the same *logical unit of work* are complete. The only type of PS header currently defined is ***sync point control***. For information on sync point protocols, see Chapter 8, "Transporting Data Through the Network."

The PS header uses a *logical length* (LL) field containing X'0001'. This differs from the conventional usage of the LL field in a GDS variable, which includes the 2-byte length of the LL field in the calculated length. All LLs that indicate a length of less than 2 are thereby reserved for use by the LU. Figure 35 shows the format of a PS header.

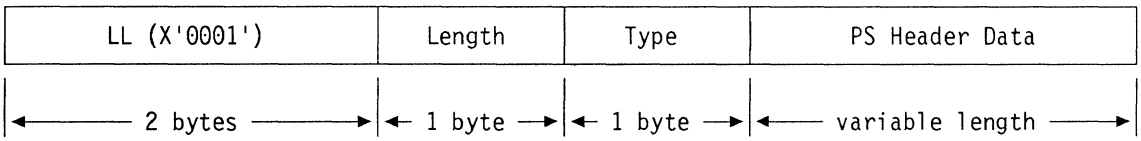


Figure 35. Presentation Services (PS) Header Format

Chapter 4. Defining Network Resources

This chapter discusses the requirements and methodology for the definition of network resources.

Resource Definition

Before a network can be activated, the network's resources must be defined to access methods, network control programs, or other network control software. The nature of the definition process depends on the type of node within which the network control software resides. Resources must be defined to a subarea control point. For subarea nodes (T5 and T4 nodes), network resources are defined by system definition. A ***system definition*** specifies to each subarea node control point the resources it can control. These network resources include logical units, physical units, links, and link stations.

A subarea node control point gains control of its resources by activating them. A system generation for a T5 node uses access-method resource-definition statements that specify the resources in the domain of the node's SSCP. It includes both the node's local resources and all the resources in its subordinate nodes and the links connecting them. Similarly, a system generation for a T4 node uses network-control-program macro instructions that specify both the node's local resources and the other resources in the node's subarea.

Not all resources need be initially defined to subarea nodes. There are several methods by which resources may be dynamically defined to the network:

- ***Dynamic reconfiguration*** allows peripheral nodes to be dynamically added to and deleted from nonswitched links, for example, without regenerating the network control program in a T4 node.
- ***Dynamic definition of independent LUs*** allows an independent LU to activate an LU-LU session with an LU in a subarea network or APPN network without prior definition of the independent LU to the subarea network. The SSCP controlling the independent LU's owning node dynamically stores the name and location of the independent LU as determined from the BIND request sent by the independent LU. Subsequently, other LUs in the network can activate LU-LU sessions with the independent LU, because the independent LU is then known to the subarea network.
- ***Dynamic definition of dependent LUs*** allows dependent LUs to be defined using information obtained in the NMVT that specifies how many dependent LUs need to be defined for a particular PU. The NMVT flows following the activation of a PU. Dynamic definition of dependent LUs may be done on switched or nonswitched lines.
- ***Dynamic definition of switched resources*** uses information from the XID exchange for switched PUs to allow switched PUs and their associated dependent LUs to be dynamically defined. As with dynamically defined independent LUs, other LUs in the network can activate LU-LU sessions with the dynamically defined dependent LU, because the dependent LU is then known to the network.
- ***Dynamic PU definition*** dynamically defines an adjacent link station (or PU). This method, like dynamic definition of switched resources, relies on information obtained in the XID to dynamically create a definition of the resource. After the adjacent link station is defined, it may be used for connectivity to its independent logical units.

For LEN and APPN nodes, resources are defined both by node operators and (for APPN network nodes) by other nodes in the network. A ***node operator*** can be either a command file or a human operator. A node operator can define a node

resource either before or after node activation. Node operators define resources using the node operator facility. The ***node operator facility*** (NOF) is the component of a LEN or APPN node through which node operators communicate with the node. The NOF enables a node operator not only to define resources, but to activate them, deactivate them, and obtain status information on them as well. The resources defined to a node can include resources external to the node as well as those local to it. Operators define resources external to a node when the resource locations cannot be determined dynamically through resource registration or search protocols.

Following node activation, APPN network nodes can learn of network resources dynamically through resource registration and search protocols. To facilitate the search for a partner LU during LU-LU session initiation, an APPN end node can send local resource information to its network node server in a process called ***resource registration***. The network node server stores the resource information as ***domain entries*** in its local directory. For further information, see "Resource Registration" in Chapter 5, "Activating the Network." Search protocols are performed by a network node on behalf of a domain LU (a local LU or an LU in a client end node) initiating an LU-LU session. After locating the partner LU, the network node server can store the location information as an ***other-domain entry*** in its distributed network directory (or as a domain entry if also in its domain). For further information on search protocols, see "Locating the Destination LU" in Chapter 6, "Establishing Routes Through the Network."

Using Directories

During resource definition, network resources are identified by names that are stored in system directories. In subarea networks, each SSCP also assigns an address to each of its resources and stores the addresses along with their associated resource names in its directory. In APPN networks, directories contain the names of resources and the names of the control points of the nodes where the resources are located. APPN nodes do not assign addresses to resources.

Both subarea and APPN networks use directories for locating resources, but they use them in very different ways. In subarea networks, SSCPs use directories to translate network names to network addresses for use by the transport network. In addition, SSCPs translate names to addresses across domains. If the named resource is not in an SSCP's domain, the SSCP uses a cooperative directory to identify which SSCP can provide the name-to-address translation. Thus, SSCPs cooperatively translate a network name that an end user in one domain specifies into a network address in another domain. APPN nodes, on the other hand, do not translate names to addresses. Instead, they use the names in the directories, and use search protocols if necessary, to determine the locations of session partners.

Virtual Routing Nodes and Connection Networks

Recall that a shared-access transport facility (SATF) allows concurrent communication between multiple pairs of link stations attached to the SATF. In a subarea network, defining nodes on an SATF represents considerable system definition, because each node attached to the facility must have defined to it all the nodes that it can reach over the facility, and the data link control (DLC) signaling information needed to do so. (An example of DLC signaling information is a ring station address on a token-ring.) Because each connection between a pair of nodes on an SATF requires two definitions, one in each node, the number of definitions required is N×(N-1), where N is the number of nodes.

In an APPN network, this definition overhead is considerably reduced by the use of the virtual routing node. A ***virtual routing node*** (VRN) is a representation of a node's attachment to an SATF. It is used during link station definition for the SATF. Any link station defined as having a connection to a virtual routing node is assumed to be able to communicate with any other link station connected to that VRN. With a VRN, therefore, the number of link definitions required for connectivity between nodes on an SATF is equal to the number of nodes on the SATF. Session traffic between two APPN nodes that have defined the VRN can be routed "through" the VRN without passing through any real network node. TDUs will never be exchanged with a VRN. The SATF, and the set of all nodes defined as having a connection to a common VRN representing the SATF, make up a ***connection network*** (CN). At each node, the NOF represents the CN to the node by a common network name.

During LU-LU session activation, the CN name is used as the CP name of the virtual routing node. When an LU in an APPN end node (EN) attached to a CN initiates an LU-LU session, it reports the CN name, and its own DLC signaling information, to its network node (NN) server in the session-initiation request. If the NN server then determines through search protocols that the destination can be reached through the same CN, the NN server calculates a route for the session that traverses the CN and returns the routing information, including the destination EN's signaling information, to the origin EN. A connection between the two ENs over the CN can then be established. Because an EN needs its NN server to provide it the control information to connect it to a partner through a connection network, it must have an explicitly defined connection to its server. Of course, this may be defined on the SATF on which it also defines a CN to other nodes.

Connection network (CN) purposes and concepts are the same in HPR networks as they are in basic APPN networks.

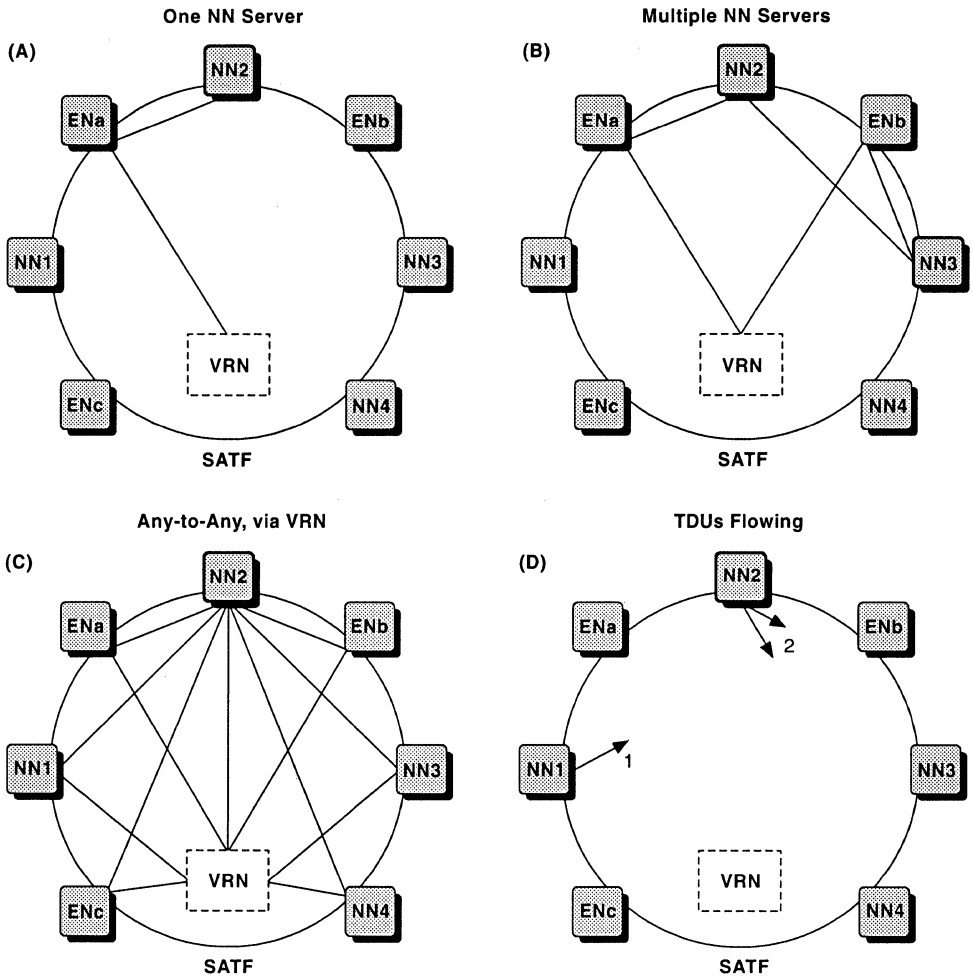


Figure 36. Shared-Access Transport Facility (SATF) with VRN

Figure 36 (A) illustrates the minimal definition requirements for an APPN end node. ENa has defined two connections: one to the VRN and one to its network node server (NN2).

Session setup data and TDUs are routed through an APPN network using CP-CP sessions between adjacent nodes. A virtual routing node is not a real node, therefore, nodes cannot establish CP-CP sessions with or through, a virtual routing node. Furthermore, if two APPN end nodes do not share the same network node server, session establishment between LUs on the two end nodes is possible only if their network node servers have CP-CP connectivity. This also applies to session establishment between LUs on two network nodes.

Figure 36 (B) illustrates CP-CP connectivity between two APPN network nodes. In this example, CP-CP connectivity requires that the network nodes have defined a link between each other, and established CP-CP sessions, or that the two network node servers can exchange data through one or more intermediate network nodes with active CP-CP sessions between each pair of adjacent network nodes.

Session establishment between LUs owned by APPN end nodes, provided no direct link has been defined between the two APPN end nodes, requires assistance from a network node server. As the APPN end nodes cannot have CP-CP sessions with a network node server through a VRN, each APPN end node must define a link to its network node server as well as define the connection to the VRN.

Figure 36 (C) illustrates the benefits of defining a VRN. As mentioned earlier, two link definitions are required in each node: one to the VRN and one to a network node defined by all nodes. NN2 is the only node that requires link definitions to all nodes. (NN2, in this case, need not define a link to the VRN.) NN2 only assists in session setup; no session data will be routed through the node. For performance reasons, more than one common network node server can be defined.

Figure 36 (D) illustrates the way TDUs are handled for a VRN. NN1, NN3, and NN4 all have CP-CP sessions only with NN2. A TDU from NN1 will be sent to NN2 and, after receipt, forwarded to NN3 and NN4. Each network node receives only one copy of the TDU. When CP-CP connectivity between network nodes is extended, then the number of TDUs flowing through the network will increase.

During LU-LU session establishment, the end nodes report to their network node servers their connection with the VRN. This information is carried in the ***TG vectors***. The TG vector describing the "link" to the VRN allows the network node server responsible for route computation to determine that two nodes can communicate directly.

For further information on the methods by which resource identifiers are used to locate resources, see Chapter 6, "Establishing Routes Through the Network." This chapter discusses the structure of resource identifiers and how networks use them for resource identification and control.

Names in SNA Networks

In all SNA networks, names identify both network resources and the networks themselves. Names fall into three categories:

- Network identifiers
- Network names
- Network-qualified names.

Network Identifiers

A ***network identifier*** (network ID) is an 8-character alphanumeric name that uniquely identifies a network. A network ID is assigned during the system-definition process. For proper routing between interconnected networks, or to connect to a switch serving multiple networks, a network's ID must be unique among the interconnected networks. To ensure the uniqueness of a network ID, a network administrator can register the network's ID with IBM's worldwide registry by contacting a service engineer or a marketing representative. The IBM registry ensures that each network ID is unique among those registered with it. Registry standards are consistent with the Open Systems Interconnection (OSI) standards, including OSI country codes, as established by the International Standards Organization (ISO).

Network Names

A ***network name*** is an 8-character alphanumeric identifier. Each CP, PU, LU, link, and link station in an SNA network can have a network name. Network names eliminate the need for application programs, network operators, and workstation operators to know the locations of different resources within the network. Assigning network names is part of the system-definition process.

Network names for LUs and CPs must be unique within a network name space. (Other network names have significance only within the network's local domain.) Consistent naming conventions should be used to ensure that duplicate network names are not assigned either initially (during system definition) or later (when the network is reconfigured). During backup or recovery of failing resources, names created under such a convention help network operators identify resources and their location in the network.

Network-Qualified Names

A ***network-qualified name*** identifies both a resource and the network in which the resource is located. It is a concatenation of the network ID and the network name of the resource, separated by a period (.).

Addresses in SNA Networks

Three kinds of addresses exist in SNA networks: network addresses, local addresses, and local-form session identifiers. ***Network addresses*** are used to route message units on transmission groups (TGs) interconnecting subarea (T5 or T4) nodes. ***Local addresses*** are used to route message units on TGs interconnecting subarea and T2.0 nodes. ***Local-form session identifiers*** (LFSIDs) are used to route message units on TGs interconnecting subarea and T2.1 peripheral nodes, and on TGs interconnecting LEN and/or APPN nodes.

Network Addresses

Network addresses uniquely identify the system services control points, logical units, physical units, links, and link stations in a subarea network. The network address of each network resource consists of a ***subarea address*** and an ***element address***.

Nonextended Network Addressing

Prior to the inclusion of extended network addressing into SNA, a network address was 16 bits long. Figure 37 shows the format of the 16-bit network address.

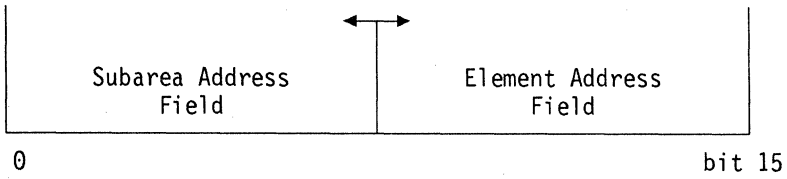


Figure 37. Format of a 16-Bit Network Address

The subarea address field of the 16-bit network address ranges from 1 to 8 bits. The number of remaining bits determines the maximum number of resources that can be defined within each subarea. The number of bits for the subarea address field and the number of bits for the element address field is called the ***network***

address split. The address split chosen for the subarea and element address fields must remain constant throughout the network.

Extended Network Addressing

With VTAM version 3 and NCP version 4, network addresses were extended from 16 bits to 23 bits. The increased size provided additional addresses for users with larger networks. An extended network address uses a fixed 8-bit subarea address field to address up to 255 subarea nodes per network. The element address field of the extended network address uses 15 bits. (Bit 0 is not used.) This increased size permits the assignment of up to 32,768 element addresses in each subarea, enabling subarea network designers to attach more resources to each T4 and T5 node. Figure 38 shows the format of the 23-bit network address.

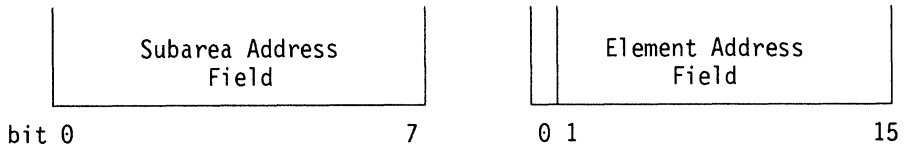


Figure 38. Format of a 23-Bit Network Address

Extended Subarea Addressing

With VTAM V3R2 and NCP V4R3.1, network addresses were extended still further, from 23 bits to 48 bits. The extended subarea address uses a fixed 32-bit subarea address field (16 of which are currently supported) allowing up to 65,535 subarea nodes per network. (The element address field of the extended subarea address still uses 15 bits, allowing up to 32,768 element addresses per subarea.)
Figure 39 shows the format of the 48-bit network address. In the 4-byte subarea address field, only the last two bytes are used.

Figure 39. Format of a 48-Bit Network Address

Subarea Addresses

Network addresses are based on the division of a subarea network into subareas. During system definition a unique number is assigned to every subarea node (T5 and T4 node) in the network. This number becomes the subarea address for:

- Network resources in that subarea node
- Network resources in peripheral nodes that are attached to that subarea node
- Links and link stations adjacent to that subarea node.

The unique number that is assigned to each subarea node becomes the subarea address for all the network resources in that subarea. Path control elements in subarea nodes use this subarea address to route message units between subareas. For example, consider the network configuration in Figure 40. Suppose that subarea node A is assigned the number 3, subarea node B is assigned the number 5, and subarea node C is assigned the number 17. Then the subarea address for all the network resources in the subarea that contains node A will be 3.

Similarly, the network resources in the subarea that contains node B will all have a subarea address of 5, and the network resources in the subarea that contains node C will all have a subarea address of 17.

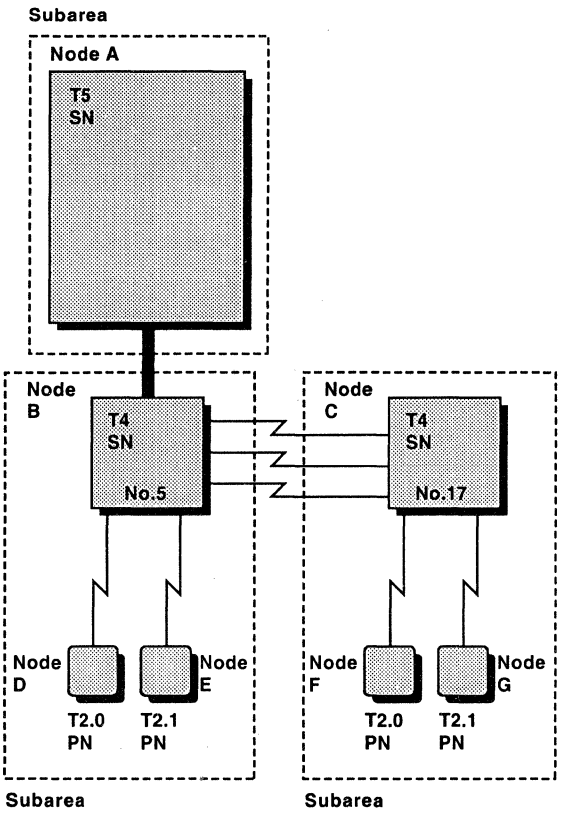


Figure 40. Assigning Subarea Addresses

Element Addresses

Element addresses identify network resources in a subarea. Path control elements in the destination subarea node use the element address field of the network address to route message units to the destination network accessible unit (NAU). Whereas each subarea address is a unique number in the network, an element address is unique only within each subarea.

Element addresses are not assigned by network systems personnel. SNA access methods and network control programs assign element addresses:

- During system generation
- During network reconfiguration
- During network activation for resources that are accessed over a switched SDLC link
- When initiating parallel LU-LU sessions.

SNA requires that certain subarea network resources always use the same element address. Figure 41 shows these constant element addresses.

		Network Resource	Element Address
NODE TYPE	5	PU SSCP LUs	0 1 2 and beyond
	4	PU LU	0 1 and beyond

Figure 41. Constant Element Addresses

SNA access methods and network control programs initially assign element addresses during system generation. The sequence in which these programs encounter the resource-definition statements for network resources determines the order in which they assign the element addresses for the resources.

Local Addresses

Whereas network addresses uniquely identify resources across subarea networks, ***local addresses*** uniquely identify resources only in T2.0 nodes. A local address for a T2.0 node resource is not the same as the element field in the network address for that resource, and it is unique only within the T2.0 node. Local addresses are used by the boundary function component in routing message units on the TGs connecting subarea nodes to T2.0 peripheral nodes. For further information on routing by the boundary function component, see "Routing in Subarea Networks" in Chapter 6, "Establishing Routes Through the Network."

Local-Form Session Identifiers

A ***local-form session identifier*** (LFSID) uniquely identifies a session stage in the pair of adjacent LEN or APPN nodes the session stage interconnects. On the transmission group between the pair, the LFSID is encoded in the FID2 transmission header (TH) as a 17-bit value. Each session stage between the session endpoints uses its own LFSID. An intermediate node along the session path is said to do *address swapping* as it forwards data received on an input session stage to the output stage.

Enhanced Session Addressing in HPR

An enhanced addressing algorithm has been developed for sessions flowing through HPR subnets. A FID5 transmission header is used to transport the 4-byte session address. Two addresses are associated with each session. Each LU (or CP for CP-CP sessions) assigns the address to be used on PIUs it receives. The primary logical unit (PLU), which is the sender of the BIND, assigns an address and sends it to the secondary logical unit (SLU) in the FID5 transmission header on the BIND request. The SLU, after having received the BIND, assigns its address and sends it in the BIND response back to the PLU. Data flowing from the PLU to the SLU will then use the address assigned by the SLU. Likewise, data flowing from the SLU to the PLU will carry the address assigned by the PLU.

Since multiple sessions of the same class of service (COS) can be multiplexed onto a single RTP connection, session addresses must be unique at least per RTP connection in a node. This is necessary to allow connection endpoints to identify data

flowing over the RTP connection based on RTP connection identifier and the individual session address.

Defining Shared Control of Resources in Subarea Networks

In a single-domain subarea network, there is only one domain to identify because there is only one SSCP. All the network resources are defined to that one SSCP. The SSCP then activates those resources in response to network-operator commands or access-method resource-definition statements.

In a multiple-domain subarea network, the resources defined to each SSCP determine the domains in the network. As with a single-domain network, each network resource is defined to an SSCP. Each SSCP in the network may control (activate and deactivate) an equal number of resources, one SSCP may control the majority of network resources, or more than one SSCP may share control of some resources.

To provide flexibility for both normal operations and backup recovery procedures, some network resources can be defined to more than one SSCP. The architecture permits SSCPs to share some network resources concurrently, some serially, and some not at all. Resources residing in a T5 node are controlled only by their local SSCP.

Shared control of network resources enables an enterprise to:

- Back up one SSCP by another to increase network availability
- Partition control of a network by use rather than by the physical location of resources
- Shift control of network resources to different SSCPs at various times of the day in response to changing traffic loads.

Concurrent Sharing

Concurrent sharing means that more than one SSCP can simultaneously control the same resource. Multiple SSCPs can concurrently share control of:

- Physical units in type 4 nodes
- Nonswitched SDLC link connections between subarea nodes
- Link stations for the nonswitched SDLC links between subarea nodes.

If one of the SSCPs fails, the other SSCPs that have activated the T4 nodes are notified. This notification serves as a signal for the other SSCPs to activate, and thus establish control of, the related peripheral resources that were previously owned by the failing SSCP.

For example, Figure 42 shows a configuration in which four SSCPs share a physical unit in a type 4 node. The type 4 node, node *E*, belongs to all four domains.

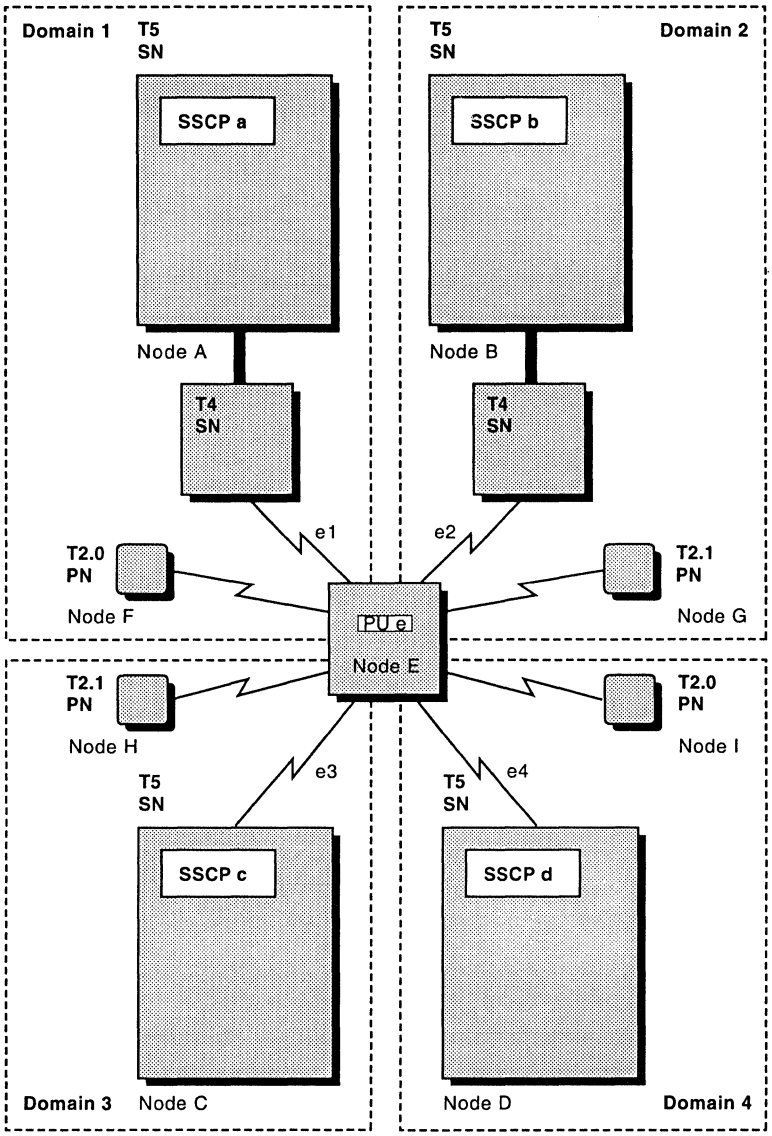


Figure 42. Concurrent Sharing of Network Resources

If PU *e* is defined to SSCPs *a*, *b*, *c*, and *d*, all four SSCPs can activate, and thus control, PU *e*. Any SSCPs that share control of PU *e* can also share control of the nonswitched SDLC link connections *e1*, *e2*, *e3*, and *e4*.

Serial Sharing

Serial sharing means that only one SSCP at a time can control a resource. When that SSCP relinquishes control, another SSCP can assume control. SSCPs can serially share four types of resources:

- Logical units in peripheral nodes
- Physical units in peripheral nodes
- Links that connect peripheral nodes to subarea nodes
- Switched SDLC links between subarea nodes.

Any of the SSCPs that concurrently share control of a type 4 node can serially share control of the resources in a peripheral node that is attached to that type 4

node. For example, any of the four SSCPs in Figure 42 can activate the resources (PUs and LUs) in peripheral nodes *F*, *G*, *H*, and *I*, but only one can do so at a time.

Share Limit

Each network resource has a ***share limit*** that specifies the maximum number of SSCPs that can share control of that resource. Resources that can be shared only serially have a share limit equal to 1. Resources that can be shared concurrently have a share limit greater than 1. (An implementation, such as for a T4 node, may include its PUCP as counting toward the share limit for a particular node resource.) The desired share limit for each resource is determined by resource-definition statements.

Reconfiguring a Subarea Network

A subarea network's configuration is statically defined in the form of tables that are loaded into access methods and network control programs during a system generation. However, the configuration of the network is unlikely to remain exactly as originally specified. When the configuration of a subarea network changes, a reconfiguration effort is required. Both scheduled and unscheduled changes are probable.

Scheduled Changes

There are two ways to schedule network configuration changes. One way is to recode one or more of the tables that define the configuration to reflect configuration changes. This method requires a new system generation. System generation requires that at least part of the network be deactivated for the amount of time necessary to load the revised information into the programs.

The other way to schedule configuration changes is through dynamic reconfiguration. ***Dynamic reconfiguration*** enables network operators to selectively add, move, or delete network resources in peripheral nodes without disrupting other network activity. The commands a network operator issues to dynamically reconfigure a network differ depending on the particular programs that are resident in the T5 subarea nodes. Typically, these network operator commands modify the original configuration data set (generated from resource-definition statements during the last system generation) to reflect a new resource hierarchy. The SSCP uses the modified data set the next time it activates or deactivates network resources. A network operator can activate another data set to terminate the modifications and restore the original configuration data set.

Peripheral node resources that are connected to a subarea node by nonswitched links can be reconfigured through dynamic reconfiguration. The PUs and LUs in the peripheral nodes that are being reconfigured must be inactive (have no active sessions) during reconfiguration.

Unscheduled Changes

Unscheduled changes to a network configuration are inevitable. The hardware or software in a node might fail, or links between adjacent nodes might fail. Network operators can sometimes circumvent these failures by reconfiguring the network. However, when an SSCP can no longer communicate with a type 4 node, the network operator cannot enter any commands to reconfigure the network resources related to that type 4 node.

When an SSCP can no longer communicate with a type 4 node, the network control program in the type 4 node begins automatic network shutdown, if required. ***Automatic network shutdown*** deactivates some or all of the resources adjacent to a type 4 node in an orderly manner. The installation manager can optionally define that traffic for existing LU-LU sessions not be disrupted by automatic network shutdown. When the network control program regains its ability to communicate with an SSCP, the SSCP reactivates any resources that were affected by the automatic network shutdown procedure. How a network control program detects the loss of communication with an SSCP, and which resources it deactivates, depends upon the specific network control program that resides in the type 4 node.

Defining Gateways in Subarea Networks

The ***SNA network interconnection*** (SNI) architecture enables end users in different subarea networks to communicate with each other while enabling each network to be configured, defined, and managed independently. This section explains why interconnected subarea networks do not require (1) a common network address structure using all unique network addresses and (2) a common network naming convention to maintain unique resource names.

SNA network interconnection provides a ***gateway*** between two or more subarea networks that permits each network to maintain the network address scheme most appropriate to its specific configuration. The subarea addresses assigned in one network can be duplicated in interconnected networks. Networks that do not yet support extended addressing can be interconnected to networks that do. In addition, networks that have different network address splits can be interconnected.

Gateways also enable interconnected subarea networks to maintain their own network naming conventions. The network names assigned in one network can be duplicated in interconnected networks.

Figure 43 shows three configurations for interconnecting subarea networks. The first configuration shows how a single gateway can interconnect two or more networks. The second configuration shows how two gateways can interconnect two networks. In this configuration, the only resources that network B contains are the links that connect the two gateways. The third configuration shows how multiple, parallel gateways can interconnect two networks. The gateway itself consists of a gateway node and one or more gateway SSCPs.

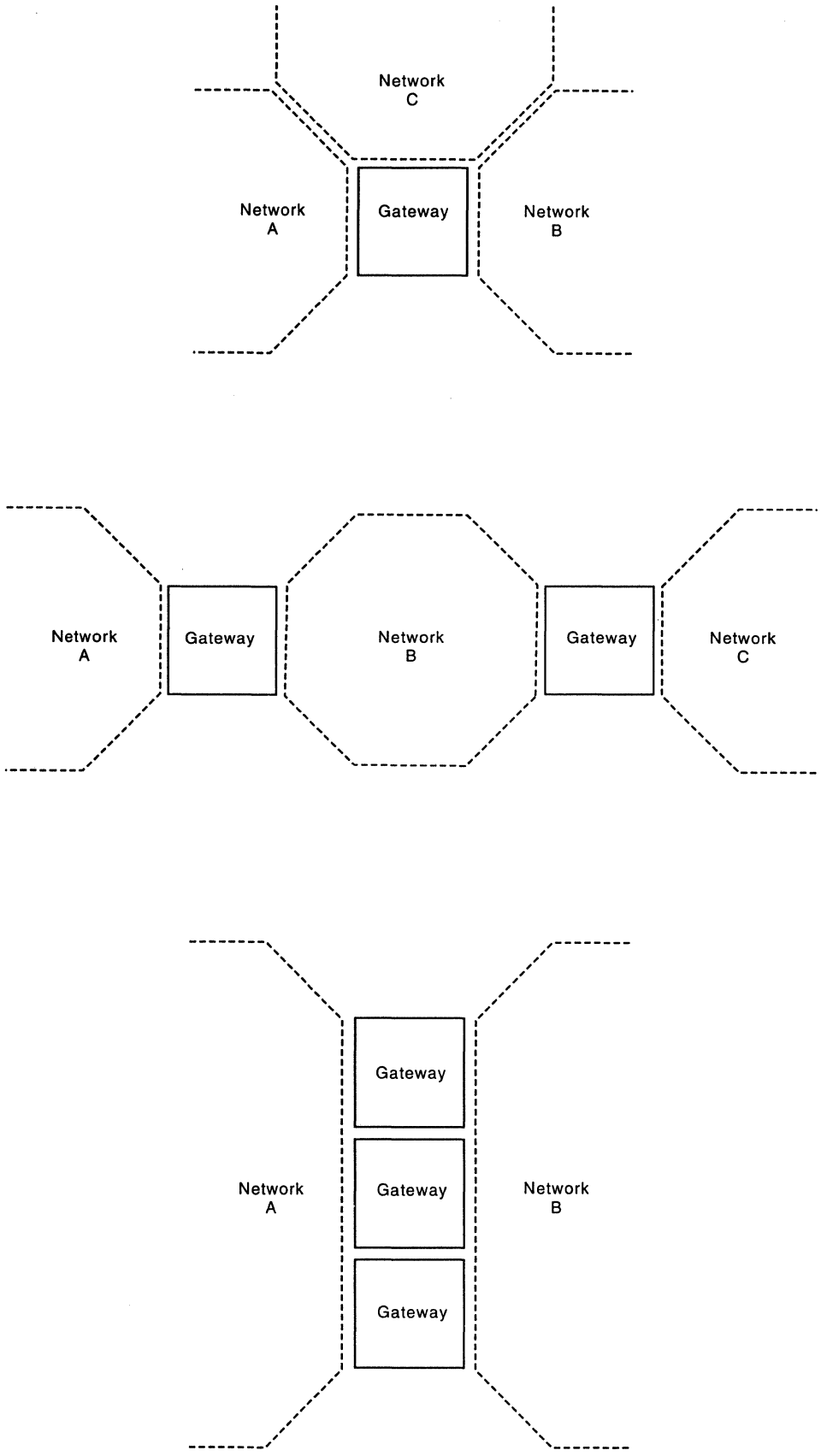


Figure 43. Interconnected Subarea Network Configurations

Subarea Gateway Nodes

A network resource is normally represented in different subarea networks by different network names and addresses. A ***subarea gateway node*** is a T4 node that translates between the network name and address used by one subarea network to represent a resource, and the network name and address used by another subarea network to represent the same resource.

The gateway node is assigned a unique subarea address in each of the interconnected networks. None of the interconnected networks is aware that the gateway node is performing name and address translations. Viewed from any one of the interconnected networks, the gateway node is a part of that network.

Because a network name or address is unique only within a given subarea network, one subarea network identifies resources in another subarea network by alias names and addresses. An ***alias name*** or an ***alias address*** identifies in one network a resource that resides in an interconnected network. Alias names are assigned during the system definition process. Alias addresses are assigned during *cross-network* session initiation. A gateway node:

- Contains a gateway function component
- Recognizes network names and addresses from two or more interconnected networks
- Is identified by a network address in each interconnected network
- Contains one path control instance for each network interconnected through the gateway.

The ***gateway function*** component acts as an intermediary between path control elements within the gateway node. It receives a message unit from the path control element in the node representing the sending network, translates the names and addresses, and routes the *transformed* message unit to the path control element in the node representing the receiving network.

Gateway SSCPs

Gateway SSCPs maintain directories of real and alias network names and prepare gateway nodes to perform network name and address translation. Every gateway includes at least one gateway SSCP. However, a network designer can choose to have more than one gateway SSCP per gateway. The share limit assigned to the PU in the gateway node determines the maximum number of gateway SSCPs in that gateway.

One advantage of configuring multiple gateway SSCPs is that the network administrators of the interconnected networks can maintain independence in their assignment of alias names. As an example, consider two networks, A and B, interconnected by a gateway with two gateway SSCPs, one in each network. Administrators for network A need to assign alias names only for LUs in network B, and administrators for network B need to assign alias names only for LUs in network A. If there were only a single gateway SSCP, for example in network A, then administrators for network A would have to assign alias names for LUs in both network A and network B.

For further information on routing by the gateway node, and interaction with the gateway SSCP, see "Routing in SNI Gateways" in Chapter 6, "Establishing Routes Through the Network."

Network Identifiers in Gateways

To guarantee that a network address or name is unique among interconnected subarea networks, gateways use network identifiers. A network identifier (ID) uniquely identifies a network within a set of two or more interconnected networks. Gateway nodes and gateway SSCPs use network identifiers to qualify the names and addresses that they exchange with one another. A unique network ID is assigned to each interconnected network during the system definition process.

A ***network-qualified address*** identifies a real network address and the network in which that address is valid. A ***network-qualified name*** identifies a real network name and the network in which that name is valid. Gateway SSCPs and gateway nodes use network-qualified names and addresses. Non-gateway resources in one subarea network do not use network-qualified names or addresses to identify resources in an interconnected network. Instead, they use alias network names and addresses.

Defining APPN Subnets Containing Peripheral Border Nodes

Topology Isolation

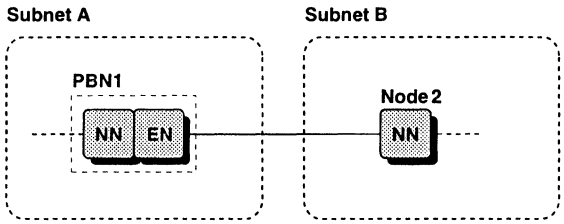
A benefit of having topology subnets is that the number of TDUs flowing through topology subnets will be lower; therefore, the storage requirements for the network topology database in network nodes in each of the subnets is reduced. This reduces network flows and allows network nodes with limited resources to participate in APPN networks. The topology databases of two adjacent subnets connected via a border node remain distinct. A peripheral border node identifies the TGs that provide connectivity to nonnative network nodes or extended border nodes as intersubnetwork TGs. For further information on TDUs and topology information, see "Topology Database" in Chapter 5, "Activating the Network."

APPN Subnet Configurations Containing Peripheral Border Nodes

The node image a border node presents depends on its relationship with the subnet it is linked to. The following are possible configurations of a peripheral border node:

1. If one subnet contains a peripheral border node that is connected to a network node in another (nonnative) subnet, the peripheral border node acts as an APPN end node. The two subnets are connected via an intersubnetwork TG.

   Figure 44 illustrates a peripheral border node in one subnet connected to a network node in another subnet.



Legend

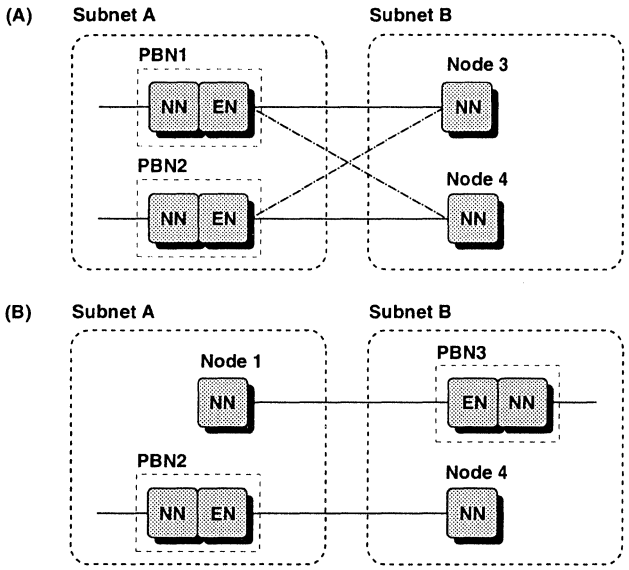
PBN = Peripheral border node
NN = Network node
EN = End node

Figure 44. Subnet Connection with a Single Peripheral Border Node connecting to a Network Node

- From subnet A's (the native subnet) viewpoint, PBN1 is a network node.
- From subnet B's (the nonnative subnet) viewpoint, PBN1 is an APPN end node.

A peripheral border node connection to a nonnative network node supports session establishment capability between two resources located in either subnet. However, if PBN1 in subnet A is connected to an APPN end node (instead of a network node) in subnet B, intersubnet session routing between the two subnets is not supported.

Figure 45 illustrates parallel subnet connections with peripheral border nodes. Two subnets can be connected by multiple peripheral border nodes in parallel. The benefit of this configuration is enhanced internetwork availability and bandwidth. Multiple peripheral border nodes can reside in one or in both subnets.



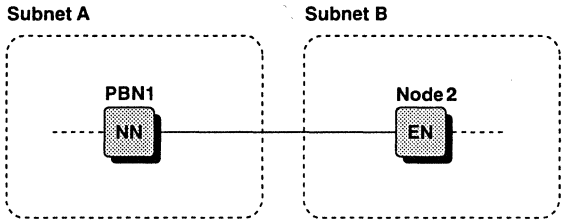
Legend

——— = With CP-CP session
–·–·– = Without CP-CP session
PBN = Peripheral border node
NN = Network node
EN = End node

Figure 45. Parallel Subnet Connection with Peripheral Border Nodes

2. If one subnet contains a peripheral border node that is connected to an APPN end node in another (nonnative) subnet, the peripheral border node acts as an APPN network node. As mentioned earlier, in this case, the partner subnet does not use the peripheral border node function; a casual connection exists.

   Figure 46 illustrates a peripheral border node in the native subnet connected to an APPN end node in the nonnative subnet.



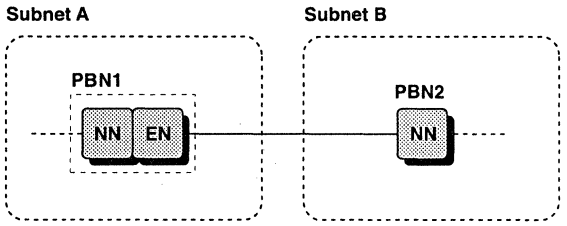
Legend

PBN = Peripheral border node
NN = Network node
EN = End node

Figure 46. Subnet Connection with a Single Peripheral Border Node Connecting to an APPN End Node

3. If two subnets each contain a peripheral border node, the image that each peripheral border node presents is determined during XID exchange. The two subnets are connected via an intersubnetwork TG.

   Figure 47 illustrates a connection between two peripheral border nodes. When two peripheral border nodes interconnect, because of their asymmetrical nature, it is decided during XID exchange which peripheral border node presents the network node image and which presents the APPN end node image.



Legend

PBN = Peripheral border node
NN = Network node
EN = End node

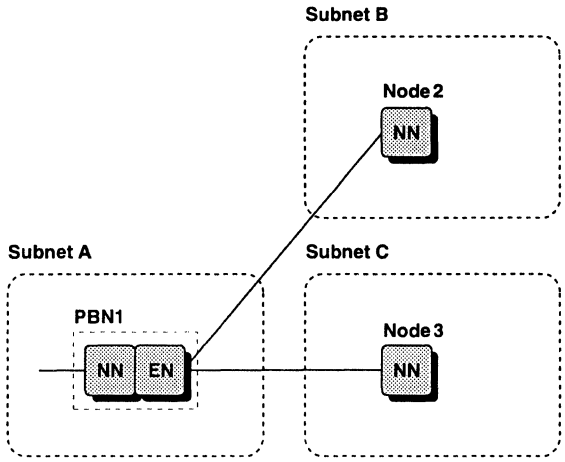
Figure 47. Peripheral Border Node to Peripheral Border Node Connection

In Figure 47, the peripheral border node roles have been negotiated as follows:

- PBN1 in subnet A (native subnet) will always present an APPN end node image to PBN2 in subnet B (nonnative subnet).
- PBN2 in subnet B will always present a network node image to PBN1 in subnet A.

Figure 48 illustrates one peripheral border node connecting many subnets. In this configuration, Node 2 (in subnet B) and Node 3 (in subnet C) each serve

as a network node server of PBN1 for their respective subnets. A session cannot be established between subnet B and subnet C via subnet A.

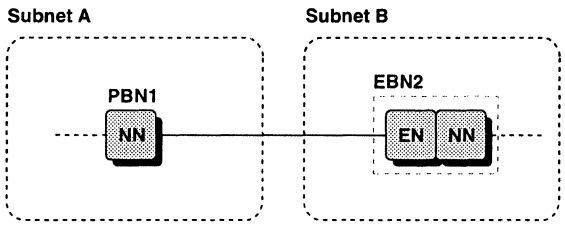


Legend

PBN = Peripheral border node
NN = Network node
EN = End node

Figure 48. One Peripheral Border Node Connecting Many Subnets

4. If one subnet contains a peripheral border node and another (nonnative) subnet contains an extended border node, the peripheral border node always acts as an APPN network node and the extended border node always acts as an APPN end node. Topology information is not exchanged between the two subnets since end nodes do not participate in topology information exchange. Therefore, the topology databases of the two subnets remain distinct.



Legend

PBN = Peripheral border node
EBN = Extended border node
NN = Network node
EN = End node

Figure 49. Subnet Connection between a Peripheral Border Node and an Extended Border Node

In Figure 49, the node roles are as follows:

- PBN1 in subnet A (native subnet) will always present a network node image to subnet B (nonnative subnet).
- From subnet A's viewpoint, EBN2 in subnet B is an APPN end node.
- From subnet B's viewpoint, EBN2 in subnet B is a network node.

Defining Subnets Containing Extended Border Nodes

Topology Isolation

An extended border node is not involved in topology update flows and algorithms of nonnative subnets. In its native network, an extended border node is involved in the normal topology update flows and algorithms since the extended border node presents a network node image to native partner nodes.

Following the intersubnetwork link activation between the extended border node and a nonnative border node or network node, no TDUs are propagated into the nonnative subnet across the intersubnetwork link. However, a TDU identifying the intersubnetwork link, marked as quiescing, is propagated into each extended border node's subnet. This ensures that a node in the native subnet never uses an adjacent, nonnative extended border node as an intermediate node to establish a session to another node in the native subnet. Although topology will be isolated between subnets, directory information such as CP names of CP(DLUs) and CP(OLUs) with TG vectors representing the appropriate intersubnetwork TGs are exchanged in the Locate flows during session initiation between subnetworks. This is similar to session initiation using peripheral border nodes.

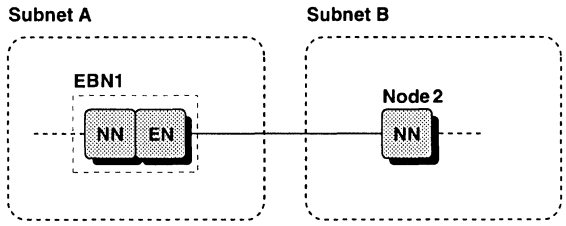
An extended border node, like a peripheral border node, identifies itself as a border node in its TDU and broadcasts the TDU into its native subnet.

APPN Subnet Configuration Containing Extended Border Nodes

An extended border node can partition a single net-ID subnet into two or more isolated topology domains, or ***clusters*** (or topology subnets). An advantage that extended border nodes have over peripheral border nodes is that they allow the subnets containing endpoint LUs of a session to be located in nonadjacent subnets. One or more intermediate subnets containing extended border nodes are located between the endpoint subnets. Extended border nodes, unlike peripheral border nodes, support intermediate network routing. The following are possible configurations of an extended border node:

1. If one subnet contains an extended border node that is connected to a network node (or even a peripheral border node) in another (nonnative) subnet, the extended border node appears as an APPN end node to the partner subnet. The two subnets are connected via an intersubnetwork TG.

   Figure 50 illustrates an extended border node in the native subnet connected to a network node in the nonnative subnet.



Legend

EBN = Extended border node
NN = Network node
EN = End node

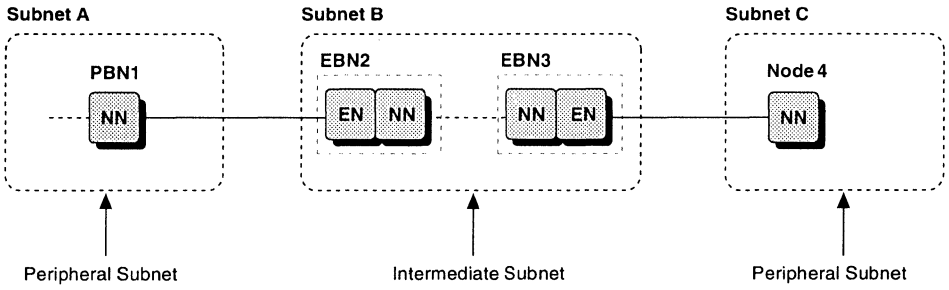
Figure 50. Subnet Connection between Extended Border Node and Network Node

In Figure 50, the node roles are as follows:

- From subnet A's (the native subnet) viewpoint, EBN1 is a network node.
- From subnet B's (the nonnative subnet) viewpoint, EBN1 is an end node.
- If Node2 in subnet B was a PBN, it would always act as a basic network node.

2. If three subnets are interconnected, with the intermediate subnet containing an extended border node, one peripheral subnet containing a peripheral border node and the other peripheral subnet containing a network node, the node roles are as described below. Each peripheral subnet can contain a peripheral border node, an extended border node, or a basic network node. The intermediate subnet can contain only extended border nodes in order to permit intermediate network routing.

   Figure 51 illustrates three subnets interconnecting containing both peripheral and extended border nodes.



Legend

EN = End node
NN = Network node
PBN = Peripheral border node
EBN = Extended border node

Figure 51. Multiple Subnets Interconnecting Both Extended Border Nodes and Peripheral Border Nodes

In Figure 51, the node roles are as follows:

- PBN1 in subnet A is always a network node.
- From subnet A's viewpoint, EBN2 in subnet B is an APPN end node.
- From subnet B's viewpoint, EBN2 in subnet B is a network node.

- From subnet B's viewpoint, EBN3 in subnet B is a network node.
- From subnet C's viewpoint, EBN3 in subnet B is an APPN end node.
- Node4 in subnet C is a basic network node.

Chapter 5. Activating the Network

This chapter explains network activation within subarea and APPN networks.

The Meaning of "Network Activation"

Network activation has different meanings depending on whether the network is hierarchical or peer-oriented. The activation of subarea networks involves activating groups of nodes and links by operator command. It is controlled by predefined tables established during system generation. The activation of APPN networks, by contrast, is distributed and subject to individual node requirements. APPN nodes become active network components by the decisions of their local operators to join the network.

Activating Nodes

Activation of an SNA network begins when the hardware is turned on and network control software is loaded. Such software includes operating systems, access methods, network control programs, and other communication software. Nodes become operational when their hardware and software are functioning properly. Once nodes become operational, their control points activate the network resources within their control.

Recall that a T5 node controls the nodes and links within its domain. Activating a T5 subarea node, therefore, normally results in the activation of much of its domain. (One exception would be switched links and the nodes connected to them.) Nodes other than T5 nodes, however, control only their own local resources. Therefore, activating any type of node other than a T5 node does not automatically result in the activation of other nodes.

Activating Links

Control points (CPs) activate, control, and deactivate links through the link stations in the node. To activate a link, configuration services in the CP causes data link control to issue link-level commands to the adjacent link station for that link. Once a link between adjacent nodes is activated, that link can carry session traffic.

The nodes at both ends of a link participate in activating a link. In subarea networks, the control points in T4 and peripheral nodes prepare their link stations so that T5-initiated link activations can occur. The SSCP in a T5 subarea node first activates the links to its adjacent nodes. The SSCP then directs the PUs in adjacent T4 nodes to activate the links to their peripheral nodes. The CPs in peripheral nodes autonomously activate the links connecting them to other nodes, as do any LEN or APPN nodes in APPN networks.

Phases of Link Activation

In the most general case, link activation involves three phases: connect, prenegotiation, and contact. The ***connect phase*** allows initial establishment of communication between nodes. It is during this phase that dialing and answering on switched links take place. In addition, modems at each end of the link *equalize* by exchanging *training sequences*. (Equalization must take place before a modem can give permission to its link station to begin transmission.)

The ***prenegotiation phase*** begins with a link-level poll to determine if the adjacent node is active. APPN or LEN nodes use a *null* Exchange ID (XID) poll to determine if the adjacent node is active. If the adjacent node supports APPN or LEN

protocols, it returns an XID type 3 (XID3) with its Exchange State indicators set to "prenegotiation."

During the ***contact phase***, link station roles, link characteristics, and certain node capabilities are conveyed by XID commands and responses. The contact phase is completed when one link station sends an appropriate mode-setting command to the other and receives an ***Unnumbered Acknowledgment*** (UA) in response.

Exchange Identification Commands

Exchange Identification (XID) commands and responses enable communicating link stations to establish mutually acceptable link station roles and link characteristics and to convey certain node characteristics and capabilities before they transmit data. XID format 1 is used on SDLC links interconnecting subarea nodes and on links connecting T2.0 nodes to boundary nodes. Format 2 is used on SDLC or S/370 data channel links interconnecting subarea nodes. Format 3 is used on links interconnecting nodes supporting APPN or LEN protocols.

Link station *role* refers to whether a link station is the primary or the secondary link station on a link. The ***primary link station*** has a controlling role over the link connection and the ***secondary link station***. On a multipoint SDLC link configuration, there are multiple secondary link stations controlled by a single primary link station. Hence, the link connection is shared by multiple links. Each transmission on a multipoint link connection is between the primary link station and a secondary link station. There are no transmissions between secondary link stations.

Note: The terms *primary* and *secondary* are used differently with respect to link stations using asynchronous balanced mode (ABM) protocols. Such stations choose primary and secondary roles, during XID negotiation, only to determine how a field in their exchanged transmission headers is to be set, not to control the link connection.

Examples of node and link characteristics include:

- Segmentation and reassembly capability of the node
- Maximum BTU sizes allowed on the link
- Transmission group numbers for links between subarea nodes and between APPN nodes
- End node or network node capability for APPN nodes.

XID3 exchanges also communicate changes after the contact phase is completed. These are called ***nonactivation XID exchanges***. The XID3s sent in these exchanges have their Exchange State indicators set to "nonactivation exchange." A nonactivation XID exchange may occur, for example, because a node wants to poll an adjacent node to see if the adjacent node, and the TG connecting to it, are still active.

XID Negotiation

Link station roles and certain link characteristics are *negotiable* on links between particular node types. The capability of certain nodes to negotiate eases the tasks of system definition and network maintenance because parameters that are negotiated need not be predefined. Negotiation takes place during the contact phase through XID command and response exchanges. After XID exchanges on a link

are complete, the nature of all negotiable parameters on that link have been decided.

The determination of which link station on a link will be the primary and which will be the secondary, for example, is negotiable on links connecting certain node types. On links using XID format 2 or 3, the nodes are capable of negotiating link station roles. Otherwise, the roles of link stations must be predefined.

For additional information on the sequence of commands that adjacent link stations exchange and the definition of primary and secondary link stations, refer to *IBM SDLC Concepts*.

Activating Subarea Networks

The activation of a subarea network begins with the activation of the T5 nodes in the network. The system services control points (SSCPs) in the T5 nodes then proceed to activate their respective domains. SSCPs maintain control over network resources through sessions established during network activation. Activating a domain involves activating the links interconnecting nodes within the domain and activating sessions with the physical units and dependent logical units in the nodes.

Activating SSCP-PU Sessions

An SSCP activates sessions with the physical units (PUs) that were defined to it during the system definition process. An SSCP must activate SSCP-PU sessions before the physical units can become active parts of the network. SSCP-PU sessions remain active until network deactivation.

An SSCP communicates with physical units over the SSCP-PU sessions established during network activation. An SSCP needs to communicate with the physical unit in each node in order to control and monitor the resources in that node. Recall that in a T2.1 peripheral node using SSCP-dependent protocols, the control point performs the functions of a PU. To control resources in such a node, therefore, an SSCP communicates with the T2.1 peripheral node CP as it would with a PU in a T2.0 node.

Activating SSCP-LU Sessions

An SSCP also activates sessions with the logical units (LUs) that are defined to it during system definition. However, not all LUs in a subarea network need be defined to an SSCP. Only an SSCP-dependent LU must be defined to an SSCP; an SSCP-independent LU does not need to be defined to an SSCP. As mentioned earlier, ***dynamic definition of independent LUs*** enables an independent LU to activate a session into a subarea network without first having been defined to the network through system generation.

An SSCP must activate SSCP-LU sessions with the dependent LUs in its domain before the LUs can become active parts of the network. An SSCP communicates with dependent LUs over the SSCP-LU sessions established during network activation, or, in the case of a dependent LU on a switched link, during link activation. Dependent LUs must submit requests for LU-LU sessions to an SSCP over an SSCP-LU session. Like SSCP-PU sessions, SSCP-LU sessions remain active until network deactivation.

An SSCP does not activate SSCP-LU sessions with the independent LUs in its domain. Independent LUs can establish sessions between themselves without requesting the mediation of an SSCP. Two independent LUs in separate T2.1 nodes attached to a subarea network using LEN protocols can establish an LU-LU session between them if the LEN boundary nodes to which the T2.1 nodes are connected, and the SSCPs for the boundary nodes, support peer-session passthrough (for independent LU protocols). Peer-session passthrough involves an exchange of messages between SSCPs and boundary functions that enables a BIND request from an independent LU to be accepted into a subarea network.

For further information on the initiation of LU-LU sessions in subarea networks, see "Initiating LU-LU Sessions in Subarea Networks" in Chapter 8, "Transporting Data Through the Network."

Hierarchy of Subarea Network Activation

There is a hierarchy of activation in a subarea network. Subarea network activation begins with the activation of the T5 subarea nodes in the network. The SSCP in each T5 node then proceeds to activate the resources in its domain. The activation proceeds in a hierarchical fashion, from the T5 node outward to attached T4 nodes, and then to the peripheral nodes. A domain can be thought of as a tree-like structure with the T5 node as its root; the links and other nodes in the tree are said to be *down-tree* from the T5 node.

The steps listed below illustrate the network activation hierarchy in a single-domain network configuration. Note that in this example, all nodes have been powered on, and the T4 and peripheral node control points have already prepared their link stations for remote activation. In addition, only one pair of link stations per link connection are shown.

1. Activate the resources in the T5 node as shown in Figure 52.

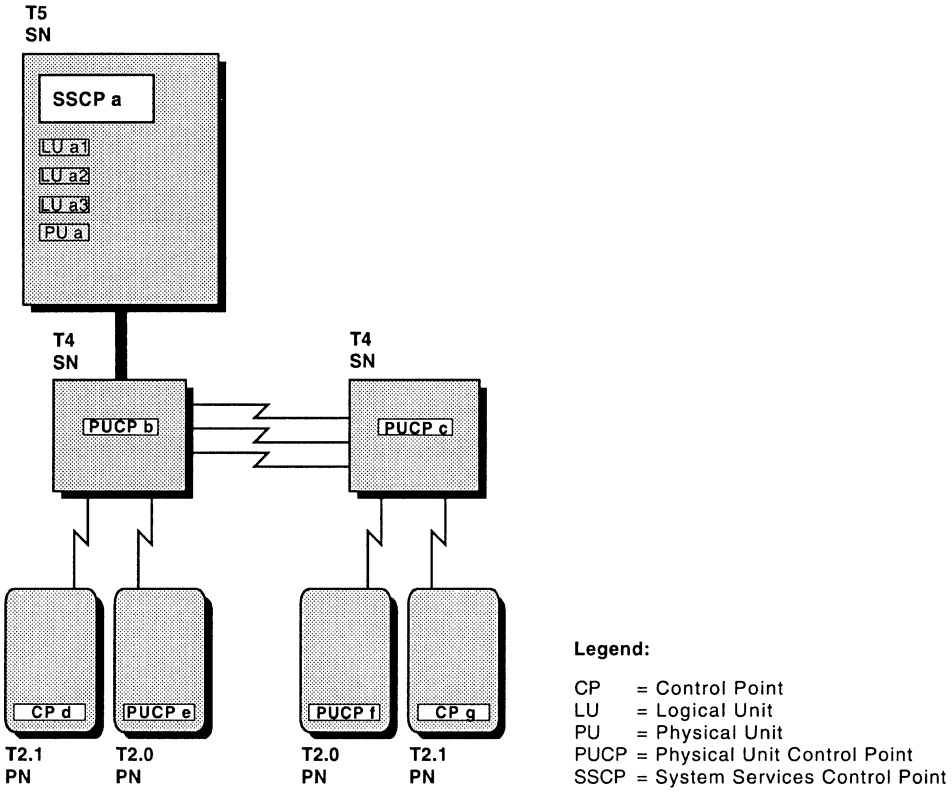


Figure 52. Hierarchy of Subarea Network Activation: Part I

a. PU *a*

   SSCP *a* requests the SSCP-PU session by sending an Activate Physical Unit (ACTPU) request to PU *a*. A positive response from PU *a* completes the activation of the SSCP-PU session.

b. LUs *a1*, *a2*, and *a3*

   SSCP *a* requests the SSCP-LU sessions by sending Activate Logical Unit (ACTLU) requests to LUs *a1*, *a2*, and *a3*. Positive responses from the LUs complete the activation of these SSCP-LU sessions.

2. Activate the link connection that is attached to the T5 node, as shown in Figure 53.

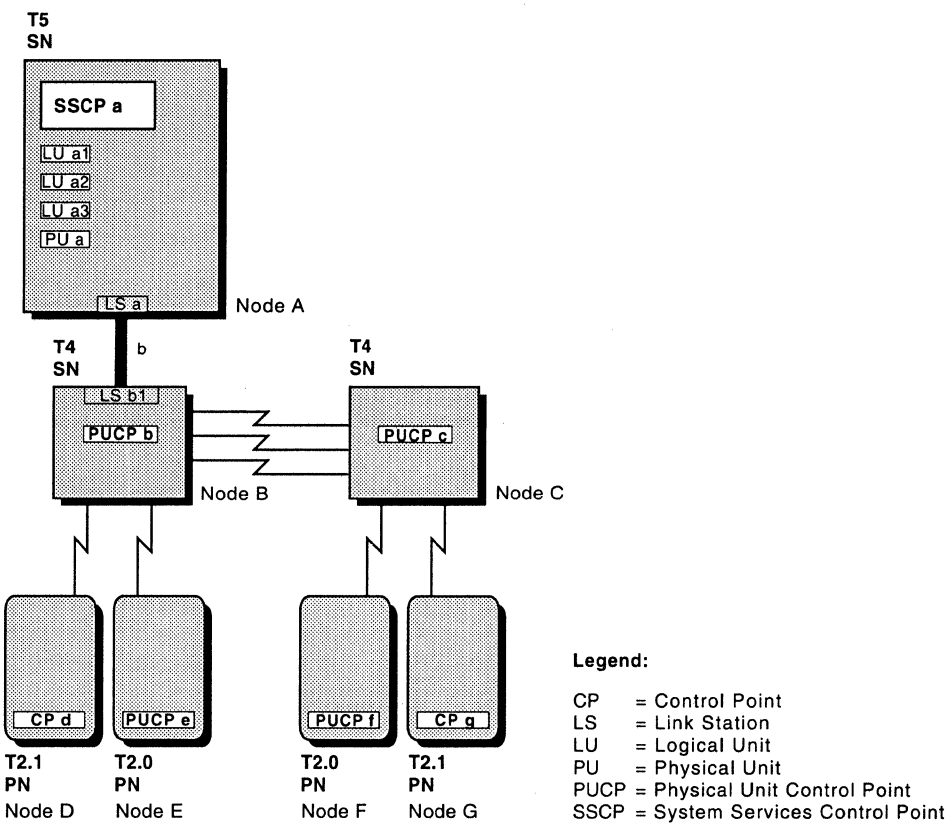


Figure 53. Hierarchy of Subarea Network Activation: Part II

a. Link *b*

1) SSCP *a* sends an Activate Link (ACTLINK) request to PU *a* over the SSCP-PU session.[5]

2) PU *a* informs the SSCP over the SSCP-PU session that link connection *b* is operational.

3) SSCP *a* issues a CONTACT command that requests PU *a* to contact LS *b1*.

4) LS *a* and LS *b1* exchange identifications and the data link protocol commands necessary to activate link *b*.

5) LS *a* informs PU *a* that it has successfully contacted its channel-attached link station.

6) PU *a* returns a CONTACTED command to SSCP *a* to inform the SSCP that LS *b1* has been contacted.

[5] The PUCP and PU in node B perform a comparable sequence

3. Activate the resources in adjacent type 4 node B, as shown in Figure 54.

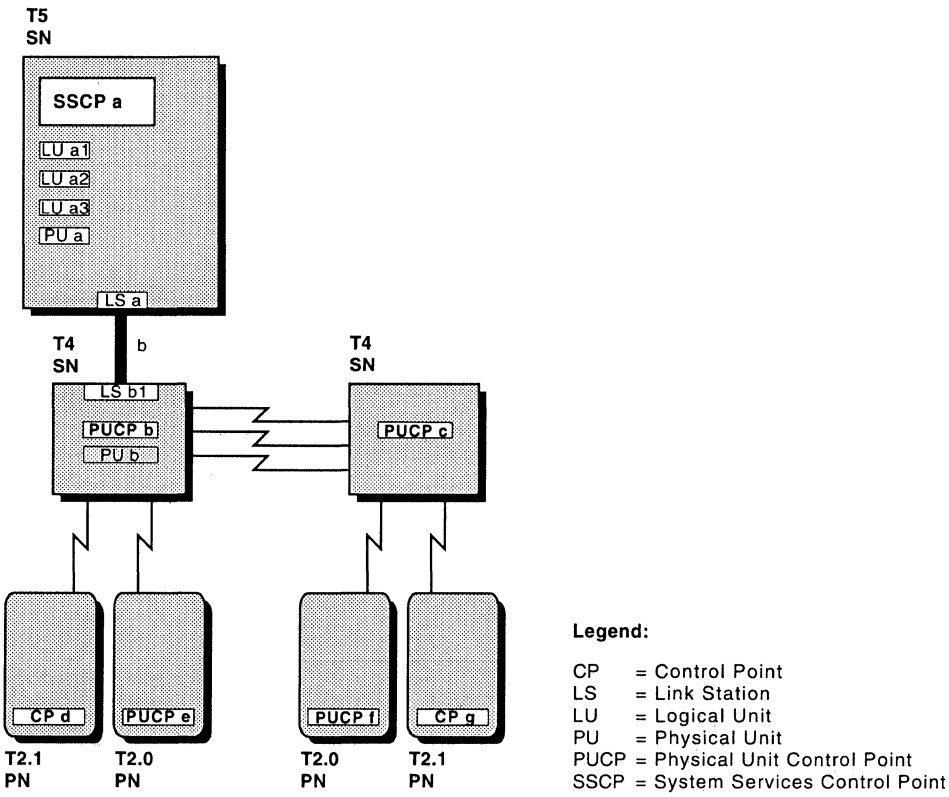


Figure 54. Hierarchy of Subarea Network Activation: Part III

a. PU *b*

The SSCP activates an SSCP-PU session with PU *b*.

4. Activate the down-tree links that are attached to type 4 node B, as shown in Figure 55.

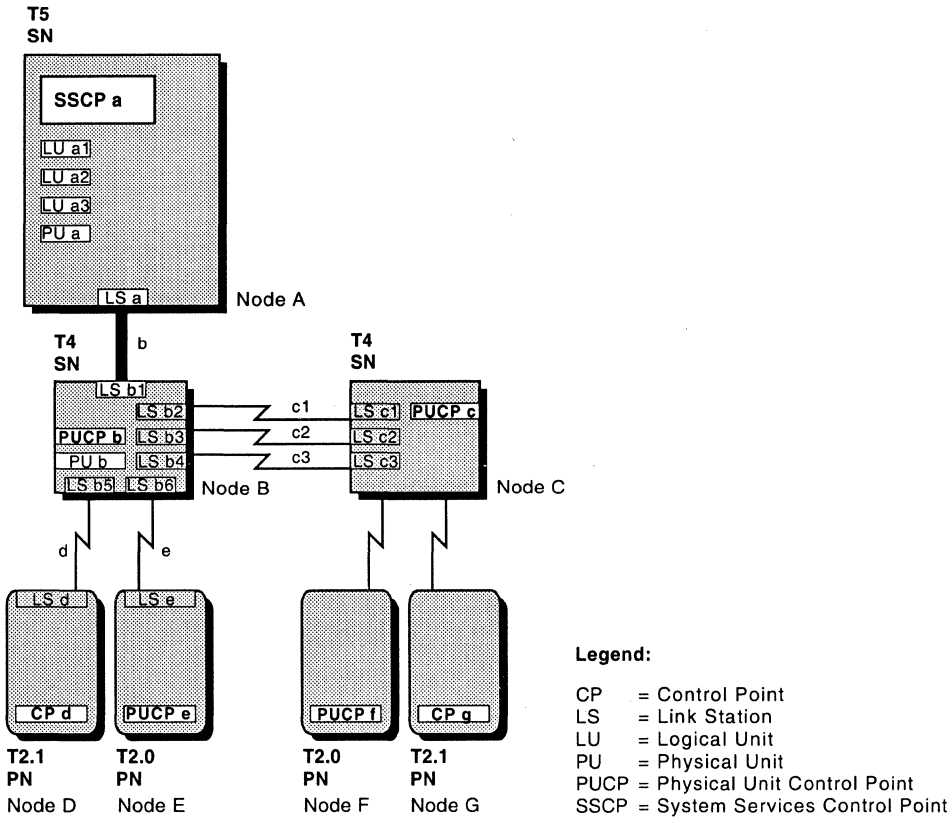


Figure 55. Hierarchy of Subarea Network Activation: Part IV

a. Links *c1*, *c2*, and *c3*

   The SSCP sends PU *b* ACTLINK and CONTACT requests for the links that connect nodes B and C.

   LS *b2* communicates with LS *c1*; LS *b3* communicates with LS *c2*; and LS *b4* communicates with LS *c3* to activate links *c1*, *c2*, and *c3*, respectively.

b. Links *d* and *e*

   The SSCP sends ACTLINK and CONTACT commands to PU *b* to activate links *d* and *e*. LS *b5* communicates with LS *d*, and LS *b6* communicates with LS *e*. PU *b* sends a CONTACTED to the SSCP to tell the SSCP when it has successfully contacted link stations LS *d* and LS *e*.

5. Activate the resources in nodes that are adjacent to type 4 node B, as shown in Figure 56.

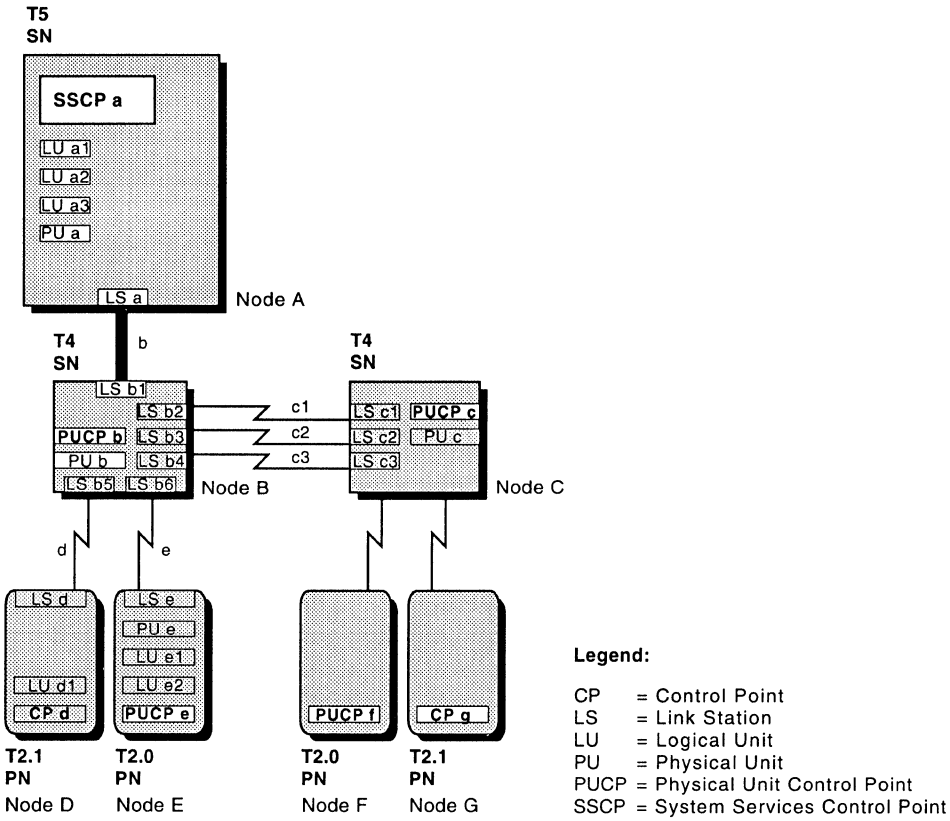


Figure 56. Hierarchy of Subarea Network Activation: Part V

   a. PUs *c*, and *e*, and CP *d*

   b. LU *d1* and LUs *e1* and *e2*

   The SSCP activates SSCP-PU sessions with the physical units in the T4 and peripheral nodes. The SSCP then activates SSCP-LU sessions with the logical units in those nodes. LU *d1* is assumed to be a dependent LU.

6. Activate the links that are attached to type 4 node C, and then activate the resources in the attached nodes, as shown in Figure 57.

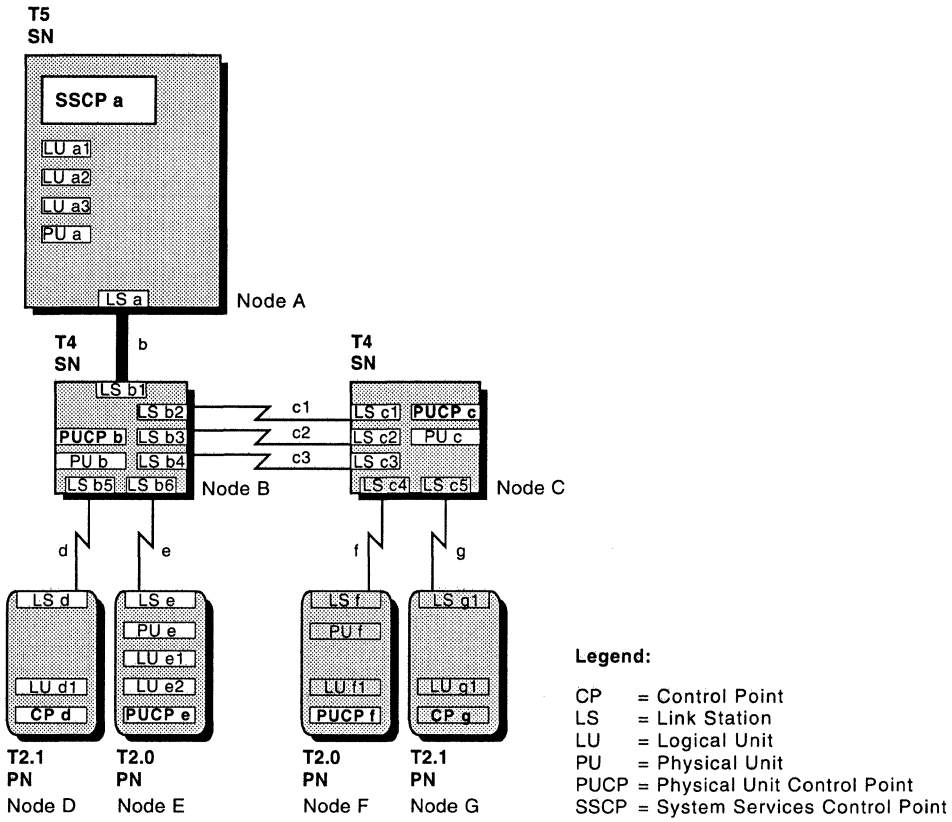


Figure 57. Hierarchy of Subarea Network Activation: Part VI

a. Links *f* and *g*

   The SSCP sends PU *c* ACTLINK and CONTACT requests for these links over the SSCP-PU session.

b. PU *f* and CP *g*

   The SSCP sends ACTPU requests to the PU in the T2.0 peripheral node and the CP in the T2.1 peripheral node.

c. LU *f1* and LU *g1*.

   The SSCP sends ACTLU requests to these dependent logical units.

The process of activating the links to adjacent nodes, then the PUs, and finally the LUs, continues until all the network resources are active. Network deactivation follows the same hierarchy as network activation, but in reverse order.

Appendix A, "Sequence Charts" contains sequence charts that illustrate the various command flows for network activation and deactivation. Detailed discussions of the commands and their sequences are provided in *SNA Format and Protocol Reference Manual: Architectural Logic*, *SNA Format and Protocol Reference Manual: Architecture Logic for LU Type 6.2*, *SNA LU 6.2 Reference—Peer Protocols*, *SNA Type 2.1 Node Reference*, and *SNA Formats*.

SSCP Takeover in Subarea Networks

SSCP takeover occurs when an SSCP that owns resources in a subarea network fails and another SSCP assumes ownership of the resources. In Figure 58, SSCP *h* assumes ownership of those resources owned by SSCP *a* when SSCP *a* fails.

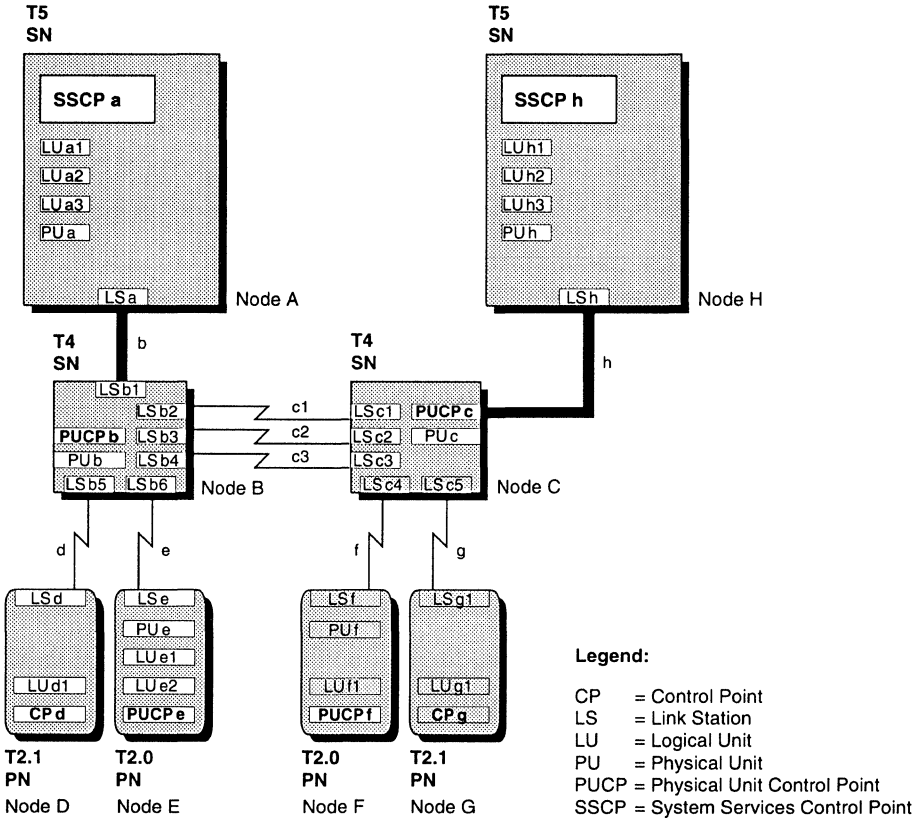


Figure 58. SSCP Takeover in a Subarea Network

1. SSCP *a* fails.
2. SSCP *h* activates PU *c* and PU *b*, respectively.
3. SSCP *h* activates LS *c4* and LS *c5*.
4. SSCP *h* activates LS *b5* and LS *b6*.

The detailed flow for the SSCP takeover in a subarea network follows that of Figure 52 through Figure 57, except that SSCP *h* initiates the activation over link h, and then the resources mentioned above are activated.

Cascaded Activation and Deactivation

The task of activating and deactivating a large network would be extremely time-consuming if a network operator had to enter separate commands for each resource, while having to remember the hierarchy of resource activation. To simplify network activation, entire subareas, or portions of subareas, can be automatically activated as the result of a single operator command. This procedure is called ***cascaded activation***.

Network personnel can specify in system definition statements that most of a network be activated as the result of one operator command and that the remaining resources be activated by separate operator commands. Parameters specified in

the definition statements vary depending on the particular software programs that are in the network. Cascaded activation can also be specified in the start parameters for a system generation.

Cascaded activation can begin or end at any point in the resource hierarchy. This allows considerable flexibility when specifying parameters in the definition statements.

The cascaded deactivation capability enables a network operator to enter a single command to deactivate multiple resources. When reactivating network resources after part of the domain has failed or has been deactivated, the operator can restore inactive resources to either (1) the status they had before the failure or deactivation, or (2) the status defined in resource-definition statements.

Activating APPN Networks

APPN networks are formed through the autonomous actions of APPN nodes in connecting with one another. In the XID3 commands exchanged during link activation, a node in an APPN network indicates whether it is a LEN end node, an APPN end node, or an APPN network node.

An end node can connect to more than one network node, but only one network node at a time can be the network node server for the end node. The network node server for a LEN end node is fixed by joint agreement of, and definition by, node operators of the two nodes. The network node server for an APPN end node is selected dynamically by the end-node session services component.

Activating CP-CP Sessions

Once two APPN nodes have connected and established mutually acceptable link parameters, the node control points may initiate CP-CP sessions between them over the newly established link. (LEN end nodes cannot support CP-CP sessions.) Two CP-CP sessions are initiated, one by each CP. Between two network nodes, it is typical for CP-CP sessions to be activated, although they do not need to be. If both network nodes are already connected to the network through other network nodes, then CP-CP sessions between them are not required to join the network. Suppressing the CP-CP sessions reduces the overhead incurred by TDU broadcasts and broadcast searches. Between a network node and an APPN end node, CP-CP sessions are activated only if the network node is to be the network node server for the end node. Between two APPN end nodes, CP-CP sessions are never activated.

Session services (SS) in APPN node CPs initiate their CP-CP sessions using LU 6.2 protocols. Under LU 6.2 protocols, a contention situation can arise if both session partners attempt to transmit on a session simultaneously. This is resolved by designating one session partner the *contention winner,* and the other the *contention loser.* Because, under LU 6.2 protocols, the partner that initiates an LU 6.2 session becomes the contention winner, each of two paired CPs initiating CP-CP sessions has one contention-winner session on which to send requests to its partner, and one contention-loser session on which to receive requests from its partner. For more information on LU 6.2 protocols, see "CPI-C and LU 6.2 Protocol Boundary" in Chapter 8, "Transporting Data Through the Network."

After CP-CP session activation, SS in each node sends a request over its contention-winner session to the SS component in the other node for information on that node's CP capabilities. Such information includes the node's management services (entry-point and focal-point) capabilities and the kinds of directory objects for which an end node may be searched by its network node server. By defining the extent of network services that each node supports, CP-capabilities information provides the basis for future CP-CP communication between the nodes. Following the CP capabilities exchange between APPN nodes, directory or network topology information may be exchanged, depending on the roles of the nodes involved.

Activating APPN End Nodes

After an APPN end node establishes CP-CP sessions and exchanges CP capabilities with its network node server, it sends information to the network node server about local resources (LUs) that it wishes to place in the distributed network directory.

Resource Registration

Resource registration places information about the location of resources in a directory services database. There are two types of resource registration:

- ***APPN end node resource registration***— An APPN end node can add a new resource to the list of resources known by its network node server, thus making these new resources known to the distributed directory of the network; likewise, an APPN end node can delete resources as needed. An APPN end node can also request that its network node server register selected resources of the end node with a ***central directory server (CDS)***.
- ***Central resource registration (CRR)***— An APPN network node can register local and same-domain resources at a CDS.

APPN End Node Resource Registration

The sending of local resource information by an APPN end node to its network node server for recording in the distributed network directory is called APPN end node resource registration. An APPN end node indicates which of those resources should also be centrally registered. An APPN end node has the option of specifying whether it is to be searched. However, if a resource is not registered and the APPN end node specifies that it should not be searched for any resource, the resource will not be found. Recall that the domain of a network node includes its local links and client end nodes, including the known resources in its client end nodes. When the resources of a network node are defined to it at initialization (system definition) time, however, only the resources local to the node or in attached LEN end nodes must be specified; resources in client APPN end nodes may be registered dynamically.

LEN nodes are unable to register resources at their network node server, since they do not support CP-CP sessions, which are a prerequisite for resource registration. Such resources must be defined at the network node server, in order to be accessible to the network.

A network node can dynamically learn of the resources in its client APPN end nodes by sending search requests to them as well as receiving search requests from them. When a network node learns of a resource in a client APPN end node, it records the resource information in its local directory database. This can relieve the network node from having to query each of its attached APPN end nodes for a

destination LU when it receives a search request for that LU from another network node. The directory database in an APPN node is, therefore, a key component of the mechanism by which APPN nodes dynamically determine the location of network resources. Rather than a number of individual local directories (one per node), the directory database should be thought of as a single distributed directory of network resources, portions of which reside in each node.

Figure 59 illustrates the registration signals between an APPN end node and its network node server.

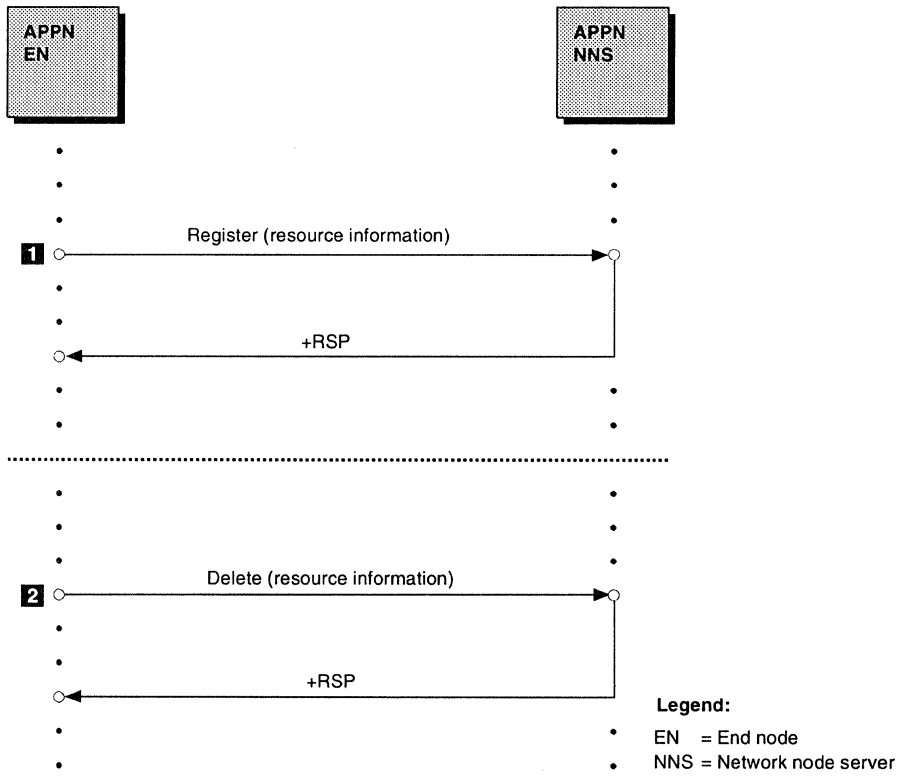


Figure 59. Resource Registration

The following comments correspond to the numbers in Figure 59:

1 An APPN end node can add to the list of resources known to its network node server by registering new resources using the Register GDS variable. This registration causes these new resources to be made available to the network.

2 As needed, the APPN end node may delete resources from the network node server's directory (using a Delete GDS variable). To change a resource entry, the APPN end node first deletes the old entry and then registers the new information.

Central Resource Registration (CRR)

CRR allows a network node to register its domain resources at a central directory server (CDS). Once the resource is registered, all network nodes in the same topology subnet can find the resource, if they do not already know it, by sending a Locate search request to a CDS. Resource registration reduces considerably the number of broadcast searches.

Instead of trying to locate an unknown resource themselves via a *broadcast search*, network nodes query their closest CDS. The CDS takes responsibility for locating the resource by querying its own directory, by querying other CDSs in the same topology subnet, or by initiating a network-wide broadcast search, in that order. The CDS concept maximizes the sharing of cached directory entries, thereby minimizing the number of network broadcast searches.

The topology database is used to allow identification of CDSs and their capabilities to every network node in the same topology subnet. Central directory servers identify themselves with an indicator in the topology database update (TDU) messages when connecting to the network, thereby informing all network nodes of their presence.

When registering its resources with its network node server, an APPN end node indicates which of those should be centrally registered, as specified in its local definitions. Since it is optional for end nodes to register specific resources with their network node servers or central directory servers, any unregistered resources may still require a broadcast search to locate the resource, which prevents total elimination of broadcast searches.

Differences Between APPN End Node Resource Registration and Central Resource Registration

Several differences exist between APPN end node resource registration (APPN end node to network node server) and CRR (network node to CDS). The differences are:

- Generally, CRR does not use a direct CP-CP session between the two endpoints (that is, the network node server and the central directory server) since they are not necessarily adjacent. For this reason, CRR does not use confirmation processing as the registration response mechanism; instead, CRR uses the Locate GDS variable as the registration response mechanism.

  APPN end node resource registration uses a direct CP-CP session between the APPN end node and its network node server to register resources.

- CRR does not use the Delete GDS variable. Updates to resource information at the CDS are made by submitting a subsequent registration request that overlays existing information. Deletion of information happens as a consequence of cache limit being exceeded. If a network node submits a search request to a CDS that has outdated information, a search to the expected destination by the CDS will fail, resulting either in that CDS querying other CDSs for the resource or a full broadcast search by the network node and new information being cached at the CDS.

  APPN end node resource registration uses the Delete GDS variable followed by the Register GDS variable to update resource information at the network node server.

- A given network node can have only one outstanding CRR request at a time. This way, a CDS is not flooded with registration requests.

  APPN end node resource registration does not limit the number of registration requests that can be outstanding.

Central Directory Server (CDS)

A CDS is a network node that includes capability to:

- Accept registration of resources by other network nodes for inclusion in its directory, called a ***central directory***. These resources include those registered by APPN end nodes with their network node server and designated for central registration, as well as resources in the network nodes themselves.
- Accept directed Locate searches (called *referred* searches) from other network nodes unable to find resources via directed searches to their actual destinations. A central directory server first checks its own directory. If it has no matching entry, it then sends directed Locate searches to any other CDSs. If none of these is successful, it then performs a general broadcast to find the resource. This function can greatly reduce network search traffic.

Figure 60 illustrates how the Locate is used to enable central resource registration to a CDS.

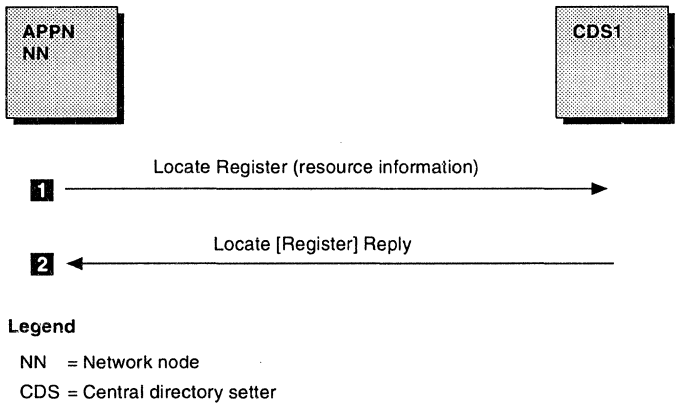


Figure 60. Resource Registration to a Central Directory Server

The following comments correspond to the numbers in Figure 60.

1 A network node sends a Register request to its closest CDS (CDS1). Locate with an RSCV is used to route the resource registration to the CDS. The Register GDS variable causes the new resources specified in the Register request to be recorded in the CDS directory database as cached entries.

2 The CDS responds to the Locate Register request by returning a Locate reply.

For performance and reliability reasons, more than one CDS can be present. The presence of a CDS is known only to the topology subnet in which the CDS resides. One subnet cannot request that a search be forwarded to a CDS in another subnet.

For further information on CDS, see "Central Directory Server (CDS)" in Chapter 1, "Introduction" and "Use of a Central Directory Server in a Search" in Chapter 6, "Establishing Routes Through the Network."

Kinds of Directory Database Entries

There are three kinds of entries in an APPN or LEN node's directory database: local entries, domain entries, and other-domain entries. As discussed previously, ***local directory entries*** define the resources local to (under the direct control of) a node. Both end nodes and network nodes have local entries in their directories.

Local entries are created by operator definition and are stored as ***home*** directory entries. A local entry includes the resource's name and type.[6]

Domain directory entries define the resources within a node's domain. Because a LEN or APPN end node's domain includes only its local resources, domain entries for an end node are synonymous with local entries. For a network node, however, domain entries include both its local resources and the resources that it knows of in its client end nodes. A network node's domain entries are created by resource registration (in which case they are ***registered*** directory entries), by node-operator definition (in which case they are either home or ***cache*** entries), or as a result of search requests received by the network node either from a client end node or from other domains. When, following a domain search, a network node learns of a resource residing in an attached end node, it stores the resource location information as a cache directory entry in its directory database. It also caches information on previously unknown client end-node resources (origin LUs) on whose behalf it performs successful searches for destination resources. A cache directory entry is a directory entry that is subject to overlay when directory storage reaches its capacity, while home and registered entries are not. A home entry in a network node cannot be changed by a Register or Delete from a client APPN end node.

Network nodes are capable of supporting end nodes that cannot register their resources, for example, LEN end nodes, or APPN end nodes that are not authorized by the network node to do so. The resources for these nodes must therefore be registered by operator definition. A domain directory entry in a network node for a resource in a client end node includes the resource's name, its type, and the name of its owning control point.

A node's ***other-domain directory entries*** define the resources within the domains of other nodes in the network. An end node requires other-domain entries for resources that it cannot locate through the services of its network node server. A LEN end node requires other-domain entries for all resources outside of its own domain that it may wish to contact. An APPN end node requires other-domain entries for all resources that its network node server does not know of from operator definition or resource registration, and cannot learn of through search protocols. Such resources might reside in adjacent LEN nodes or subarea networks. For LEN and APPN end nodes, other-domain entries are defined by their local operators. Network nodes may also create other-domain entries as cache directory entries after locating them through search protocols. An other-domain directory entry includes the resource's name, its type, the name of its owning control point, and the name of the control point in its network node server.

Some products can support ***safe store*** of the directory cache. The cache entries in a network node's directory database are periodically written to a permanent storage medium, permitting faster recovery after a network node failure or initial power-on.

For information on locating resources in an APPN network, see "Locating the Destination LU" in Chapter 6, "Establishing Routes Through the Network."

---

[6] The only type of resource currently defined is the LU.

Activating Network Nodes

After two adjacent network nodes interconnect, they normally exchange network topology information over their CP-CP sessions. The rules governing the exchange of topology information are described in the next section.

Topology Database

The primary use of local and network topology databases is for route calculation. When an LU residing in one APPN node wishes to establish a session with an LU residing in another, topology databases enable topology and routing services in the network node (NN) server for the *origin* LU to determine the best possible route to the *destination* LU.

There are two kinds of topology databases in an APPN network: the local topology database and the network topology database.

Local Topology Database

The ***local topology database*** contains information on all of the transmission groups (TGs) attached to the node. Every APPN node maintains a local topology database. An APPN end node uses the information in its local topology database to send local TG information to its network node server in Locate requests and replies. In a network node, the local topology database includes information about the attached end nodes. In HPR networks, the APPN end node's TG information (carried in TG vectors) indicates whether the nodes are HPR-capable.

End Node Topology Database Manager: The topology database manager (TDM) creates and maintains the topology database. Entries in the topology database are created automatically when configuration services informs TDM about newly activated or changed TGs. The operator updates the topology database through configuration services. The topology database is searched by TDM when it receives a query from route selection services or from session services.

Network Topology Database

The ***network topology database*** contains information on all network nodes in the APPN network as well as the transmission groups interconnecting them. Every APPN network node maintains a network topology database in addition to its local topology database. The network topology database does not include information on LUs, APPN end nodes, LEN end nodes, or the transmission groups attached to them. (Although the TGs connecting a network node to attached end nodes are not included in the network topology database, the network node, nevertheless, knows of those TGs through its local topology database.) Network nodes in an APPN network send one another topology database updates (TDUs) over CP-CP sessions whenever a resource (network node or a TG between network nodes) is activated or deactivated, or its characteristics change. Only the current changes are included in the TDU. Every network node receives the TDU containing the current change, so each has the same view of the network. The network topology database is used by the network node to select routes for sessions that originate at the LUs in it and at the end nodes that it serves. Unlike the directory database, which is *distributed* among network nodes, the network topology database is fully *replicated* at each network node within the topology subnet. APPN protocols for the distribution of network topology information ensure that every network node is provided with a complete view of its subnet topology.

The topology formats and protocols are basically the same for HPR as for APPN, except that an HPR transmission group is indicated as such in a TDU. Nodes in the HPR subnet appear as APPN nodes to the nodes in the APPN subnet. Nodes in the HPR subnet can distinguish between the APPN and HPR TGs and nodes.

The primary use of local and network topology databases is for route calculation. When an LU residing in one APPN node wishes to establish a session with an LU residing in another, topology databases enable topology and routing services in the network node (NN) server for the *origin* LU to determine the best possible route to the *destination* LU. For information on establishing routes in APPN networks, see Chapter 6, "Establishing Routes Through the Network."

Network Node Topology Database Manager

The network node topology database manager (network node TDM) is a component that resides in every network node and is responsible for maintaining the local copy of the network topology database.

Topology Database Updates (TDUs): Topology information is sent in ***topology database update (TDU)*** messages. Each network node TDM creates and broadcasts topology database updates (TDUs) about the node itself and locally-attached intermediate routing TGs to adjacent network nodes using its CP-CP sessions with adjacent network nodes. A network node TDM stores information obtained from TDUs received from adjacent network nodes in its copy of the network topology database, and forwards the TDU to adjacent network nodes. This allows every network node TDM in a topology subnet to maintain a consistent copy of the topology database.

A timer interval is assigned to every resource entry in the local copy of the network topology database, to allow the resource to be discarded if no information about the resource has been received for 15 days. This process is known as ***garbage collection***. Every network node TDM broadcasts TDUs containing local information every five days, to prevent other network nodes from discarding valid information.

Upon receiving a TDU message, a network node updates all new resource information (information on added or changed resources) in its network topology database. Each network node is also responsible for retransmitting any new network topology information to all its adjacent network nodes. Retransmission of TDU messages to adjacent network nodes is called ***TDU broadcast***. This process ensures that a common network topology database is distributed to all network nodes throughout a topology subnet.

Resource Sequence Number (RSN): An ***RSN*** is associated with the current information about each node and TG in the network topology database; this RSN is assigned by the network node that owns the particular resource. A network node owns the node definitions for itself, and the TG characteristics in the direction of an adjacent network node.

Whenever a network node detects a change in the state of a locally owned resource, it increments the RSN to the next even value. It then creates a TDU including the new RSN and broadcasts it to all its adjacent network nodes.

The use of RSNs in TDUs and the network topology database allows a network node to determine whether resource information has been received before. A TDU is discarded and not rebroadcast if the RSN in the TDU is equal to the RSN in the

topology database and the information in the TDU is the same as in its database, preventing endless retransmission of resource information.

Flow-Reduction Sequence Number (FRSN): Every network node in the network maintains a current ***FRSN*** that it increments each time it sends a TDU. Also, every network node keeps track of the last-received and last-sent FRSN for each adjacent network node. The current FRSN is included in each TDU and stored with every topology database entry included in a TDU. FRSNs are associated with TDUs, unlike RSNs, which are associated with resources.

Each network node keeps the FRSN to prevent unnecessary transmission of TDUs. When two network nodes reconnect and establish CP-CP sessions, FRSNs are exchanged to determine what information has not yet been received by the other end. For example, if NodeB reconnects with NodeA, NodeA will compare its last-sent FRSN to the last-received FRSN in NodeB. (NodeB's FRSN was originally received from NodeA.) If NodeA's FRSN for its last-sent TDU to NodeB is higher than NodeB's FRSN for its last-received TDU from Node A, then NodeA will send NodeB all of the information, via TDUs, that NodeB is missing. One or more TDUs will be built and sent including all entries of the network topology database that, according to their stored FRSNs, have not been previously sent to the respective node. This ensures that only information that has changed will be sent during the time the two nodes were disconnected.

Initial Topology Exchange

After a contention-winner CP-CP session is activated with another node, the topology database manager sends topology database updates (TDUs) that contain the node's topology database to the other node. This is called ***initial topology exchange***. A TDU message contains one or more ***resource updates***, each pertaining to a unique node resource. Using the information contained in the TDU messages, network nodes update the network topology database. In order to reduce the number of TDUs sent in the initial topology exchange, the node includes only those topology database entries that the other network node has not previously received. This optimization is called ***flow reduction***, which is achieved through use of a flow-reduction sequence number, described previously.

Products can support ***safe store*** of the topology database. A network node's topology database and necessary related information are periodically written to a permanent storage medium, thereby extending the effectiveness of initial topology exchange and garbage collection. This permits faster network recovery after node failure or initial power-on.

Sequence of APPN Network Activation

This section illustrates the sequence of events that takes place when APPN nodes connect to one another. An APPN end node connecting to a network node is shown first, followed by the same network node connecting to an APPN network through two other network nodes.

1. End node A connects to network node B.

   a. CP *a* and CP *b* activate the link connecting nodes A and B, as shown in Figure 61, and activate the CP-CP sessions between them.

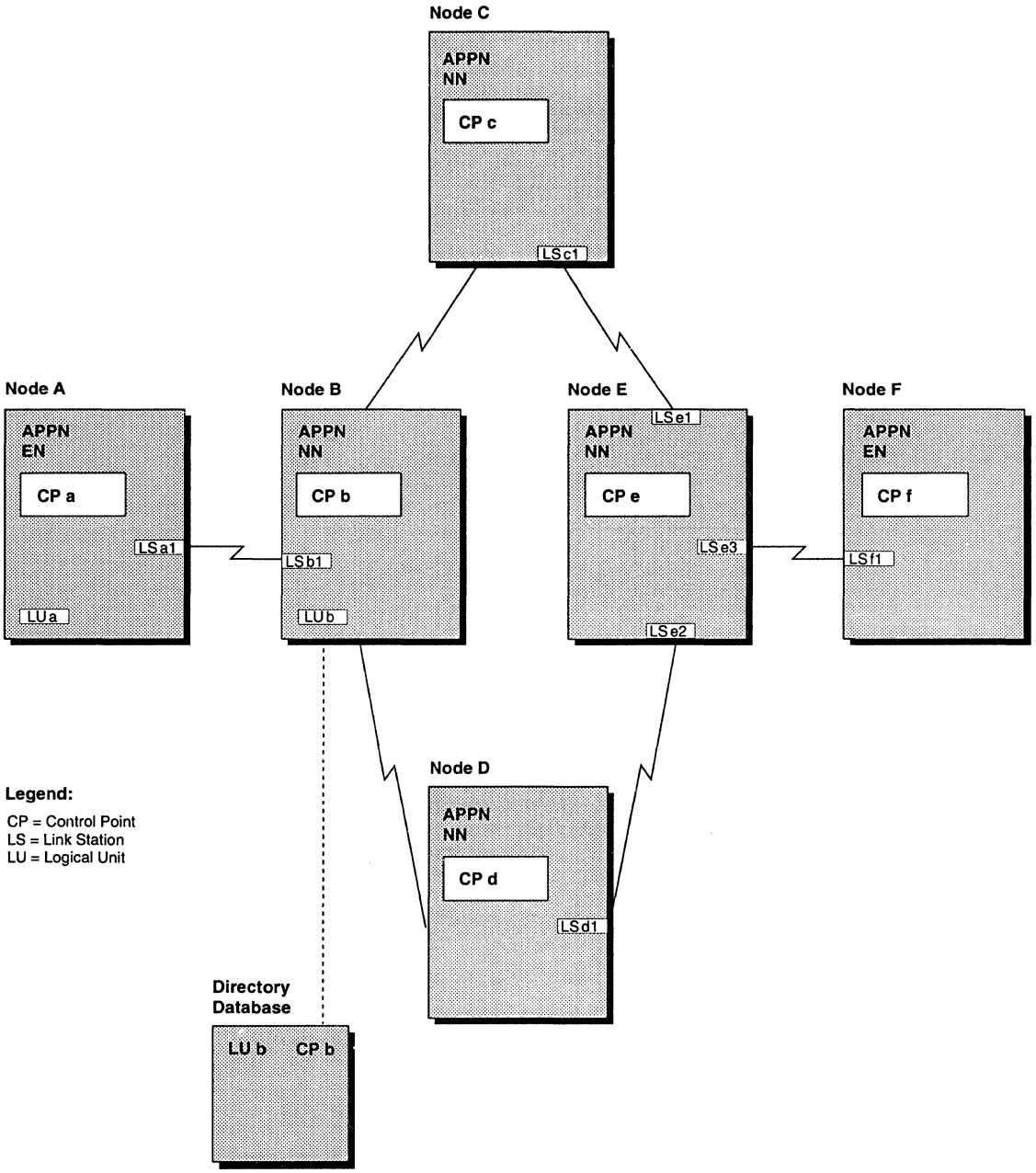


Figure 61. Sequence of APPN Network Activation: Part I

1) CP *a* activates LS *a1*.[7]

2) CP *a* issues a CONNECT_OUT command that requests LS *a1* to connect with LS *b1*.

---

7 CP *b* and LS *b1* perform a comparable sequence.

3) LS *a1* informs CP *a* that it has successfully connected with LS *b1*.

4) CP *a* and CP *b* exchange identifications and the data link protocol commands necessary to activate the link.

5) CP *a* and CP *b* activate two CP-CP sessions between them.

6) CP *a* and CP *b* exchange information on their respective capabilities.

b. CP *a* registers the resources in its domain with CP *b* as shown in Figure 62.

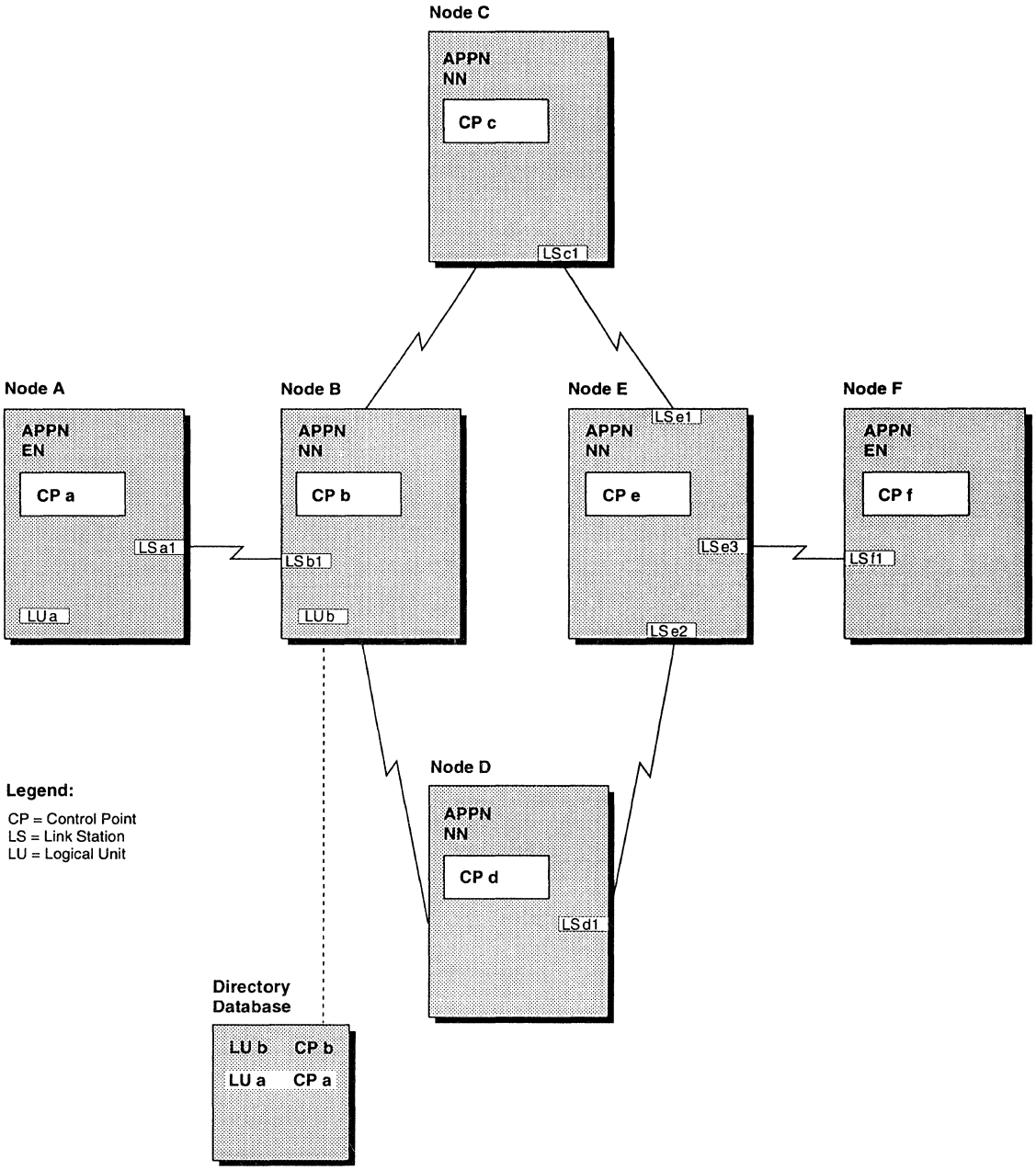


Figure 62. Sequence of APPN Network Activation: Part II

1) CP *a* sends CP *b* a list of its resources (in this case, just LU *a*).

2) CP *b* updates its local directory and returns a response to CP *a*.

2. Network node B connects to network node C.

   a. CP *b* and CP *c* activate the link connecting nodes B and C, as shown in Figure 63, and activate two CP-CP sessions between them.

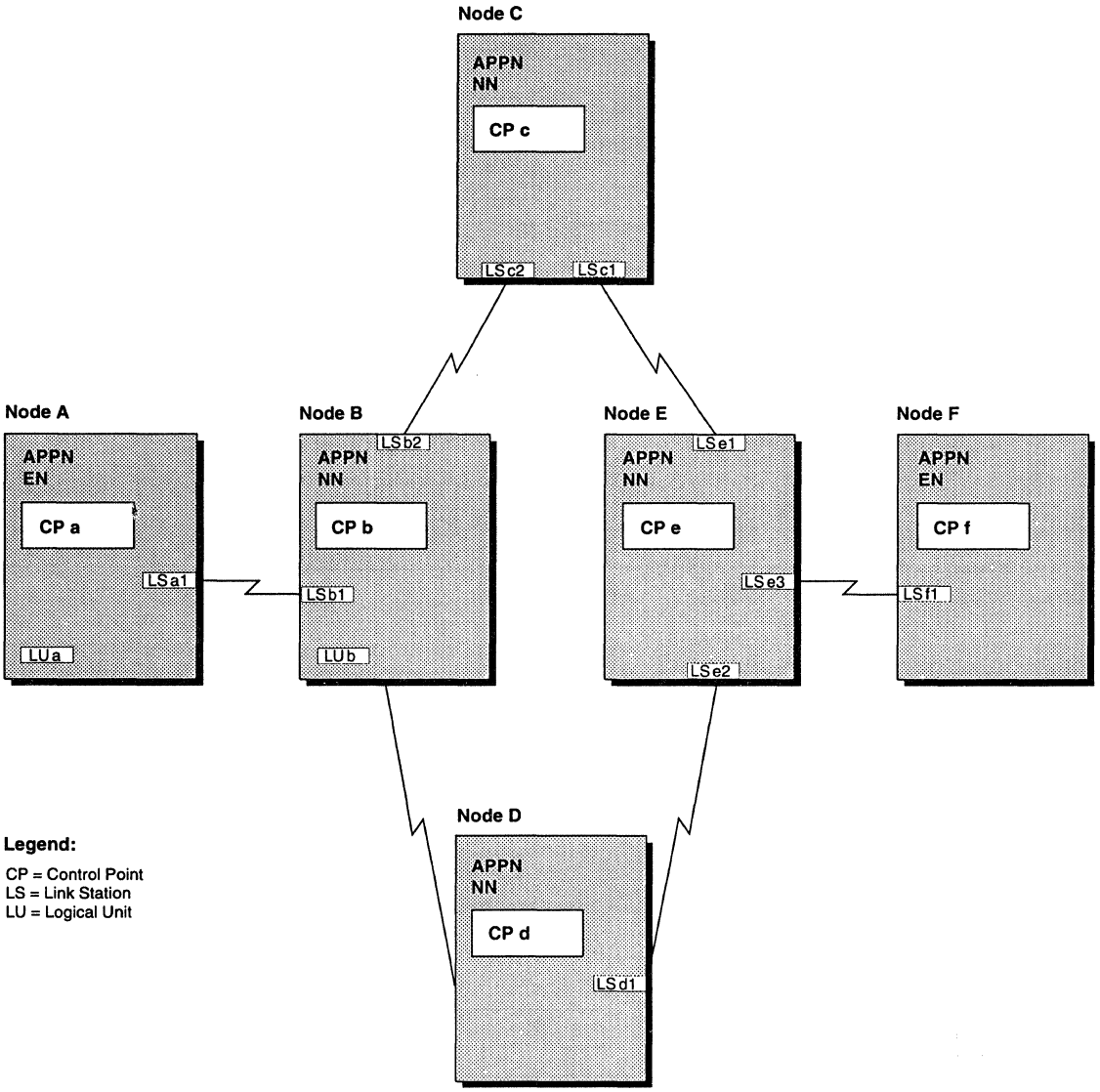


Figure 63. Sequence of APPN Network Activation: Part III

1) CP *b* activates LS *b2*.[8]
2) CP *b* issues a CONNECT_OUT command that requests LS *b2* to connect with LS *c2*.
3) LS *b2* informs CP *b* that it has successfully connected with LS *c2*.
4) CP *b* and CP *c* exchange identifications and the data link protocol commands necessary to activate the link.
5) CP *b* activates a session with CP *c*, and CP *c* activates one with CP *b*.

[8] CP *c* and LS *c2* perform a comparable sequence.

6) CP *b* and CP *c* exchange information on their respective capabilities.

b. All network node CPs update their network topology databases, as shown in Figure 64 on page 121, to reflect the new network topology.

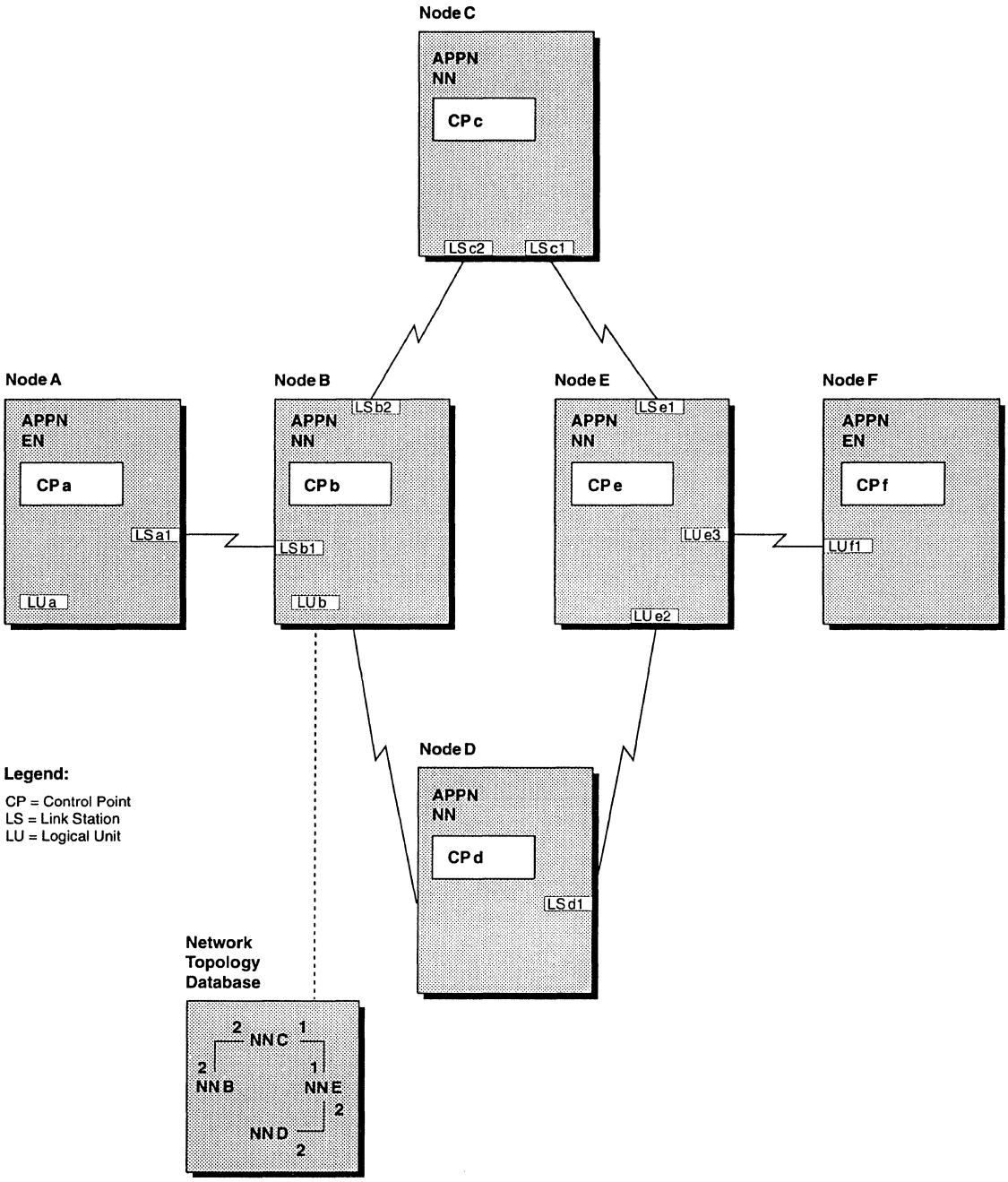


Figure 64. Sequence of APPN Network Activation: Part IV

1) CP *b* begins the initial topology exchange by sending TDU messages to CP *c* regarding the active status of NN B and the TG represented by LS *b2*.

2) CP *c* completes the initial topology exchange by sending TDU messages to CP *b* regarding the active status of NNs C, D, and E, and the TG numbers each network node uses to represent the active TGs to its

adjacent network nodes. Every TG is represented by a TG number that is negotiated by the two nodes at opposite ends of the TG.

3) CP *b* updates its network topology database to reflect the current status of the network.

4) CP *c* updates its network topology database as well, and broadcasts TDU messages to CP *e* regarding the active status of NN B and the TG information NN B and NN C maintain on their connection.

5) CP *e* updates its network topology database to reflect the current status of the network, and forwards the TDU messages to CP *d* that it received from CP *c*.

6) CP *d* updates its network topology database.

3. Network node B connects to network node D.

   a. CP *b* and CP *d* activate the link connecting nodes B and D, as shown in Figure 65, and activate two CP-CP sessions between them.

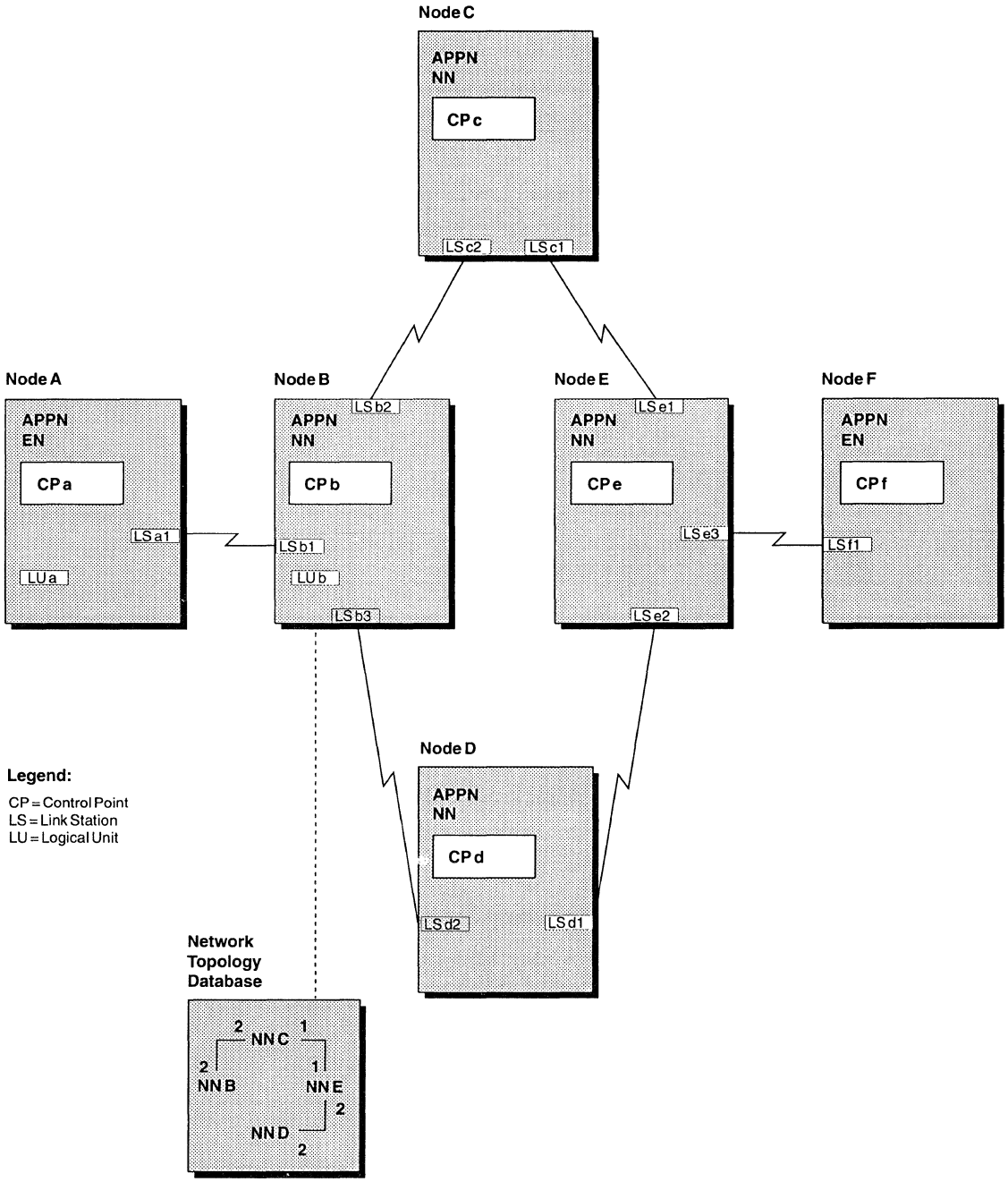


Figure 65. Sequence of APPN Network Activation: Part V

1) CP *b* activates LS *b3*.[9]

2) CP *b* issues a CONNECT_OUT command that requests LS *b3* to connect with LS *d2*.

---

9 CP *d* and LS *d3* perform a comparable sequence.

3) LS *b3* informs CP *b* that it has successfully connected with LS *d2*.

4) CP *b* and CP *d* exchange identifications and the data link protocol commands necessary to activate the link.

5) CP *b* activates a session with CP *d*, and CP *d* activates one with CP *b*.

6) CP *b* and CP *d* exchange information on their respective capabilities.

b. All network node CPs update their network topology databases as shown in Figure 66.

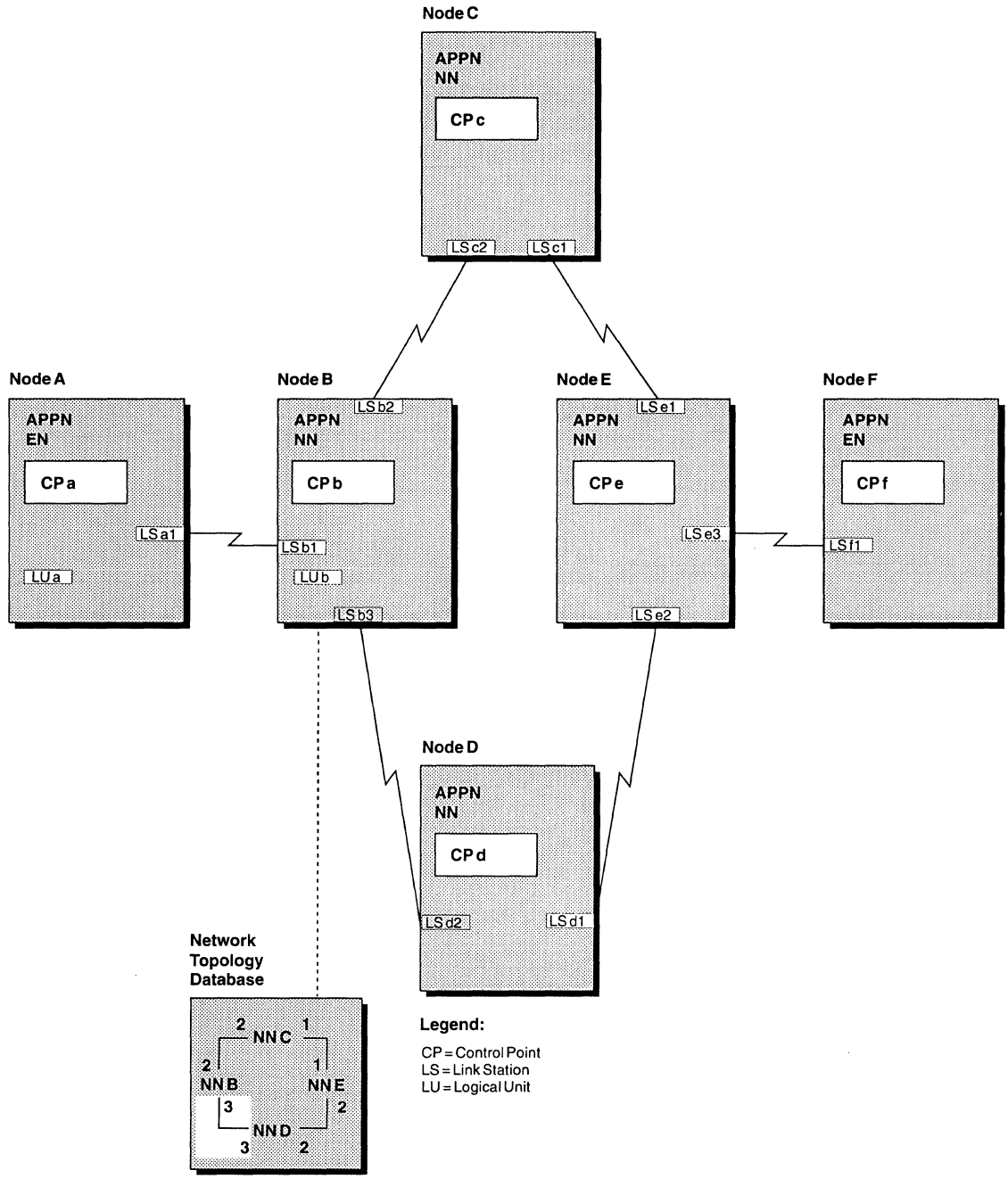


Figure 66. Sequence of APPN Network Activation: Part VI

1) CP *b* begins the initial topology exchange by sending TDU messages to CP *d* regarding the active status of NNs B, C, D, and E, and information on their corresponding TGs, including NN B's representation of the TG to NN D.

2) CP *d* completes the initial topology exchange by sending TDU messages to CP *b* regarding the active status of NNs B, C, D, and E, and information on their corresponding TGs, including NN D's representation of the TG to NN B. It then updates its network topology data-

base, using the TDU messages received from CP *b*, and broadcasts TDU messages to CP *e* regarding the active status of its TG to NN B.

3) CP *b* updates its network topology database as well and broadcasts TDU messages to CP *c* regarding the active status of its TG to NN D.

4) CPs *c* and *e* update their network topology databases and broadcast the TDU messages to each other. Each recognizes the messages as duplicates of those already received, and does not forward them.

SSCP Takeover and APPN Connections

A connection that is established as an APPN or LEN connection remains such until it is deactivated. However, a system services control point (SSCP) taking over an APPN connection views it as a LEN connection when the takeover SSCP has not implemented APPN. CP-CP sessions cannot be established over the connection until the original SSCP regains control.

In Figure 67, if both the adjacent CP and HOSTB support CP name change, CP-CP sessions can be established at the time that HOSTA fails. Nonactivation XID exchange is used to determine the CP names that are being switched. TDUs are sent throughout the network to notify it of the new topology (that is, that HOSTB has gained control).

In Figure 67, if HOSTA fails and HOSTB takes over, the CP name (from the adjacent CP's perspective) changes from HOSTA to HOSTB. If the adjacent CP does not support the changing of CP names, HOSTB treats the connection to the adjacent CP as a LEN connection (even if HOSTB has implemented APPN) and CP-CP sessions cannot be established again over the connection until HOSTA regains control, or until the TG is deactivated and then reactivated.

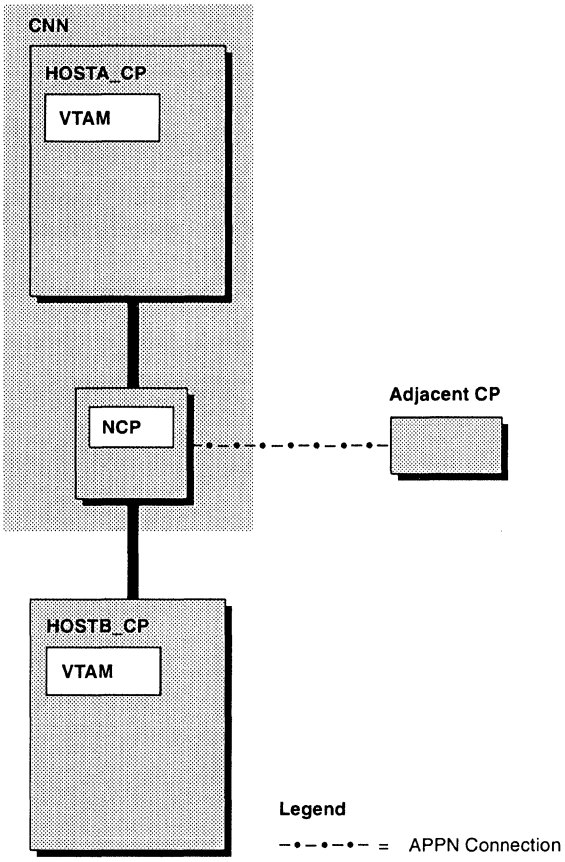


Figure 67. SSCP Takeover in a Combined APPN and Subarea Network

Chapter 6. Establishing Routes Through the Network

This chapter explains how the control points select routes for LU-LU sessions through the network in accordance with a specified class of service.

Requirements for Routing in SNA Networks

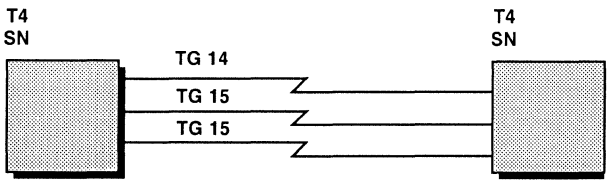
Routing is the forwarding of a message unit along a particular path through a network as determined by parameters carried in the message unit. The routing function within APPN networks is different from that within subarea networks. In both types of networks, however, certain preliminary activities must take place before nodes can perform routing. These activities can be placed into the following categories:

- Defining transmission groups
- Defining routes
- Defining class of service
- Selecting routes.

Defining Transmission Groups

A ***transmission group (TG)*** is a link or group of parallel links connecting adjacent nodes that is viewed by path control as a composite unit for routing purposes. Adjacent type 4 nodes define multilink transmission groups. Adjacent type 5 nodes can support multipath channel, which is a type of multilink TG. All other nodes define only single-link transmission groups.

In subarea networks, two or more parallel SDLC links connecting adjacent T4 nodes can be assigned to the same or different transmission groups. Each transmission group is identified by assigning the same number (called a ***transmission group number***) to each link in the group. Links can be assigned to transmission group numbers 1 through 255. (In actual implementations, System/370 data channels are always assigned to transmission group number 1.) For example, three parallel links can be grouped into one, two, or three transmission groups. Figure 68 shows three parallel links grouped into two transmission groups: transmission group 14 and transmission group 15.



Legend:

TG = Transmission Group

Figure 68. Transmission Groups

A transmission group that consists of parallel links is more likely to be available than a transmission group that consists of a single link. If one of the links fails, data traffic continues to flow on the remaining links in the group. This enables the network to keep active all sessions that are traversing the TG, despite the failed link.

Only links that have similar transmission characteristics are normally placed in the same transmission group. For example, links that have a high transmission speed would be placed in one transmission group while links that have a medium transmission speed would be placed in another.

Path control assigns transmission group sequence numbers to path information units (PIUs) before transmitting them across a transmission group. The assignment of sequence numbers to PIUs is called ***PIU sequencing***. Because data link control can route related path information units (PIUs) over different links in a transmission group, path control in another node might not receive the PIUs in the order in which they were sent. With the PIUs sequenced, path control on the other side of the transmission group can use the sequence numbers to reorder any out-of-sequence PIUs before continuing to route the data through the network. This ensures that the arrival order at the destination session endpoint matches the sending order at the origin endpoint.

APPN Transmission Groups

An ***APPN transmission group (APPN TG)*** is the physical connection between two nodes that support APPN or LEN protocols. Transmission groups are described with common APPN characteristics:

- Cost-per-connect time
- Cost-per-byte
- Security
- Propagation delay
- Effective capacity
- User-defined values.

The values specified for these characteristics are saved in the topology database and are used to determine the weight of the TG for route calculation. If the physical characteristics of the link change, the values specified for the TG characteristics should be changed to reflect the new physical characteristics.

APPN TGs (using FID2 protocols) are defined between adjacent APPN and/or LEN nodes, but FID4 connections between T5 and T4 nodes are not identified in an APPN topology database.

Parallel Transmission Groups

An APPN node can have multiple links to another APPN node. ***Parallel TGs*** have different TG numbers and can have different TG characteristics. Multiple connections allow an LU in the APPN network to establish sessions through or into the subarea network by using several APPN TGs.

In VTAM, an LU can have multiple sessions with one or more LUs through multiple APPN TGs and through different physical paths that function as parallel TGs. For example, in Figure 69, the TG between NNa and VTAM1 is parallel to the TG between NNa and NCP1. They are parallel because both TGs go to the composite network node containing VTAM1 and NCP1.

APPN Connections through Virtual-Route-Based Transmission Groups

A ***virtual-route-based transmission group (VR-based TG)*** is the logical connection between the domains of two T5 (VTAM) APPN nodes in which the underlying physical connection consists of subarea virtual routes. This capability enables data using APPN protocols to be routed through the subarea network. The VR-based TG function allows greater network flexibility, since it supports APPN flows across existing virtual routes. Existing subarea functions remain supported across the virtual route; the VR-based TG function enables the APPN networks attached to each T5 node to be connected through a subarea network. They may

be disjoint portions of the same net-ID subnet, in which case they may comprise one topology subnet. A benefit of using a VR-based TG is that dynamic route selection is used with the performance of a FID4 connection.

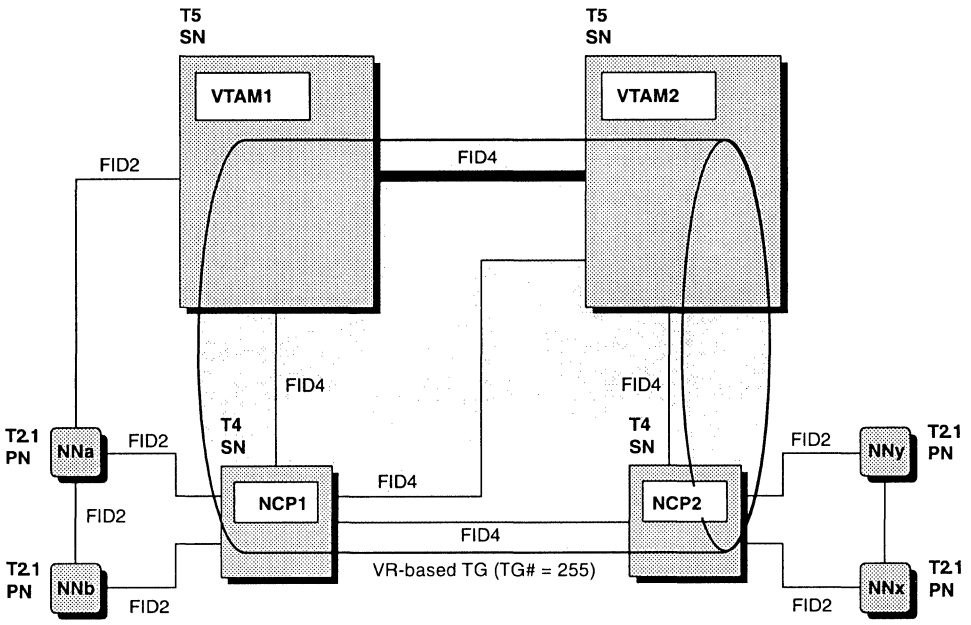


Figure 69. Virtual-Route-Based TG in an APPN-Subarea-APPN Configuration

Figure 69 illustrates an example of a VR-based TG being defined between two VTAM nodes. CP-CP sessions can now be established between the two T5 nodes and, therefore, a common APPN network topology database is available to all nodes in the diagram. APPN route selection can be performed throughout the network. The VR-based TG represents all possible VRs between any subarea in VTAM1's domain to any subarea in VTAM2's domain. For VR-based TGs, a reserved TG number of 255 is always used. The data will still be sent through the subarea network using FID4 protocols. For example:

- A session between NNb and VTAM2 uses a VR between NCP1 and VTAM2.
- A session between NNx and VTAM1 uses a VR between VTAM1 and NCP2.
- A session between NNa and NNy uses a VR between NCP1 and NCP2.
- A session between VTAM1 and VTAM2 uses a VR between VTAM1 and VTAM2.

Defining Routes in Subarea Networks

Route definition is an activity specific to subarea networks. Because the nodes comprising a subarea network are predefined, the routes between the nodes can be predefined as well. The routes are represented in tables that are distributed among the nodes. The topology of an APPN network, however, is formed dynamically as nodes join the network. Routes in APPN networks are therefore not predefined; they are established dynamically at the time of LU-LU session initiation.

The task of designing routes through a subarea network ranges in difficulty from simple to complex. The degree of complexity depends on the number of subareas,

connections between subareas, and types of sessions that a network supports. Routes through a network are designed with some combination of the following objectives in mind:

- Maximize quantity of data transmitted
- Maximize data security
- Maximize route availability
- Minimize transmission time
- Minimize cost
- Minimize the necessity to retransmit data
- Minimize traffic congestion.

Route design involves design objective trade-offs; one objective is often achieved at the expense of another.

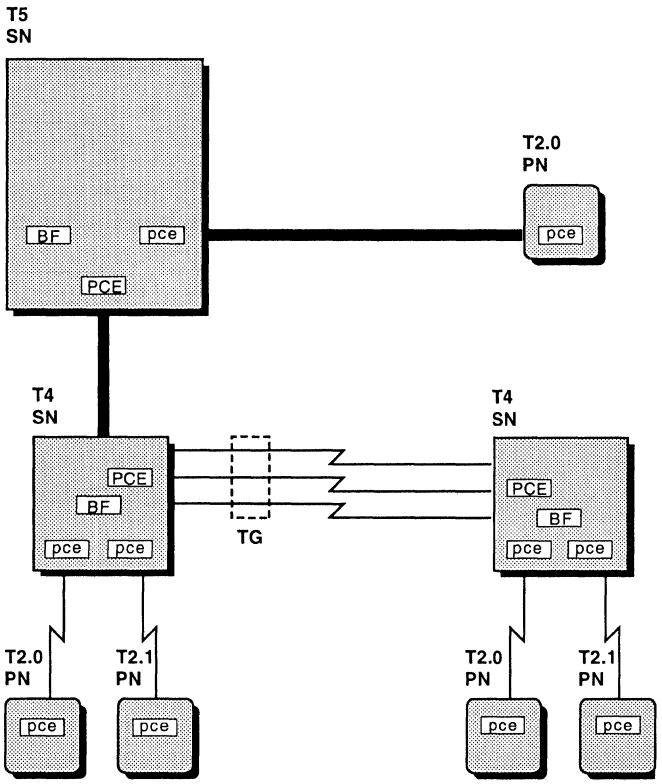
The process of designing routes in a subarea network involves defining one or more paths between each pair of subarea nodes that need to communicate with one another. A ***path*** consists of a series of path control elements, data link control elements (the link stations), and link connections. In a subarea network, there are two kinds of path control elements: subarea path control elements and peripheral path control elements.

Subarea and Peripheral Path Control

Subarea path control elements route data between subareas. Subarea path control elements use network addresses to route message units through a network. ***Peripheral path control*** elements route data between subarea nodes and peripheral nodes. Peripheral path control elements use local addresses to route message units. Subarea boundary function components interconnect subarea and peripheral path control elements.

The Boundary Function Component

The subarea ***boundary function*** component pairs the network addresses that subarea path control elements use with the local addresses that peripheral path control elements use. The boundary function thereby insulates peripheral nodes from network address changes that result from network reconfigurations. As shown in Figure 70, every subarea node that has a peripheral node attached to it contains a boundary function component.



Legend:

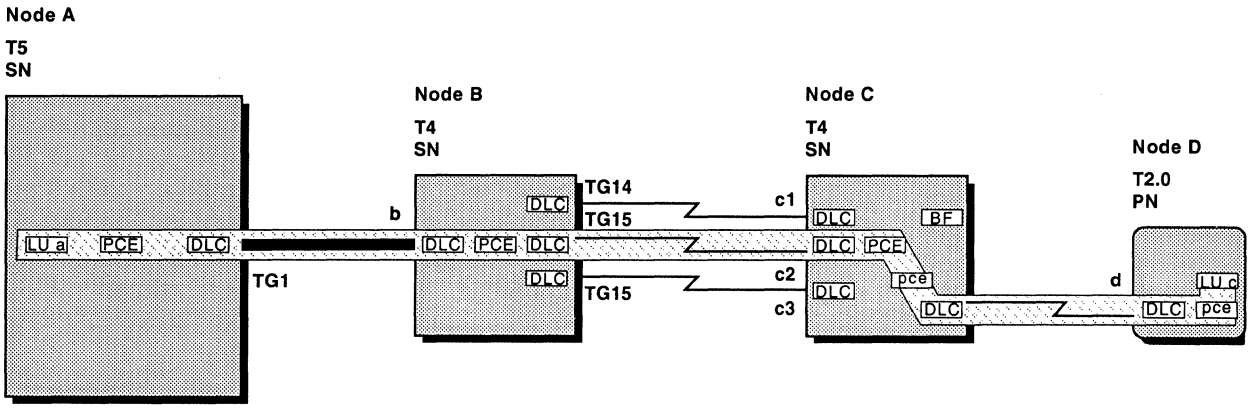
BF = Boundary Function
pce = Peripheral Path Control Element
PCE = Subarea Path Control Element
TG = Transmission Group

Figure 70. Boundary Function Components

Although the boundary function interconnects subarea and peripheral path control elements, it is not considered part of a path. For example, consider the path of a message unit that is sent from an LU in a T5 node to an LU in a peripheral node. Figure 71 shows a path between the two LUs.

The path shown:

- Proceeds from LU *a* in node A
- Over link *b* (transmission group 1)
- Through node B
- Over link *c2* or *c3* (transmission group 15)
- Through node C
- Over link *d*
- To LU *d* in node D.

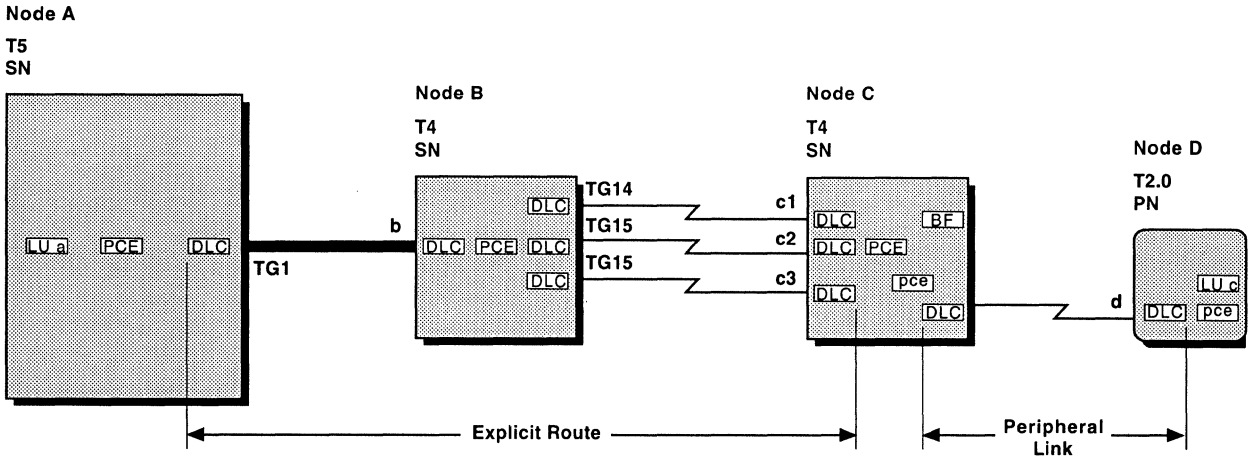


Legend:

BF = Boundary Function
DLC = Data Link Control
LU = Logical Unit
pce = Peripheral Path Control Element
PCE = Subarea Path Control Element
TG = Transmission Group

Figure 71. A Path between Logical Units

Defining a path in a subarea network requires an explicit route and, if required, a peripheral link to be specified. The ***explicit route*** is the portion of the path between the two endpoint subarea nodes, and the ***peripheral link*** is the portion of the path between a subarea node and a peripheral node. Figure 72 illustrates an explicit route and a peripheral link for the path between LU *a* and LU *d*.



Legend:

BF = Boundary Function
DLC = Data Link Control
LU = Logical Unit
pce = Peripheral Path Control Element
PCE = Subarea Path Control Element
TG = Transmission Group

Figure 72. An Explicit Route and a Peripheral Link

Defining Explicit Routes

SNA defines an ***explicit route*** as an ordered set of subarea nodes and transmission groups along a path. Up to 16 explicit routes can be defined between any two subarea nodes; an explicit route number is assigned to each of these routes.

Different explicit routes can include the same subarea nodes or transmission groups. For example, consider the path in Figure 73. Two explicit routes can be defined between subarea nodes A and C.

Explicit route 0 could be defined as:

- Subarea node A
- Transmission group 1
- Subarea node B
- Transmission group 14
- Subarea node C.

Explicit route 1 could be defined as:

- Subarea node A
- Transmission group 1
- Subarea node B
- Transmission group 15
- Subarea node C.

Explicit routes 0 and 1 include the same subarea nodes. Both explicit routes also include transmission group 1 between subarea nodes A and B.

The only difference between these two explicit routes is the transmission group between subarea nodes B and C. Just as transmission groups can be defined to have multiple parallel links, more than one explicit route can be defined between subarea nodes in order to increase the probability that a path will be available between the two nodes.

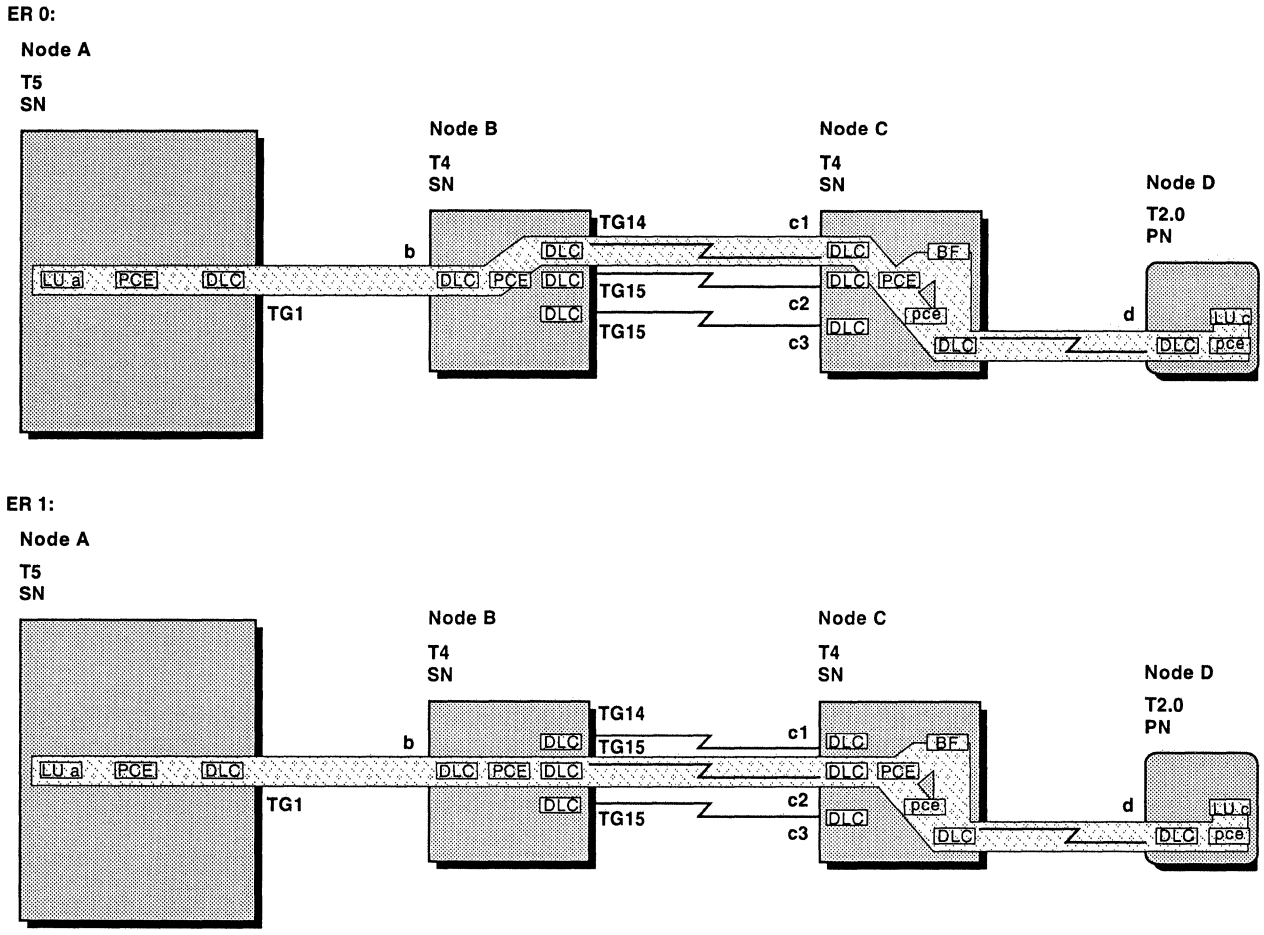


Figure 73. Multiple Explicit Routes

Explicit routes are bidirectional. Numbers 0 through 15 could be assigned to explicit routes in the forward direction, and the same numbers could be assigned to explicit routes in the reverse direction. Explicit routes in the forward and reverse directions must use the same set of subarea nodes and transmission groups. However the same explicit route number does not have to be assigned to both the forward and reverse directions.

Defining Virtual Routes

Whereas an explicit route is a physical connection between two subarea nodes, a ***virtual route*** is a logical connection between two subarea nodes. Each virtual route is mapped to an explicit route during system definition. One or more virtual routes can be defined to a single explicit route. A virtual route takes on the physical characteristics (bandwidth and transmission rates) of the explicit route to which it is assigned.

A virtual route is represented by a ***virtual-route-number*** and a transmission priority. Up to eight virtual route numbers can be assigned, each with three different transmission priorities, between two subarea nodes. The three transmission priorities, when combined with the eight possible virtual route numbers, allow up to 24 virtual routes to be defined between two subareas.

A ***transmission priority*** identifies the priority of LU-LU session data flowing over an explicit route. One of three levels of transmission priority for each virtual route can be specified: 0 (lowest), 1, or 2 (highest). Certain control signals, however, such as for virtual-route pacing, flow at a special network priority.

Path control queues message units before transmitting them over a transmission group. It transmits message units that have a high transmission priority ahead of those that have a medium or low transmission priority. Path control provides an aging algorithm that periodically reorders the transmission priority of message units in a queue to ensure that low-priority message units do not remain in the queue for an extended period of time.

Defining Class of Service

A ***class of service*** designates the transport network characteristics of a session. It includes such characteristics as security, transmission priority, and bandwidth. The components of a class of service differ between subarea and APPN networks. But the process of defining class of service is an activity that must take place in both types of networks before route selection can take place. During session initiation, the class of service for the session is obtained from the session-initiation request, or derived from a mode name specified in the session-initiation request. The route then selected for the session depends on the class of service for the session.

In an SNA network, different classes of service can be specified, based upon the needs of the end users in the network. End users typically require sessions with widely varying data transmission requirements. A range of classes of service can therefore be provided to accommodate their session requirements. For example, the following classes of service can exist in a network:

- A class that provides response times suitable for high-priority interactive sessions
- A class that provides response times suitable for low-priority interactive sessions
- A class that provides routes that have the best availability
- A class suitable for batch processing
- A class suitable for high-security transmissions.

The classes of service available in a network are identified with alphanumeric names. After the classes of service are named, these names are used to label entries in a class-of-service table. A ***class-of-service table*** (COS table) defines a range of acceptable characteristics and transmission priorities for each class of service in the table.

In subarea networks, the characteristics of a COS are defined *implicitly* by the virtual routes that are defined for it. The characteristics of a COS are the characteristics of the underlying explicit routes to which the virtual routes for that COS are

assigned. In APPN networks, COS characteristics are defined *explicitly* with COS table parameters.

In a typical network, requested sessions have differing data transmission requirements. Inquiry-response sessions usually require faster data transmission and more predictable response times than data-collection sessions. Because sessions for several different kinds of applications can be in progress over a given route, multiple classes of service specifying different transmission priorities should be provided for the route. Classes of service intended for sessions that require rapid response times should be assigned higher transmission priorities than classes of service for sessions for which slower data flow is acceptable.

Defining Class of Service in Subarea Networks

In a subarea network, an SSCP uses the COS table to extract a list of virtual-route/transmission-priority pairs to send to the primary NAU's path control. Path control's ***virtual route manager*** then selects a pair from among those specified in the list. For the duration of the session, all session data uses the same virtual route and is assigned the same transmission priority.

COS Table Entries in Subarea Networks

An entry in a subarea network COS table contains a class-of-service name and the list of virtual-route-number (VRN)/transmission-priority pairs that are allowable for that class of service (see Figure 74). Note that it is possible and expected for two VRN/transmission-priority pairs, each residing in a different COS table entry, to specify the same virtual route number, but different transmission priorities. This would enable two sessions, specifying the two classes of service, to use the same underlying explicit route, but with one session's data traffic having priority over the other's.

COS Name	VRN	TPRI	Row 1
	VRN	TPRI	Row 2
	VRN	TPRI	Row 3

Legend:
COS = Class of Service
TPRI = Transmission Priority
VRN = Virtual Route Number

Figure 74. An Entry in a Subarea Network COS Table

The order in which the VRN/transmission-priority pairs are listed in a COS table entry determines the order in which a route for a specified class of service is selected for use. The virtual route manager selects the first active or available virtual route it finds in the received list. Therefore, the most desirable virtual routes should be listed first in the entry and the least desirable listed last. Note that all sessions in a subarea network (LU-LU, SSCP-PU, SSCP-LU, and SSCP-SSCP) are assigned to virtual routes. At least one COS table entry must be provided for SSCP-PU, SSCP-LU, and SSCP-SSCP session traffic.

Selecting Virtual Routes

When assigning a virtual route to a class of service, care should be taken to select a virtual route whose characteristics best match the requirements of the class of service. In making the selection, the physical characteristics of the explicit route that is to be used needs to be considered. Some explicit routes, and therefore the virtual routes that use them, might be better than others.

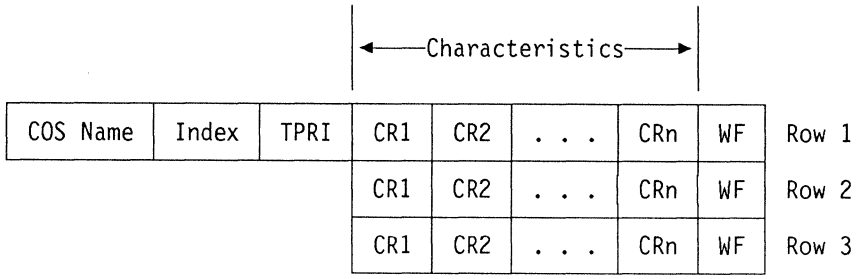
For example, virtual routes assigned to explicit routes that have fewer physical elements could be listed before virtual routes assigned to longer explicit routes. Then the longer routes would be used only for backup purposes when the shorter routes became inoperative. As another example, virtual routes assigned to explicit routes that include multiple-link transmission groups could be listed ahead of those virtual routes assigned to explicit routes that include only single-link transmission groups.

Defining Class of Service in APPN Networks

In APPN networks, the COS table is maintained by the class-of-service manager component of topology and routing services (TRS). The ***class-of-service manager*** takes direction from network operators in defining classes of service in, and deleting classes of service from, the COS table. The route selection services component of TRS can then use the COS table in selecting routes for sessions.

COS Table Entries in APPN Networks

As in subarea networks, an APPN COS table entry specifies transmission priorities. It does not, however, specify virtual routes, because virtual routes do not exist in APPN networks. An entry in an APPN COS table includes a class-of-service name, an index value, a single transmission priority, and one or multiple rows of COS characteristics that are acceptable for that class of service (see Figure 75).



Legend:
COS = Class of Service
CR = Characteristic Range
TPRI = Transmission Priority
VRN = Virtual Route Number
WF = Weight Factor

Figure 75. An Entry in an APPN Network COS Table

Unlike a subarea COS table entry, which contains multiple transmission priority fields, only a single transmission priority field is contained in an APPN COS table entry. Traffic on sessions with the same class of service in an APPN network, therefore, can flow only at the same transmission priority. Note that it is possible and expected for two COS table entries to specify the same COS characteristics, but different transmission priorities. As in a subarea network, this would enable two sessions, specifying the two classes of service, to use the same transmission facilities, but with one session's traffic having priority over the other's.

One of the fields in a row of characteristics, the ***weight field***, enables route selection services (RSS) to assign a ***weight*** to a given potential route component (node or TG). The weight is used by RSS to determine the desirability of including that component in a route. The weight field can contain a constant value, or the name of a function that RSS uses in calculating the weight for the component. Rows containing constant weights are typically ordered least-weight first. This aids the efficiency of the algorithm used by RSS in calculating routes.

The index field also aids in efficient route calculation. This field enables the computed weight values for route components to be stored and retrieved rather than repeatedly recalculated. The weights are stored in a data structure called the ***class-of-service weights array*** (COS weights array). The index field in the COS table entry points to the entry in the COS weights array where the weights for that class of service are stored. The class-of-service manager ensures that whenever a COS table entry is altered that the values in the COS weights array for that class of service will be subsequently recalculated. This is done by setting the index value to 0.

For all sessions (LU-LU and CP-CP), RSS in the network node server for the node containing the origin NAU uses the COS table to compute the best session route to the node containing the destination NAU. Because CP-CP sessions exist only between adjacent nodes, they use only single-hop routes. Even for such short routes, however, the RSS component in each node uses the COS table to compute the optimal TG for its CP-CP session. When the adjacent nodes are connected by multiple parallel links, RSS selects the optimal TG.

Specifying COS Characteristics

The rows in an APPN COS table entry define node and transmission group (TG) characteristics that are acceptable for the specified class of service. Each row defines either a set of node characteristics or a set of TG characteristics. The characteristics include such factors as security, cost-per-connect time, and effective capacity.

For any given characteristic, the field representing it contains a range of acceptable values for the characteristic. For example, cost-per-connect time for a TG ranges from 0 to 255, with 0 representing the minimum cost. The range enables RSS to test the cost-per-connect time of an actual TG to determine if the TG is acceptable for a given class of service. The APPN route calculation technique is described under "Calculating Routes for BIND Requests" on page 153.

Selecting Routes in Subarea Networks

Route selection takes place at the time of session initiation. When an end user requests an LU-LU session, the session-initiating logical unit places the specified class of service in the session-initiation request. In subarea networks, the system services control point (SSCP) handling the request uses this class of service as an entry into the class-of-service table. An end user requests a given class of service either directly, as a class-of-service name, or indirectly, by a mode name. A ***mode name*** identifies a set of session parameters that the requesting logical unit can support. For further information on mode names, see Chapter 8, "Transporting Data Through the Network."

Selecting a Route

When an LU-LU session is requested, the LU's SSCP locates the appropriate entry in the COS table, and passes the list of VRN/transmission priority pairs to path control in the origin LU. Path control's virtual route manager (VR manager) then selects the first available VRN/transmission priority pair for the specified class of service. If the explicit route to which the virtual route is mapped is inactive, the VR manager attempts to activate the explicit route. If it is unable to activate the explicit route, it tries the next VRN/transmission priority pair. If the VR manager is unable to activate an explicit route for the specified class of service, it informs the LU that no LU-LU session can be activated. The LU requesting the session must then resubmit the session-initiation request at a later time. If the LU requesting the session resides in a T5 node, the VR manager notifies the LU when a virtual route that can provide the specified class of service becomes available.

Note that as long as the explicit route for the first virtual route specified in a VR list is operable, that virtual route is selected for every session requesting the same class of service. This can result in congestion caused by too many sessions assigned to the same virtual route. The distribution of LU-LU session traffic among the virtual routes available for a class of service can be controlled by an access method *user exit* routine. VTAM for example, provides a user exit that enables an installation to manage the reordering of the VR list each time an LU-LU session with the same COS is activated.

Activating a Route

Access methods and network control programs in subarea nodes activate explicit and virtual routes as needed. Explicit routes must be active before an SSCP can assign a session to a virtual route. The network operator never enters commands to activate routes. Path control activates a virtual route automatically when a session requires a path.

To activate an explicit route, path control in the originating subarea sends an Explicit Route Activate (NC-ER-ACT) request to path control in the destination subarea node. Upon receipt of the NC-ER-ACT, path control in the destination subarea node determines the length of the explicit route by totaling the number of transmission groups traversed in the path. Then it verifies that the route:

- Is usable
- Connects the origin and destination subarea nodes
- Is reversible
- Does not pass through any node more than once.

Selecting Routes in APPN Networks

As in a subarea network, a control point handling a session request in an APPN network uses the specified class of service as an entry into a class-of-service table, and classes of service are specified by mode name. Beyond these similarities, however, the concepts of route specification and selection in an APPN network differ from those in a subarea network.

In an APPN network there are no predefined routing tables by which to associate network resources with routes through the APPN network. Therefore, when an LU-LU session is initiated, the network node control point serving the origin LU

must perform two major services: it must locate the destination LU and calculate the best route for the session to the destination LU.

The basic APPN route selection algorithm is used also in HPR. However, an HPR node can calculate an all-HPR path. Nondisruptive path switch requires that both the original path, as well as the path taken after the original path fails, must contain only HPR nodes. The routes (paths) within HPR subnets are expressed in terms of a sequence of automatic network routing (ANR) labels instead of TG numbers and CP names. The basic APPN design provides various ways to make HPR links preferred over APPN links. The installation manager must define characteristics in order to make HPR routes preferred over APPN routes.

Locating the Destination LU

Locating the destination LU for an LU-LU session is the responsibility of a control point's directory services. The process begins with the *origin LU* (OLU) requesting its local control point to activate a session to the *destination LU* (DLU). Directory services (DS) then has the responsibility of undertaking a search for the DLU. A *search* involves (1) sending a ***Locate Search*** request to the control point (CP) of the node containing the DLU (the *owning* CP) and (2) receiving back a Locate Search response containing the owning CP's name.

In an end node, DS begins a search by interrogating the local directory database. In an end node, the local directory contains entries for local LUs and those in adjacent nodes not known to, or able to be located by, its network node server. (Such adjacent nodes may include, for example, LEN nodes not otherwise connected to the APPN network.) If DS fails to find the DLU in the end node's local directory, it sends a Locate Search request to its network node server over its contention-winner CP-CP session. This type of search is referred to as a one-hop search. A ***one-hop search*** traverses only a single transmission group, either from an APPN end node to its network node server or from a network node server to a client APPN end node.

Besides locating a destination resource, Locate Search requests and replies serve another purpose as well: they contain information needed by the originating LU's network node server for route calculation. Resources that are strictly local to a node are not recorded in the network topology database. This implies that information on an end node's transmission groups (TGs) is not accessible in the network topology database. But the origin network node needs this information for calculating the entire route, from end node to end node. Therefore, an end node includes data fields called *TG vectors* describing the end node's transmission groups in the Locate Search request sent to its network node server. One ***TG vector*** (also called a ***tail vector***) is included for each TG connecting the origin end node to a network node or a connection network. The destination end node will return TG vectors on its Locate Search reply for the TGs connecting it to network nodes, connection networks, and the origin end node.

In a network node, the CP undertakes a search as a result of having received either a session-initiation request from a local LU or a Locate Search request from a CP in a client end node. As in an end node, a network node DS begins by interrogating the local directory database. In a network node, the local directory contains not only local entries, but *may* contain domain and other-domain entries as well. (Recall that the only resources that *must* be in a directory database are a node's local resources.) DS in a network node undertakes one of three kinds of searches depending on whether or not it finds the DLU in its local directory.

- If the network node finds the DLU in its directory, it begins a directed search.
- If the network node fails to find the DLU in its directory or if the directed search initiated by the network node fails, and a CDS exists, the network node sends a referred search to the nearest CDS. If the CDS, based on its own directory, successfully locates the resource for the network node, it sends the positive reply (with TG vectors) to the network node. Otherwise, the CDS queries other CDSs for the resource. Any positive reply it obtains, it forwards to the network node (after also caching the information). If none of the CDSs replies to the search positively, the original CDS initiates a broadcast search. If this also fails, it returns a negative reply to the origin network node, and the network node then abandons the search.
- If the network node fails to find the DLU in its directory and a CDS does not exist or cannot be reached, the network node begins a broadcast search.

For further information on CDS, see "Central Directory Server (CDS)" in Chapter 1, "Introduction" and "Central Directory Server (CDS)" in Chapter 5, "Activating the Network."

The Broadcast Search

With a ***broadcast search***, Locate Search requests are sent to all nodes in the network until the one containing the name of the destination LU in its local directory is found. Only network nodes perform broadcast searches. A network node participates in a broadcast search in two ways: (1) it propagates a Locate Search request to all attached network nodes, and (2) it performs a ***domain search*** by sending Locate Search requests to those of its client APPN end nodes that allow Locate Search requests for unregistered resources. (APPN end nodes signal their willingness to accept such requests in the CP capabilities exchanged during CP-CP session initiation.)

The network node does not assume that the destination LU, because it was not registered by any of the network node's end nodes, is not located in the network node's domain. An APPN end node might be unable to, or its operator might elect not to, register some or all of its resources with its network node server. This might be the case if the end node contains a large number of infrequently accessed resources. In that case, it might be more efficient for the network node to perform an occasional domain search than to register all the end node's resources during CP-CP session activation.

If a network node that receives a broadcast search finds the destination LU in its local directory, the LU can be recorded as a local, domain, or other-domain entry. If the LU is a local entry, a positive response to the Locate Search request is returned. If it is a domain entry, a directed search is sent to the CP of the node containing the destination LU. (For further information, see "The Directed Search" on page 150.)

A domain entry in a network node's directory database can be a ***wild-card entry***, which associates a resource with an attached end node or boundary node by default. This allows a network node to represent for search purposes a vast number of LUs (for example, in an attached subarea network) that are not registered explicitly in the APPN network. The network node replies positively to the search requests, allowing the subsequent specific BIND requests to be sent through it into the node or network represented by the wild-card entry.

Similarly, an APPN network node may reply positively to search requests based on only partial identification of the target resource. For example, the entry "RAL*" would direct all search requests for resources beginning with "RAL" to the end node with which the entry is identified.[10]

Wild-card entries provide support for APPN end nodes that choose not to register large numbers of resources because of the overhead involved. Wild-card entries also provide support for LEN end nodes, the resources for which would otherwise require static definition in the network node server. When the end node is a LEN end node (which does not support Locate Search requests), the location of a resource cannot be verified by a directed search to the end node. The origin network node may therefore receive positive search replies from more than one destination network node. The origin network node can tell, however, whether a positive Locate Search reply resulted from a wild-card entry or an explicit entry, based on an indicator carried in the Locate Search reply.

During a broadcast search, a network node can receive multiple Locate Search requests, from different adjacent network nodes, for the same resource. In that case, the network node discards the duplicate requests rather than continue to propagate them through the network. The parameter in the Locate Search request that enables the network node to recognize a duplicate request is the ***fully qualified procedure correlation identifier*** (FQPCID), which uniquely identifies a session across an APPN network. It is generated by an origin node during session-initiation. It includes an 8-byte ***procedure correlation identifier*** (PCID) concatenated with the variable-length network-qualified name of the CP that generated the PCID, thus guaranteeing the uniqueness of the FQPCID. The PCID contains a 4-byte value derived from the network-qualified name of the originating CP and a 4-byte counter, or sequence number (initialized at IPL time using a hashing function and the time-of-day clock). The sequence number is incremented by 1 each time session services assigns an FQPCID. For implementations that do not have a suitable clock, the PCID value is safe-stored across IPLs. All messages sent between nodes that relate to a particular session, including Locate Search, BIND, and UNBIND, contain the FQPCID for that session.

An APPN end node can receive a Locate Search request from its network node server when the server has received either a broadcast or directed search. An APPN end node responds to a Locate Search request with a reply indicating whether or not its local directory contains the name of the destination LU. The reply is then routed back to the origin network node and the origin end node. The destination end node includes its TG vectors in the Locate Search reply. A TG vector is sent for every TG connecting the destination end node to a network node, a connection network, or the origin end node. (The destination end node can determine whether or not it has a TG connecting directly to the origin end node by searching its local directory for the CP name of the origin end node, which is carried in the Locate Search request.)

When the destination network node receives a positive Locate Search reply from its client end node, it may update its directory database with a cache domain entry. If the network node had been conducting a domain search, the reply discloses the

---

[10] An APPN end node may also use partially specified entries in its local directory to reduce its size, and reply positively to search requests for any LU represented by the entry. Of course, each such LU would need to be known in the directory used by the address space manager (ASM), which processes the BINDs.

location of the destination LU. The destination network node then sends the Locate Search reply back to the origin network node.

When the origin network node (the node that initiated the broadcast search) receives a positive reply to the search, it first updates its local directory by adding the location of the destination LU as an other-domain cache entry. It then uses the TG vectors returned by the node containing the destination LU, together with those sent originally by the origin end node (if any), to calculate a route for the session. If the network node contains the origin LU, the network node then reports back to the LU. Otherwise, it sends the Locate Search reply, containing the selected session route, to the CP in the end node containing the origin LU, which, in turn, reports back to the origin LU. If the origin and destination end nodes share a common TG or connection network (CN), the selected route may traverse that TG or CN rather than pass through the network node server or any other directly attached network node.

A Broadcast Search Sequence

The following section describes the sequence of messages that flow as a result of a broadcast search (see Figure 76). The destination LU is unknown to the originating node, and is recorded only in the directories of the destination end node and its network node server. No central directory server (CDS) is present in the network.

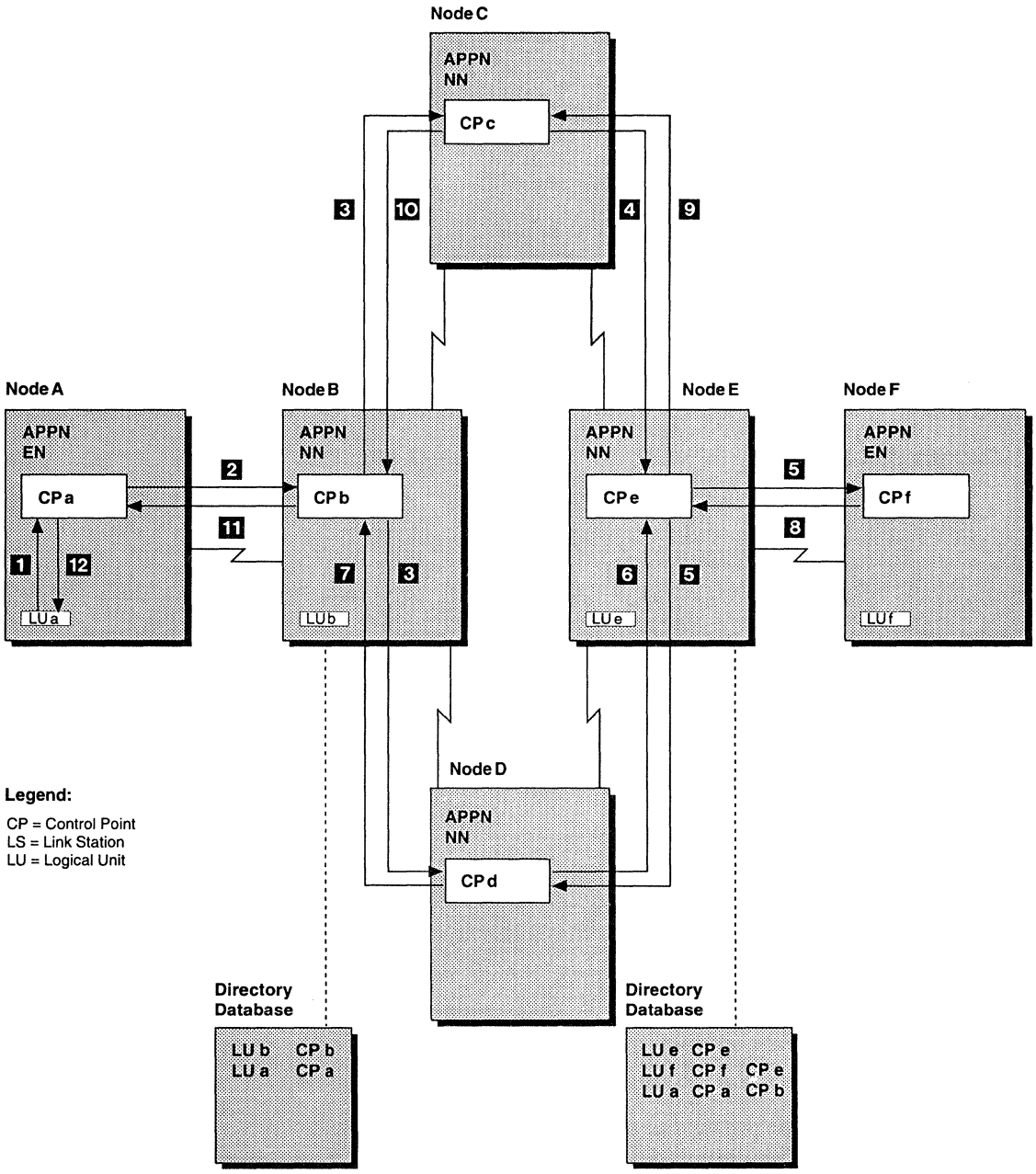


Figure 76. A Broadcast Search

The following comments correspond to the numbers in Figure 76.

1 LU *a* sends a session-initiation request to CP *a* for a session with LU *f*.

2 CP *a* interrogates node A's local directory without success, and sends a Locate Search request to CP *b* over a CP-CP session. The request includes a TG vector for node A's TG to node B.

3 CP *b* interrogates node B's local directory without success, and (since no CDS exists) initiates a broadcast search by sending Locate Search requests to CP *c* and CP *d*.

4 CP *c* interrogates node C's local directory without success, and forwards the Locate Search on to node E.

5 CP *e* receives the Locate Search request from CP *c*, and finds the DLU in its directory database. Despite the fact that the DLU is recorded in the directory, CP *e* initiates a one-hop directed search to node F, and also forwards the Locate Search request on to CP *d*. (If the DLU had not been in CP *e*'s directory, CP *e* would have initiated a domain search, which, in this case, would have included only node F.)

6 CP *d* receives the Locate Search from CP *b*. CP *d* interrogates node D's local directory without success, and forwards the Locate Search on to node E. Because CP *e* had already sent a Locate Search to CP *d* for the same LU, CP *e* considers the request the equivalent of a negative reply and discards it. (*Crossed* Locate Search requests serve as negative replies to the two nodes that receive them.) CP *e* continues to wait for a reply from CP *f*.

7 CP *d* receives the Locate Search request from CP *e*. Because it had already sent a Locate Search to CP *e* for the same LU, it considers the request the equivalent of a negative reply. CP *d* sends a negative reply to CP *b*. CP *b* receives the negative reply from CP *d* and continues to wait for a reply from CP *c*.

8 CP *f* receives the Locate Search request from CP *e*. It finds the LU in its local directory and returns a positive reply to CP *e*. The reply includes a TG vector for node F's TG to node E.

9 CP *e* receives the positive Locate Search reply from CP *f* and forwards it to CP *c*. (If CP *e* had not already had LU *f*'s location and LU *a*'s location recorded in its directory database, it would cache the location information at this time.)

10 CP *c* receives the positive Locate Search reply from CP *e* and forwards it to CP *b*.

11 CP *b* receives the positive Locate Search reply from CP *c*. It caches the location of LU *f* as an other-domain entry in its directory database, computes a route for the session, and forwards the Locate Search reply with the route selection information to CP *a*. (If CP *b* did not already have LU *a's* location recorded in its directory database, it would also cache that location information at this time.)

12 CP *a* receives the positive Locate Search reply from CP *b* and sends a Control Initiate message containing the route information to LU *a*. LU *a* can now build a BIND to be sent to LU *f*, using the route information.

The Directed Search

In a ***directed search***, the network node server for the node containing the OLU sends a Locate Search request directly to the network node server for the node where it believes the DLU to be. Like broadcast searches, multihop directed searches are performed only by network nodes. (End nodes perform only one-hop searches.)

A CP does not assume that because a remote destination LU is recorded in its directory that therefore the LU has been located. It is possible that since the time the LU was recorded it has been moved, renamed, or even destroyed. The Locate Search request, therefore, verifies that the information in the directory is accurate.

When a directed search reaches the destination network node, the network node looks up the CP name of the DLU in its directory database. If the DLU is not located in the network node, but in a client end node, the network node routes the Locate Search request to the end node containing the DLU. If the destination CP finds the LU in its directory, it returns a positive Locate Search reply, which includes its TG information, to the origin network node server CP. When the origin network node server receives a negative reply and a CDS exists, it performs a referred search; that is, the origin network node server forwards the search request to the central directory server. The central directory server then handles the search request. If the origin network node server CP receives a negative reply to the referred search, the network node abandons the search. This is because the CDS has already queried all other CDSs and none of the CDSs contained information on the resource and a broadcast search has also failed. If a CDS does not exist or cannot be reached, the network node begins a broadcast search. In any case, it reports back to the origin LU whether the search was successful.

Calculating Routes for Locate Search Requests

There are two kinds of route calculation that a network node server's route selection services (RSS) performs. It calculates a route for transmitting a directed Locate Search request to the destination network node (the network node server of the node containing the DLU), and it calculates a route for transmitting the BIND request, which activates the LU-LU session, to the destination LU. These two routes need not coincide. The Locate Search route represents the most direct route, using adjacent CP-CP sessions, to the destination. network node. The BIND route represents the optimal route, based on the requested class of service, to the destination LU. For information on calculating the BIND route, see "Calculating Routes for BIND Requests" on page 153.

When calculating either kind of route, RSS uses the topology database to construct a route selection control vector, which is subsequently appended to the Locate Search or BIND request. A ***route selection control vector*** (RSCV) is a string of control vectors representing the components of a route. The RSCV in a Locate Search request represents a ***locate chain***. The control vectors for a locate chain identify contiguous network nodes along the route, from the origin network node to the destination network node. Locate Search requests and replies can therefore be routed from network node to network node, over the CP-CP sessions.

A Directed Search Sequence

The following section describes the sequence of messages that flow as a result of a directed search (see Figure 77 on page 152). The destination LU is recorded in the directory of the origin network node.

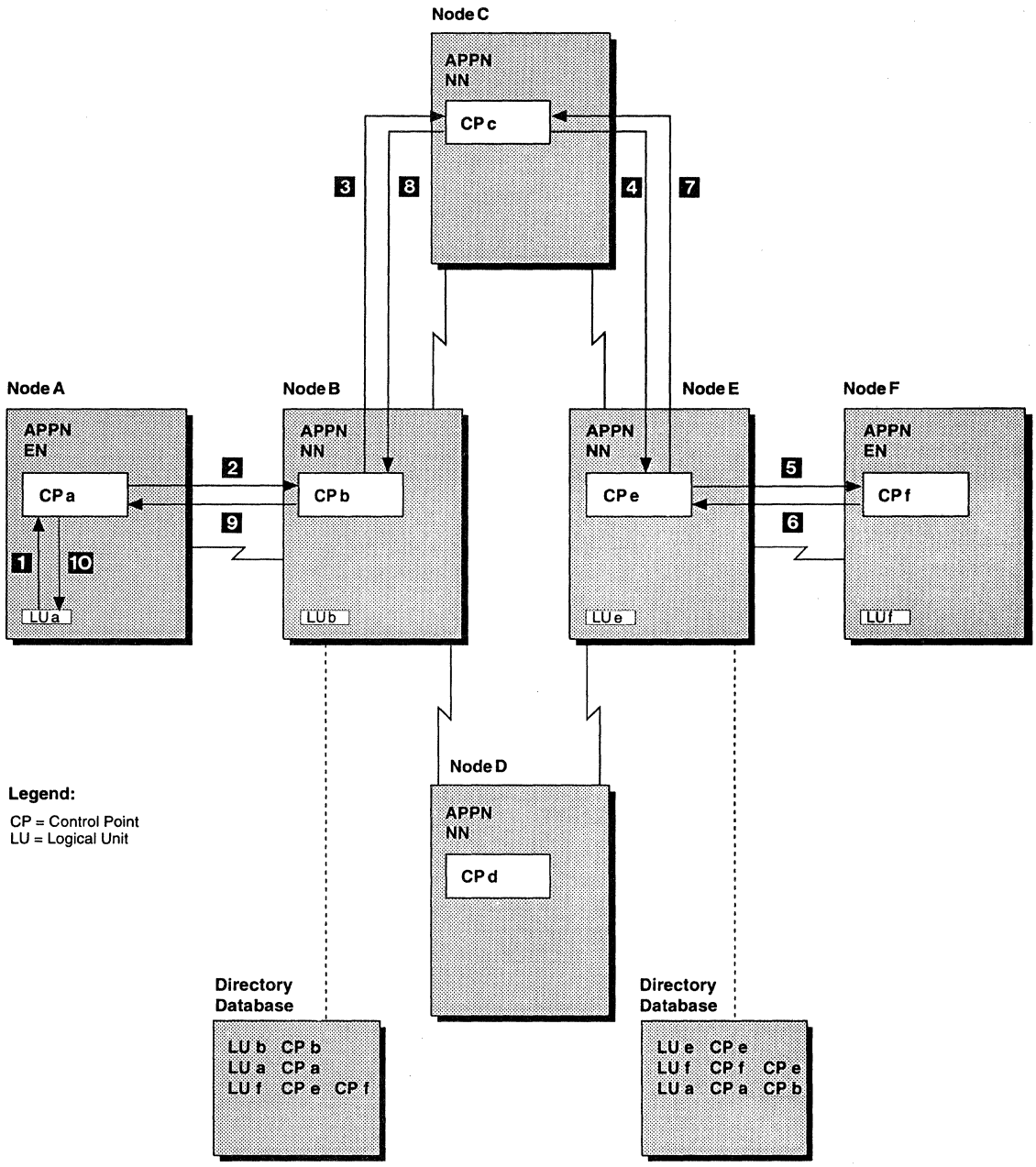


Figure 77. A Directed Search

The following comments correspond to the numbers in Figure 77.

1 LU *a* sends a session-initiation request to CP *a* for a session with LU *f*.

2 CP *a* interrogates node A's local directory without success, and sends a Locate Search request to CP *b* over its contention-winner CP-CP session.

3 CP *b* interrogates node B's local directory and finds the LU. It then builds an RSCV from the network topology database for the route to node E. Assuming that node C has been selected as the next node in the route, CP *b* sends the Locate Search request to CP *c*.

4 CP *c* receives the Locate Search request from CP *b*. CP *c* recognizes that the request represents a directed search, determines the next node in the route from the RSCV, and forwards the Locate Search to CP *e*.

5 CP *e* receives the Locate Search request from CP *c*, and finds the DLU in its local directory. CP *e* recognizes that the request represents a directed search, and initiates a one-hop search to CP *f*.

6 CP *f* receives the Locate Search request from CP *e*. It finds the LU in its local directory, and returns a positive reply to CP *e*.

7 CP *e* receives the positive Locate Search reply from CP *f* and forwards it to CP *c*.

8 CP *c* receives the positive Locate Search reply from CP *e* and forwards it to CP *b*.

9 CP *b* receives the positive Locate Search reply from CP *c*, computes a route for the session, and forwards the Locate Search reply to CP *a*.

10 CP *a* receives the positive Locate Search reply from CP *b* and sends a Control Initiate message containing the route information to LU *a*. LU *a* can now build a BIND to be sent to LU *f*.

Use of a Central Directory Server in a Search

When ***central directory servers (CDSs)*** are present in the topology subnet, the network nodes search the closest CDS whenever a destination LU is unknown or a directed locate search has failed. The CDS then takes responsibility for locating the destination LU. In general, if an unsuccessful referred search (network node server to CDS) is made to the CDS, there is no need for the network node server to send another search to any other CDS to find the resource. This is because a CDS exhausts the search for a resource if the resource is not in its central directory. If a negative reply to the referred search indicates a session outage in the path to the CDS, an additional search is tried. This search is sent to the original CDS if a route exists; otherwise, the search is sent to another CDS in the topology subnet. If no other CDS is available, a broadcast search is initiated by the network node server. Any network node referring a directory query to a CDS is called a ***central directory client***.

Calculating Routes for BIND Requests

When calculating a *locate-chain* route for a Locate Search request, route selection services (RSS) is not concerned with any characteristics of the route, such as efficiency or reliability. The individual CPs along the route have already chosen the transmission groups (TGs) over which their CP-CP sessions were activated. Therefore, whatever characteristics are manifested by the nodes and TGs along the route are pre-existing and beyond the ability of RSS to control.

In contrast to a Locate Search route, the characteristics of a BIND route are important to RSS. A route for a BIND request, and the session activated by it, is a series of consecutive nodes and transmission groups (TGs) that best serves the requested class of service. The RSCV in a BIND request identifies contiguous TGs along the route, from the node containing the OLU to the node containing the DLU. In determining the route, the characteristics of the nodes and the TGs composing it are those that best match the characteristics required by the specified class of service.

The Least-Weight Routing Algorithm

The logic by which an APPN network node control point determines the route for a session is referred to as the ***least-weight routing algorithm***. The algorithm is invoked by a *route request* sent to route selection services by session services. The route request contains the endpoint transmission group vectors, one for each endpoint, which are required for complete route specification. Other inputs to the algorithm include the topology database, the specified class of service, and the COS database. The output from the algorithm is the complete route selection control vector defining the route for the BIND request, and the session activated by it.

The logic for the algorithm proceeds as follows:

1. Find the destination network node in the topology database and determine all possible routes back to the origin network node. A route consists of a unique set of intermediate network nodes and connecting transmission groups (TGs).
2. Look up the table entry for the specified class of service in the COS database.
3. For each possible route, compare the characteristics of the component nodes and transmission groups to the ranges of acceptable characteristics as defined in the COS table for that class of service. For each component node or TG in a given route:
   a. If the index value in the COS table entry is nonzero, see if a weight value for the component exists in the COS weights array entry pointed to by the index. If so, assign that weight to the component; otherwise, continue.
   b. Look up the characteristics of the component in the topology database.
   c. Select the first row of characteristics for the component in the COS table entry. If the component is a node, use the rows of node characteristics; otherwise, use the TG characteristics.
   d. For a given row, if each characteristic of the component falls within the range of tolerance for that characteristic as specified in the row, assign a weight to that component as specified by the weight field in that row. The weight could be a constant, or a function of one or more of the characteristics of the component. Store the computed weight in the COS weights array.
   e. If any characteristic for the component falls outside the range of tolerance for the characteristic, select the next row of characteristics in the COS table entry and try again. (The characteristics for consideration of a component include temporary states such as the operability of a TG or congestion within a node.)
   f. If there is no row of characteristics in the COS table entry for which the component can qualify as acceptable, assign an infinite weight to that component.
4. When all components of a route have been weighted, determine the weight of the route by summing the weights of its components.
5. When the weights of all routes have been determined, select the route assigned the least weight.
6. Create an RSCV from the component intermediate nodes and TGs of the selected route, and from the specified endpoint TGs.

Route calculation in an APPN network is a complex procedure and, for lengthy routes, may involve considerable overhead. To enhance efficiency in the network, the results of the least-weight routing algorithm are not discarded. Following a route calculation, the route is stored in a database at the origin network node. Then, when a route for a session involving the same destination network node and class of service is requested, the CP at the origin node can retrieve the results of a prior calculation from the database.

The Tree Database

The ***tree database*** is the database at a network node where the optimal routes to other network nodes are stored. Routes from the network node to others in the network form a *tree* structure, with the origin network node at the *root* of the tree. Within the tree database there is one tree per class of service, and within each tree are stored the optimal routes for that class of service. A tree is comprised of network nodes, their connecting transmission groups, and the weights assigned to both.

When route selection services receives a route request, it first checks the tree database to see if a route has already been calculated to the destination network node for the specified class of service. If so, RSS uses the same route, together with the endpoint TGs provided, to construct the RSCV. If such a route has not been previously stored in the database, RSS calculates and stores it.

Figure 78 on page 156 illustrates a tree database containing trees for two classes of service: *Batch* and *Interactive*. The trees contain the optimal routes from node B to nodes C, D, and E.

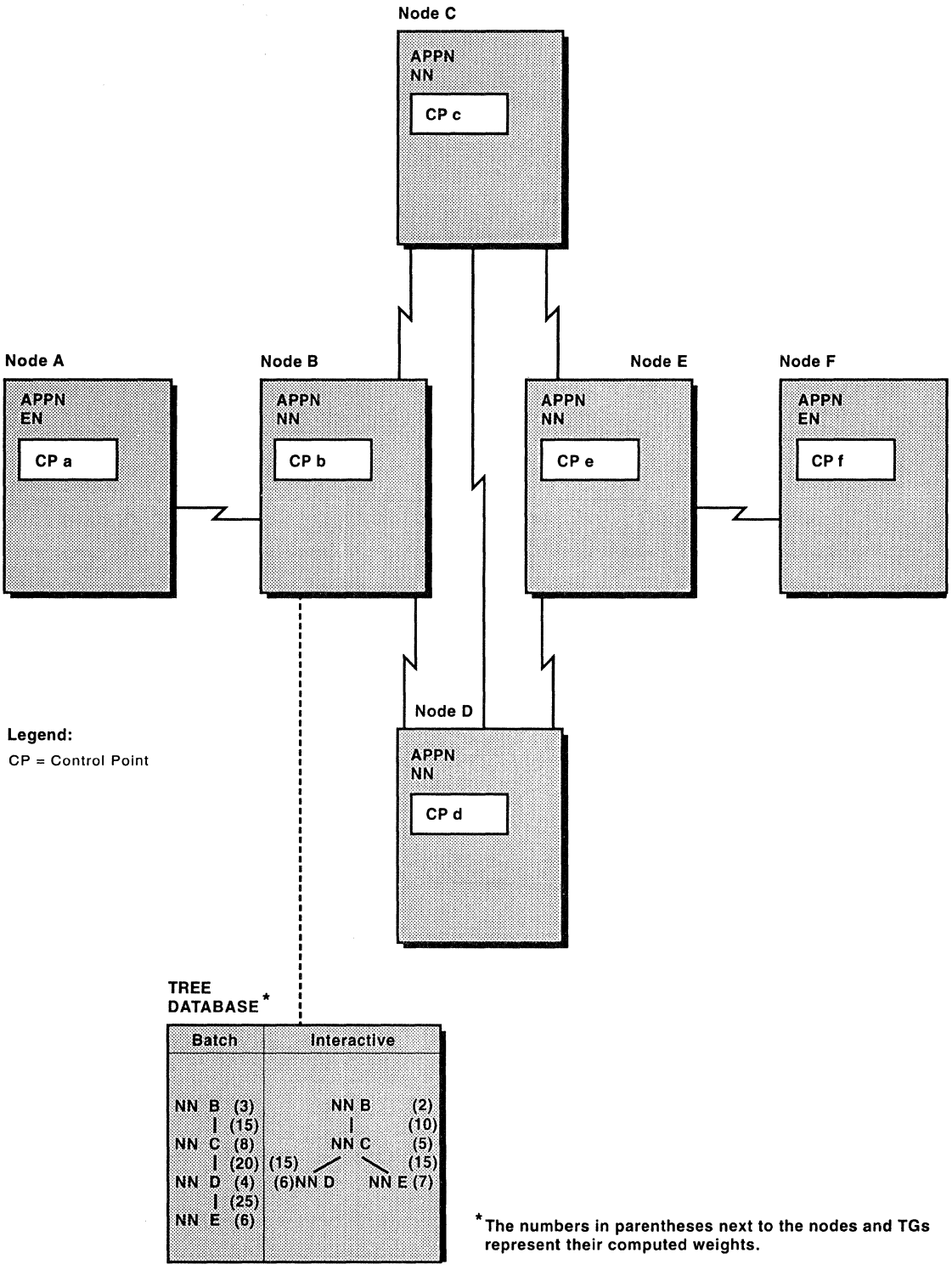


Figure 78. The Tree Database

The tree for class of service *Batch* shows that the optimal route from node B to node E runs through both nodes C and D. Although a more direct route would appear to run through only node C, the chosen route can be accounted for by the weights specified in the COS table for class of service *Batch*. Evidently, the weights specified for TGs with the characteristics of the TG connecting node C to node E prevented the selection of that TG. Among the possible routes from node

B to node E, the one with the least weight was assigned a weight equal to the sum of those shown in the tree. An RSCV constructed for the route would consist of three TGs: the TG connecting nodes B and C, the one connecting nodes C and D, and the one connecting nodes D and E.

The tree for class of service *Interactive* shows that the optimal route from node B to node E now runs directly from node C to node E. Evidently, the weights specified under class of service *Interactive* for TGs with the characteristics of the TG connecting nodes C and E favored the selection of that TG. Among the possible routes from node B to node E, the one with the least weight was assigned a weight equal to the sum of those shown in the tree, excluding node D and the TG branching to it. An RSCV constructed for the route would consist of two TGs: the TG connecting nodes B and C, and the one connecting nodes C and E.

Calculating Routes for LEN End Nodes

The routing information in a LEN end node is predefined. This includes the locations of resources in adjacent end nodes and those accessed through its network node server. In the latter case, it appears to the LEN end node as though the resources are located in the network node server. The LEN end node is unaware of the services performed by its network node server.

Upon receiving a session-initiation request from a local LU for a destination LU that it believes to be in its network node server, the LEN end node constructs a BIND and sends it to the network node. The network node server then performs a search for the destination LU. When the network node server receives the Locate Search reply, it calculates a session route, and affixes the resulting RSCV to the BIND request before sending the BIND to the destination LU. When the network node receives the BIND response, it passes the response to the LEN end node, as it would to an APPN end node, but without the RSCV.

Similarly, when a network node server receives a Locate Search request from outside its domain for an LU residing in an attached LEN end node, it handles the Locate Search reply on behalf of the LEN end node. The LEN end node's resources must previously have been defined to the network node through the network node's node operator facility. When the network node server receives the BIND request, the network node forwards the request to the LEN end node, just as it would to an APPN end node, but without the RSCV. When the LEN end node returns the BIND response, the network node server affixes the original RSCV to the response before sending the response to the origin network node.

Calculating Routes Across Interconnected APPN Networks Containing Peripheral Border Nodes

As discussed in "Interconnecting Networks" in Chapter 1, "Introduction," two APPN networks can be interconnected by way of a peripheral border node in one or both APPN networks. The two partners establish CP-CP sessions between them over which search protocols (but not topology database updates) can be conducted.

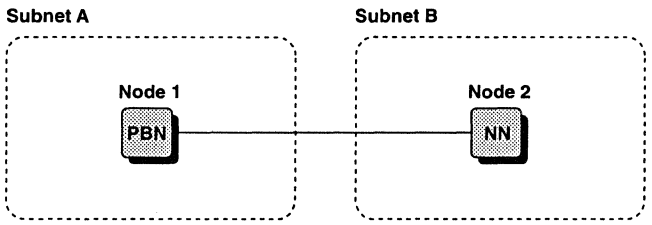


Figure 79. Simple Peripheral Border Node Configuration

For example, as shown in Figure 79, one subnet may contain a peripheral border node and its partner subnet may contain a basic network node. The peripheral border node (hereafter called *Node1*) may take on the end node role and knows that its partner (hereafter called *Node2*) is a network node. The partner network node, Node2, acts as if it is unaware, however, that Node1 is a network node. To Node2, Node1 is indistinguishable from any other client APPN end node.

Node1 performs the protocols that allow cross-network searches and session initiation to take place. Such protocols involve separate route calculations for each interconnected network. For both directed search and session activation requests, Node1 recalculates and swaps the appropriate locate-chain and session-route RSCVs for the requests as they traverse the node.

When Node1 receives a Locate Search request, it initiates a search in the interconnected network if the net ID of the LU being searched for is the same as the net ID of the interconnected network. If the Locate Search request originated in its native network, Node1 first removes the RSCV affixed to the Locate Search request. It then acts as a client APPN end node, passing the request, together with its own TG vector information, to its partner, node B, in the interconnected nonnative network. Node2 then conducts a search in the interconnected network, just as it would for any client APPN end node. When the Locate Search reply is received, Node1 saves the session-route RSCV calculated by Node2 and replaces it by TG vector information representing its connection to Node2, and returns the reply to the origin network node in its native network.

If Node1 receives a Locate Search request from Node2 in the interconnected network and the net ID of the LU being searched for is the same[11] as Node1's net ID, it first removes the locate-chain RSCV from the request. It then places TG vector information for itself and its partner network node (which it represents in its native network as a client end node) in the request. It then conducts a search for the destination LU. If Node1 finds the destination LU in its directory, it conducts a directed search; otherwise, it sends a directed search to the CDS if one exists, or it conducts a broadcast search. When the Locate Search reply is received, Node1 deletes the received TG vectors from the reply it will forward (although using them later to calculate the portion of the session path from it to the destination LU) and replaces them by the TG vectors for the intersubnetwork TG connecting it to Node2, and sends the reply (indicating that Node1 owns the LU) to its partner, Node2, in the interconnected nonnative network.

[11] A current restriction of the architecture is that a peripheral border node does not search for resources in "casually connected" end nodes (those having nonnative net IDs).

A similar RSCV-swapping process takes place for cross-network LU-LU session activation. If Node1 receives a BIND request from a node in its native network for an LU in the other, it removes the session-route RSCV affixed to the BIND, saving it so that it can be returned to the origin LU on the BIND response, and substitutes the session-route RSCV that Node2 had sent to it in the previous corresponding Locate Search reply. It then passes the request to its partner, Node2, in the interconnected network. When the BIND response is received, Node1 replaces the RSCV in the response with the original one, and returns the response to the origin network node in its native network.

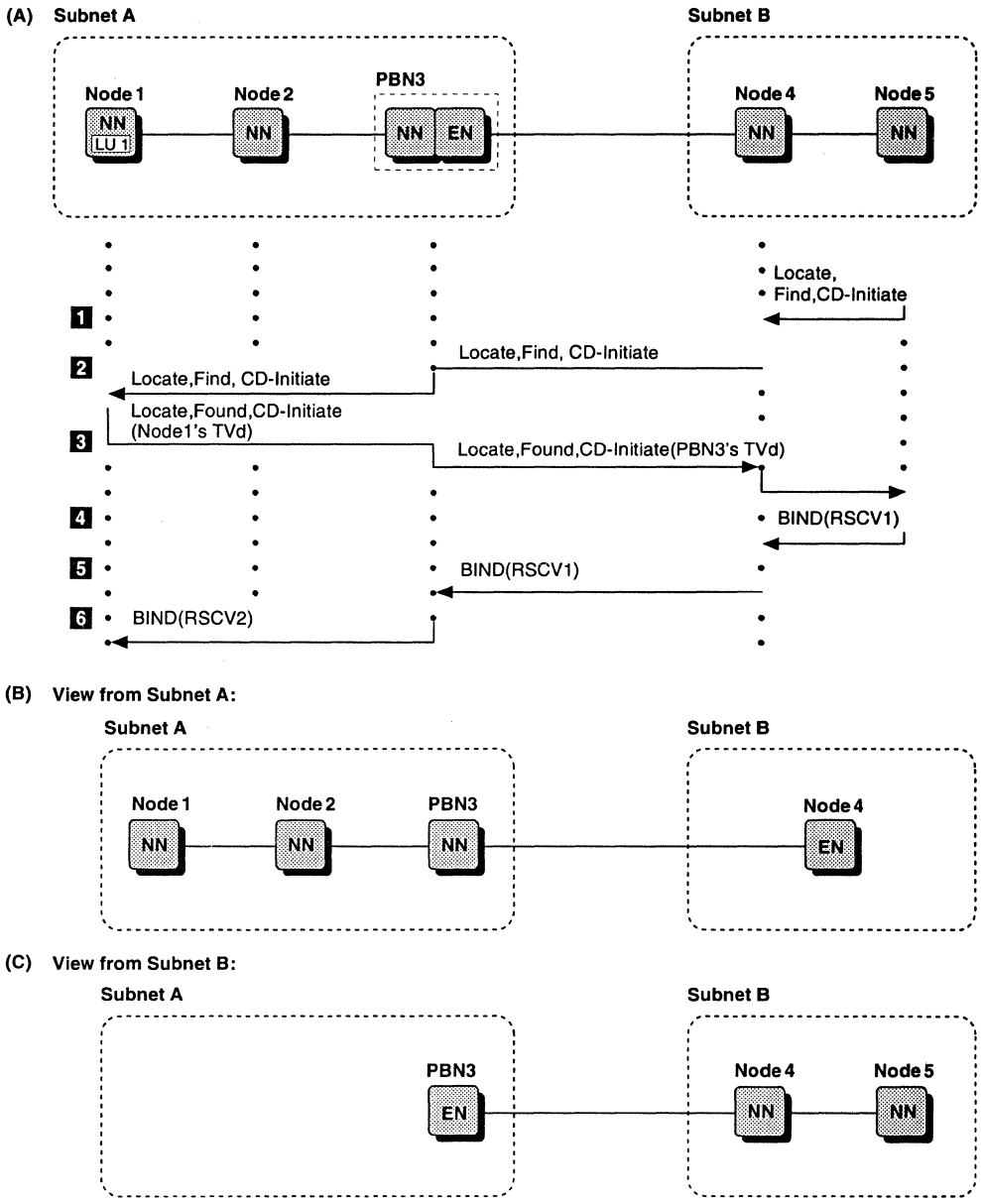
If Node1 receives a BIND request from Node2 in the interconnected network, it removes the session-route RSCV from the request. Node1 then calculates a new RSCV for the BIND, from itself to the destination network node in its native network. When the BIND response is received, Node1 replaces the session-route RSCV in the response with the original one, and sends the response to Node2 in the interconnected network.

CP Capabilities Exchange in APPN Subnets Containing Peripheral Border Nodes

A peripheral border node presents the capabilities of a network node to native nodes and client nonnative APPN end nodes; it presents the capabilities of an APPN end node to nonnative network nodes. A peripheral border node does not indicate border node support to its adjacent partner nodes during CP capabilities exchange.

Session Initiation Across Interconnected APPN Networks Containing Peripheral Border Nodes

Figure 80 (A) illustrates the flow sequence involved in a session initiation request through a border node from a PLU in a PLU-initiated nonnative network. Figure 80 (B) and (C) illustrate the way subnet A and subnet B, respectively, view the interconnected subnets.



Legend:

Locate	= Locate GDS variable	TVd	= Destination tail vectors
Find	= Find GDS variable	PBN	= Peripheral border node
Found	= Found GDS variable	NN	= Network node
CD-Initiate	= CD-Initiate GDS variable	EN	= End node
RSCV	= Route selection control vector	LU	= Logical unit

Note: The Locate GDS variable includes the locate-chain RSCV for a directed search (as opposed to a broadcast search).

Figure 80. Nonnative PLU-Initiated Session Initiation from Subnet B to Subnet A

The following comments correspond to the numbers in Figure 80 (A):

1 Node5 sends a Locate/Find (directed or broadcast) to Node4 to search for *A.LU1*. Node4 forwards the Locate/Find to PBN3 in subnet A, since PBN3 appears as an APPN end node to Node4 and PBN3 has requested to be included on domain broadcasts.

2 PBN3 forwards the Locate/Find to Node1 via Node2.

3. Node1 returns a Locate/Found to PBN3 to say that the destination LU has been found. The Locate/Found contains Node1's destination TG vector (TVd). PBN3 forwards the Locate/Found to Node4 in subnet B with PBN3's destination TG vector (instead of Node1's TVd).

4. Node5 computes session-route RSCV1 with PBN3's TVd and forwards the BIND containing RSCV1 to Node4.

5. Node4 forwards the BIND to PBN3 in subnet A. Upon receipt of the BIND, the PBN3 verifies the BIND and the location of the SLU in its cache (based on the FQPCID). Once the BIND is verified and the location of the SLU is determined, RSCV2 is computed.

6. PBN3 forwards the BIND containing session-route RSCV2 to Node1.

Calculating Routes Across Interconnected APPN Networks Containing Extended Border Nodes

APPN networks can be interconnected by way of an extended border node in one or more APPN networks, with the option of having peripheral border nodes at the peripheral networks. The networks establish CP-CP sessions between them over which search protocols (but not topology database updates) can be conducted.

The extended border node appears as a basic network node with extended border node capability to other extended border nodes. It appears as a client APPN end node to a peripheral border node or to a basic network node in an interconnected network.

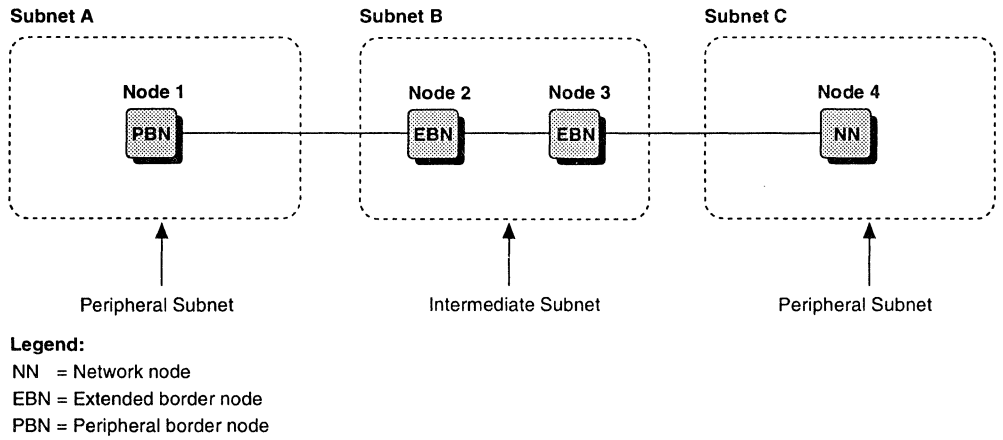


Figure 81. Simple Extended Border Node Configuration

For example, as shown in Figure 81, a peripheral subnet may contain one peripheral border node (hereafter called *Node1*); an intermediate subnet may contain two extended border nodes (hereafter called *Node2* and *Node3*); a peripheral subnet may contain a basic network node (hereafter called *Node4*).

Node1 performs the protocols that allow cross-network searches and session initiation to take place. Such protocols involve separate route calculations for each subnet as the Locate Search reaches each subnet. For both directed search and session-activation requests, Node2 and Node3 recalculate and swap the appropriate locate-chain and session-route RSCVs for the requests as they traverse the node.

When Node1 receives a Locate Search request, it initiates a search in the adjacent interconnected network, by forwarding the Locate Search to Node2 (as it would to any other client APPN end node). Node2 checks its subnet list to determine that the search for this resource should be sent to Node3. Node3 removes any network node server entry in the resource hierarchy, removes the RSCV affixed to the Locate Search request and forwards the Locate to Node4 in subnet C (based on Node3's own subnet list). Node4 treats the search like any other request from a client APPN end node and attempts to locate the resource. Since Node4 is acting as the origin network node server (within subnet C), when Node4 receives a positive reply to the search, Node4 calculates a session-route RSCV to the destination and returns this RSCV (in the CD-Initiate) to Node3 (which appears as a client APPN end node to Node4). Node3 saves the session-route RSCV calculated by Node4 and replaces it by TG vector information representing its connection to Node4. Node3 represents itself as the destination network node server and sends the Locate reply on to Node2 which removes the network node server entry in the resource hierarchy, replaces the TG vector with one for the TG to Node1 and returns the Locate reply to Node1. Node1 returns the reply to the origin network node in its own network.

If Node1 receives a Locate Search request from Node2, it conducts a search for the destination LU in its native network (as it would for any other client APPN end node). When the Locate Search reply is received, Node1 acts as the network node server of the origin LU, so Node1 calculates an RSCV to the destination LU using TG vectors from the CP of the destination LU and TG vectors provided by Node2 in the Locate Search request. The Locate Found is returned to Node2. The session RSCV is included in the CD-Initiate.

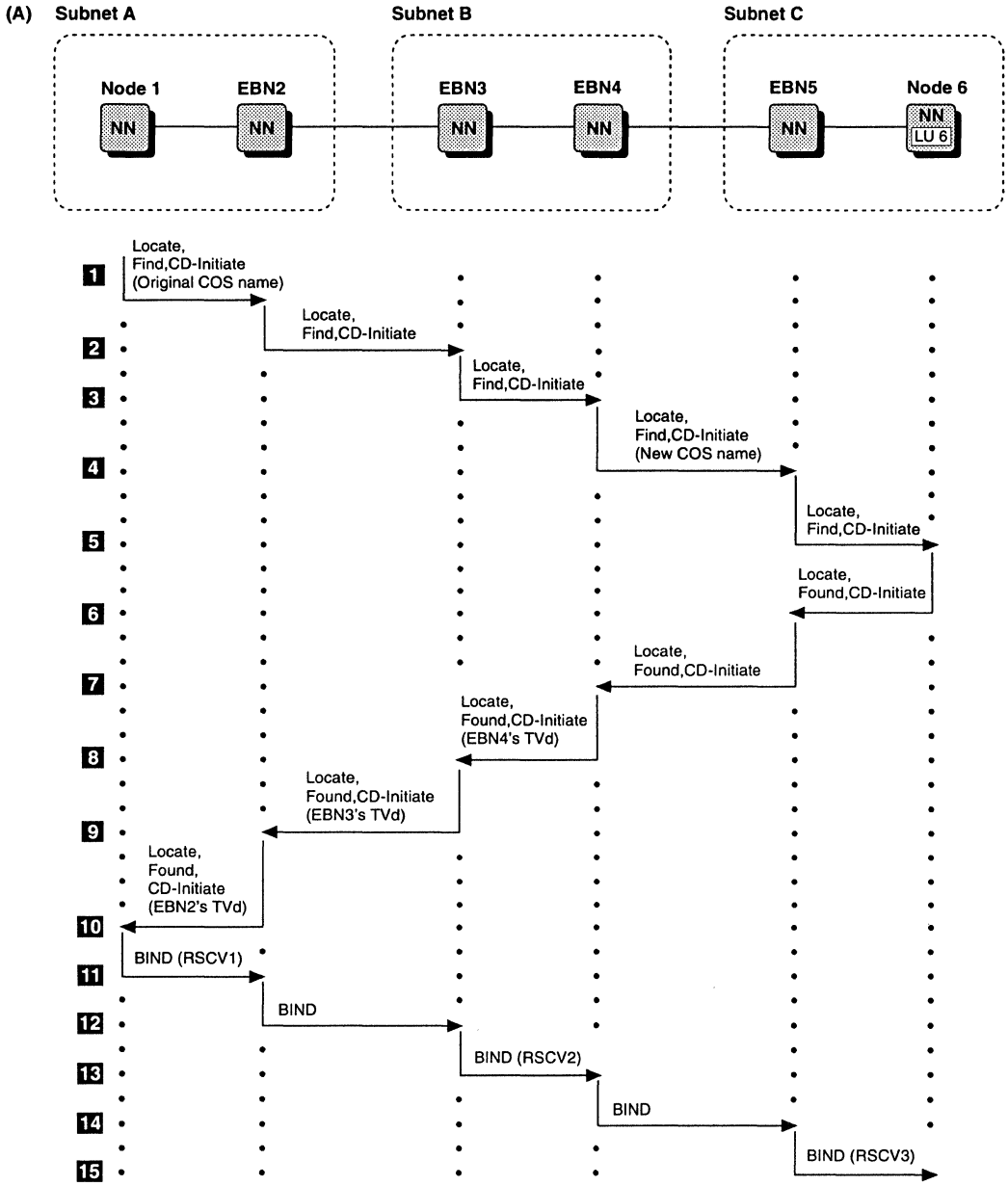
If Node1 receives a BIND request from a node in its native network for an LU in subnet C, it passes the BIND request to Node2 in subnet B. Node2 replaces the RSCV with the session-route RSCV from Node2 to Node3 that was previously calculated during the Locate process. Node2 forwards the BIND request to Node3. Node3 replaces the session-route RSCV in the BIND with the RSCV previously calculated by Node4 acting in its role as the network node server (DLU) when processing the Locate Found. Node3 forwards the BIND request to Node4. Node4 sends the BIND on to the destination, which returns a BIND response back through Node4 to Node3. Session-route RSCVs are replaced in the BIND response in both Node3 and Node2 as the response makes its way back to Node1.

CP Capabilities Exchange in APPN Subnets Containing Extended Border Nodes

An extended border always indicates in the XID exchange that it is a network node with border node capability to *all* native nodes and to nonnative end nodes. An extended border node also indicates it is an APPN end node with extended border node capability to nonnative network nodes and nonnative peripheral border nodes. It indicates that it is a network node with extended border node capability to another extended border node.

Session Initiation Across Interconnected APPN Networks Containing Extended Border Nodes

Figure 82 (A) illustrates the flow sequence involved in a cross-network session initiation in a multisubnet environment containing extended border nodes. Figure 82 (B), (C), and (D) illustrate the way subnet A, subnet B, and subnet C, respectively, view the interconnected subnets.



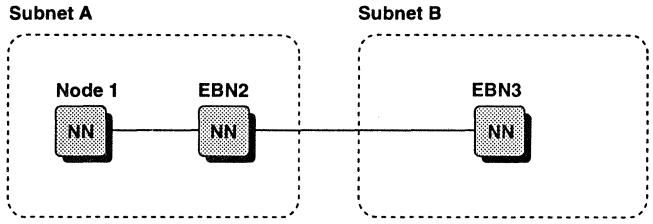
Legend:

Locate	= Locate GDS variable	TVd	= Destination tail vectors
Find	= Find GDS variable	EBN	= Extended border node
Found	= Found GDS variable	NN	= Network node
CD-Initiate	= CD-Initiate GDS variable	EN	= End node
RSCV	= Route selection control vector	LU	= Logical unit

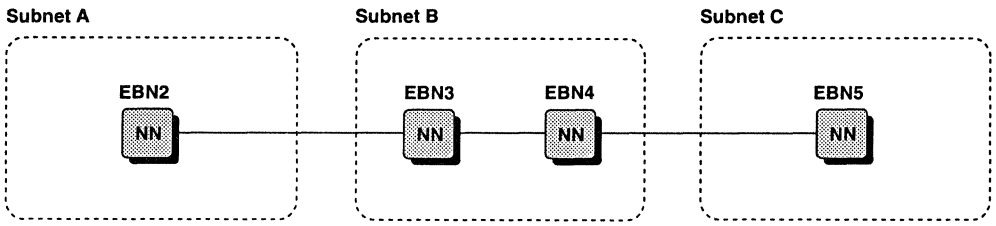
Note: The Locate GDS variable includes the locate-chain RSCV for a directed search (as opposed to a broadcast search).

Figure 82 (Part 1 of 2). Session Initiated from Subnet A to Subnet C

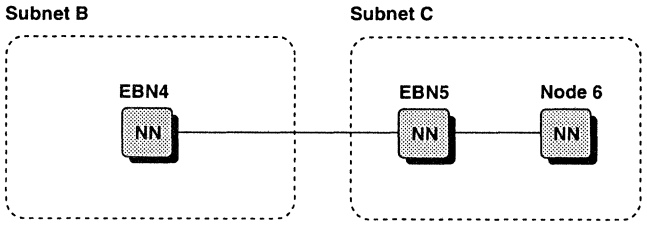
(B) View from Subnet A:



(C) View from Subnet B:



(D) View from Subnet C:



Legend:

Locate	= Locate GDS variable	TVd	= Destination tail vectors
Find	= Find GDS variable	EBN	= Extended border node
Found	= Found GDS variable	NN	= Network node
CD-Initiate	= CD-Initiate GDS variable	EN	= End node
RSCV	= Route selection control vector	LU	= Logical unit

Note: The Locate GDS variable includes the locate-chain RSCV for a directed search (as opposed to a broadcast search).

Figure 82 (Part 2 of 2). Session Initiated from Subnet A to Subnet C

The following comments correspond to the numbers in Figure 82 (A):

1 Node1 sends a Locate/Find (directed or broadcast) to EBN2 to search for *C.LU6*.

2 EBN2 strips off the original locate-chain RSCV before forwarding the Locate/Find to EBN3. This prevents EBN3 from becoming aware of subnet A's topology. EBN2 forwards the Locate/Find to EBN3 in subnet B, since EBN3 is indicated in EBN2's installation-defined or dynamically built "subnet list" as a node on the path of a search for resources with net ID=C.

3 EBN3 forwards the Locate/Find to EBN4 in subnet B, since EBN4 is indicated in EBN3's subnet list as a node that leads to resources with net ID=C. Session-route RSCV2, with the COS, is calculated between EBN3 and EBN4 and is saved by EBN3 for use in future BINDs.

4 EBN4 must verify the COS for this route before forwarding the search. It translates the COS name to an appropriate value for this subnet in CD-Initiate and replaces the locate-chain RSCV before sending the Locate/Find to EBN5. This prevents EBN5 from becoming aware of topology in subnet B. EBN4 forwards the Locate/Find to EBN5 in subnet C, since EBN5 is indicated in EBN4's subnet list as a node that leads to resources with net ID=C.

5 EBN5's subnet list indicates that LU6 might be in its local subnet (subnet C). The Locate/Find is forwarded to Node6 as a broadcast search or directed search depending on whether the location of LU6 has been cached.

6 Node6 returns a Locate/Found to EBN5 to indicate that the destination LU (LU6) has been found. Node6 caches the origin LU location for use in future searches.

7 EBN5 verifies a COS-acceptable route to Node6. Session-route RSCV3, with the COS, is calculated between EBN5 and EBN6. EBN5 caches the session-route RSCV3 and forwards the Locate/Found to EBN4. EBN5 identifies itself as the network node server of the destination LU (C.LU6).

8 EBN4 forwards the Locate/Found to EBN3. EBN4 identifies itself as the network node server of the destination LU (C.LU6). TG vectors indicating the route between EBN4 and EBN5 are included in the Locate/Found.

9 EBN3 forwards the Locate/Found to EBN2. EBN3 identifies itself as the network node server of the destination LU (C.LU6). TG vectors indicating the route between EBN3 and EBN4 are included in the Locate/Found.

10 EBN2 forwards the Locate/Found to Node1. EBN2 identifies itself as the network node server of the destination LU (C.LU6). TG vectors indicating the route between EBN2 and EBN3 are included in the Locate/Found.

11 Node1 computes session-route RSCV1 (the route to EBN2) and sends the BIND with the RSCV1 to EBN2.

12 EBN2 intercepts the BIND and removes the session-route RSCV1 from the BIND to prevent EBN3 from becoming aware of topology in subnet A. EBN2 forwards the BIND to EBN3 in subnet B.

13 EBN3 makes the appropriate COS name substitution in the BIND. EBN3 places session-route RSCV2 to EBN4 (which was saved in step 3) in the BIND. EBN3 forwards the BIND to EBN4.

14 EBN4 intercepts the BIND and removes the session-route RSCV2 from the BIND to prevent EBN5 from becoming aware of topology in subnet B. EBN4 forwards the BIND to EBN5.

15 EBN5 makes the appropriate COS name substitution in the BIND. EBN5 places session-route RSCV3 to Node6 (which was saved in step 7) in the BIND. EBN5 forwards the BIND to Node6.

Routing Data in the Network

Once a route for a session is selected, and the session is activated, data can begin to flow on the session. During data transfer, each intermediate node along the route determines the next designated TG on the route, and the transmission priority at which to transmit the data. Intermediate nodes make these determinations differently in subarea and APPN networks. In a subarea network, the explicit and virtual routes for a session are statically defined, and the session transmission priority is conveyed to path control in the FID4 transmission header. In an APPN network, however, both the transmission group sequence and transmission priority for a session are carried in the BIND request. The routing information is held by the nodes along the route only as long as the session is active.

Routing in Subarea Networks

A subarea node performs an ***intermediate routing function*** when it routes a message unit between two other subarea nodes; it performs a ***boundary function*** when it routes a message unit to and from an adjacent peripheral node. Subarea path control uses the subarea address field of the destination network address to identify the destination subarea node; it uses the element address field to identify the destination NAU within a subarea. If the destination NAU is in the subarea node itself, the message unit is routed to a local half-session. If the destination NAU is in an adjacent peripheral node, the message unit is routed to a boundary function session connector.

Routing in the Boundary Function Component

The boundary function acts as an intermediary between subarea and peripheral path control elements in a subarea node. It contains a session-connector manager and one session connector for each active session traversing the node between subarea and peripheral nodes. (See the generic session-interconnection function structure in Figure 19 on page 32.) Each session connector routes traffic for the session between the subarea and peripheral path control elements, and controls congestion by pacing the session traffic.

When a boundary function session connector receives a message unit from a subarea path control element, the message unit is flowing from the subarea node to a peripheral node. It then routes the message unit to a peripheral path control element in the subarea node. The peripheral path control element in the subarea node then uses the appropriate local address or local-form session identifier (LFSID) to route the message unit over a transmission group to a peripheral path control element in the peripheral node.

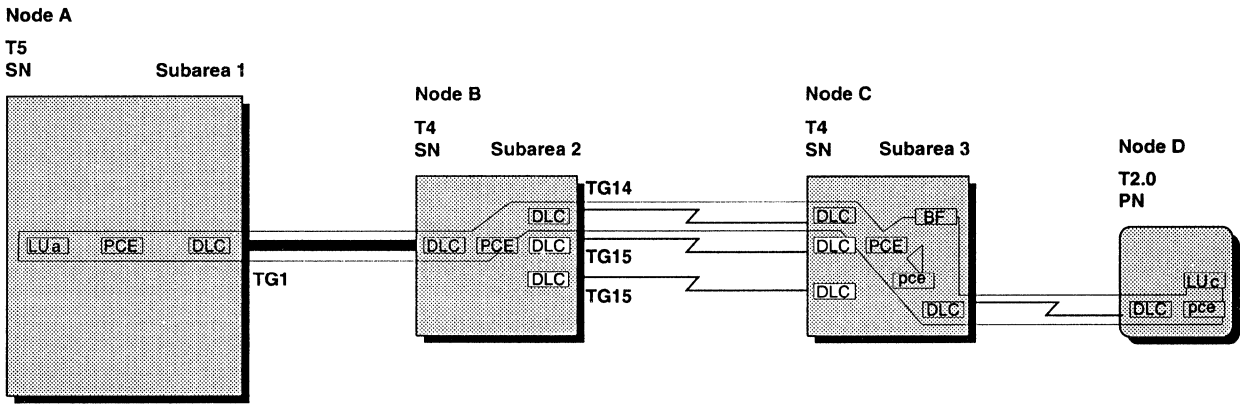
When a boundary function session connector receives a message unit from a peripheral path control element, the message unit is flowing from the peripheral node to the subarea node. It then routes the message unit to a subarea path control element. The subarea path control element then uses the appropriate network address to route the message unit through the network.

Routing Tables in Subarea Networks

An intermediate subarea node need not be aware of all nodes and transmission groups in a path. Instead of an entire path being defined to each node along the path, path definitions are distributed among the nodes along the path. This simplifies system definition and saves storage space in the individual nodes. Therefore, each node contains only a part of the path specification.

Path control routes data on a node-by-node basis. For each node only the information that path control needs for routing data to the next subarea node along the path is defined. This information is defined in the form of ***routing tables***. Path control in each node uses the explicit routes listed in its routing table to determine where to route the data next.

For example, consider the explicit routes (having numbers 0 and 1 in the forward direction, and numbers 2 and 1 in the reverse direction) in Figure 83. The figure shows the *simplified* routing table segments for each node that would be associated with the explicit routes.



Legend:

BF = Boundary Function
DLC = Data Link Control
LU = Logical Unit
pce = Peripheral Path Control Element
PCE = Subarea Path Control Element
TG = Transmission Group

Note: ER0 is the explicit route indicated by the shading between Nodes A and C. Link d is a peripheral link completing the path to peripheral Node D.

Routing Table Location	Destination Subarea	Explicit Route Number	Next Subarea	Transmission Group to Next Subarea
Node A	3 3	0 1	2 2	1 1
Node B	1 1 3 3	2 1 0 1	1 1 3 3	1 1 14 15
Node C	1 1	2 1	2 2	14 15

Figure 83. Routing Table Segments for Two Explicit Routes

Using Routing Tables to Route Data

Using the routing table segments shown in Figure 83, the routing of a message unit from LU *a* to LU *d*, over explicit route 0, would be:

Node A

1. Subarea path control uses the destination network address that the message unit contains to identify node C as the destination subarea node.
2. The node A routing table indicates that transmission group 1 should be used to route the message unit to the next subarea node along the path.

   Path control gives the message unit to data link control for transmission over transmission group 1.

Node B

1. Data link control gives the message unit that it received over transmission group 1 to its subarea path control.
2. The node B routing table indicates that transmission group 14 should be used to route the message unit to the next subarea node along the path.

   Subarea path control gives the message unit to data link control for transmission over transmission group 14.

Node C

1. Data link control gives the message unit that it received over transmission group 14 to its subarea path control.

2. Subarea path control uses the destination subarea address to confirm that it is the destination subarea node along the path.

   Subarea path control uses the boundary function component to pair the destination network address with the local address of LU *d* in node D.

3. Subarea path control routes the message unit to the peripheral path control element in its node.

4. Using a supplemental table called an *element routing table*, the peripheral path control element determines that data link control should transmit the message unit over peripheral link *d* to node D.

Note that no single node knows about all of the subarea nodes and transmission groups that either explicit route 0 or 1 defines. All that a particular node knows about each explicit route in a given direction is the destination subarea node, the next subarea node along the route, and the transmission group that it uses to reach that node.

Routing in SNI Gateways

As discussed in "Defining Gateways in Subarea Networks" in Chapter 4, "Defining Network Resources," the SNA network interconnection (SNI) gateway routes between two or more interconnected subarea networks. The gateway contains a gateway node, and one or more gateway SSCPs. In order to perform its routing function, the gateway must translate the names and addresses used by the sending network into the names and addresses used by the receiving network.

During cross-network LU-LU session initiation, gateway SSCPs prepare the gateway node to perform network name and address translation for the session. To prepare for address translation, a gateway SSCP first sends a Request Network Address Assignment request to the PU in the gateway node. A ***Request Network Address Assignment*** (RNAA) request provides the gateway node with the real network address of the origin LU and requests the gateway node to assign alias addresses for both the origin and destination LUs. If there is more than one gateway SSCP in the gateway, it is the gateway SSCP in the destination network that sends the RNAA request. The gateway SSCP then sends a ***Set Control Vector*** (SETCV) request to the gateway node providing the gateway node with the real address of the destination LU.

To prepare for name translation, a gateway SSCP sends a SETCV request to the gateway node providing the gateway node with the real and alias names of both the origin and destination LUs. If there is more than one gateway SSCP in the gateway, it is the gateway SSCP in the origin network that sends the SETCV request.

During cross-network LU-LU session activation, the gateway node translates the LU names carried in the BIND request received from the origin LU. The gateway node translates the real network name of the origin LU, which is used by the origin network, into the alias name of the origin LU, which is used by the destination network. It also translates the alias name of the destination LU, which is used by the origin network, into the real network name of the destination LU, which is used by the destination network. During session activation, the gateway node also assigns a session to an appropriate virtual route in accordance with the class of service specified in the BIND request. In each interconnected network, the virtual route to which the session is assigned terminates at the gateway node; the two virtual routes are concatenated at the gateway node.

A gateway node contains one path control element for each interconnected network. The ***gateway function*** acts as an intermediary between gateway path control elements. The gateway function contains a session-connector manager and one session connector for each active session traversing the gateway. (See the generic session-interconnection function structure in Figure 19 on page 32.) Each session connector translates network names and addresses for a session, and routes traffic for the session between the path control elements for each network. Congestion in the gateway node is controlled on a virtual route basis, not a session-by-session basis.

Message units transmitted on the session contain the network addresses of both the sending and receiving LUs. The gateway node translates the real network address of the sending LU, which is used by the sending network, into the alias address of the sending LU, which used by the receiving network. It also translates the alias address of the receiving LU, which is used by the sending network, into the real network address of the receiving LU, which used by the receiving network. For a detailed description of a cross-network LU-LU session initiation, see "Initiating Cross-Network LU-LU Sessions" in Chapter 8, "Transporting Data Through the Network."

Routing in APPN Networks

Like routing in subarea networks, routing in APPN networks uses both information carried in the transmission header (TH) of the message unit and information stored at the intermediate node. The two methodologies differ, however, in terms of the nature of the information carried in the TH and stored at the node. Routing information in a subarea network is route-oriented. The TH carries destination and route identifiers that are compared to those in the routing tables. By contrast, routing information in APPN networks is generally session-oriented. The TH carries session identifiers that are temporarily associated with transmission groups for the duration of the session.

Local-Form Session Identifiers

A ***local-form session identifier*** (LFSID) is a 17-bit value that uniquely identifies a session on a transmission group (TG). Because multiple sessions can concurrently traverse a TG, each session stage on the TG must be uniquely identified. In an APPN network, a ***session stage*** is a part of a session that extends from a half-session or session connector in one node to the corresponding half-session or session connector in an adjacent node. A session can therefore be thought of as a sequence of session stages, between adjacent nodes, extending from the origin node to the destination node. For a given session, the origin and destination nodes each control one session stage; each intermediate node controls two.

The LFSID consists of a 1-bit origin-destination assignor indicator and a 16-bit session identifier. The ***origin-destination assignor indicator*** (ODAI) bit identifies which node on a stage assigned the LFSID. The LFSID for a session stage between two nodes is assigned during session activation. As the BIND for a session flows across a TG, the address space manager (ASM) in the CP of the *sending* node assigns the LFSID, and the ASM in the CP of the *receiving* node accepts the LFSID as its own identifier for the session. But because BINDs can flow in both directions on a TG, a mechanism is required to prevent the two sides of a TG from assigning the same identifying value for two different sessions.

The mechanism that prevents ambiguity between session identifiers on a TG is the usage of the ODAI bit. During TG activation, adjacent APPN nodes connected by the TG negotiate through XID exchange the value that the ODAI bit will be set to for all sessions they initiate on that TG. One node is assigned the value 0, and the other node is assigned the value 1. Thereafter, for as long as the TG is active, the two sides assign LFSIDs with the ODAI bit set according to the value negotiated during link activation. If the BIND requests flowing from one node to the other have the ODAI set to 0, then all BIND requests flowing from the other node to the first have the ODAI bit set to 1. This prevents session stages for two concurrent sessions over the same TG from ever being assigned identical LFSIDs.

The Intermediate Session Routing Component

The component in an APPN network node that is responsible for routing between session stages is the ***intermediate session routing (ISR) component***. The ISR component contains a session-connector manager and one session connector for each active session traversing the ISR component. (See the generic session-interconnection function structure in Figure 19 on page 32.) Each session connector routes traffic for the session between path control elements in the network node, and controls congestion by pacing session traffic.

When a message unit is received by a session connector, the transmission header (TH) of the message unit carries the LFSID of the session stage over which it was just received. During session activation, the session connector manager associates the LFSID of the inbound session stage to the LFSID of the outbound stage, and stores the association as part of the session connector's linkage to the appropriate path control elements used for the interconnected session stages. Then, after the session is activated, the session connector assigned to the session transfers session traffic between the two path control elements, effectively causing proper swapping of the LFSIDs used in the TH on the two session stages.

Building ISR Session Connectors

When a BIND request is created, an LFSID for the first session stage is assigned at the origin node and passed along with the BIND. When the BIND is received by an intermediate network node, the LFSID is extracted from the TH received with the BIND and saved. A new LFSID is then assigned for the outbound session stage, and replaces the old LFSID in the TH forwarded with the BIND. When the destination node receives the BIND request, it extracts the LFSID from the BIND TH and saves it. At this point all LFSIDs for the session have been assigned, but the session connectors will not be completely initialized until the BIND response flows back through them.

When the BIND response is sent out from the destination node, the same LFSID that was received by it is placed in the BIND response TH. When the response is received by an intermediate network node, it swaps the LFSID in the message for the one pertaining to the next session stage on the return route, completes session connector initialization, and sends the BIND response on its way. When the origin node receives the BIND response, all session connectors for the session have been initialized and are ready to route data. Figure 84 illustrates the swapping of LFSIDs.

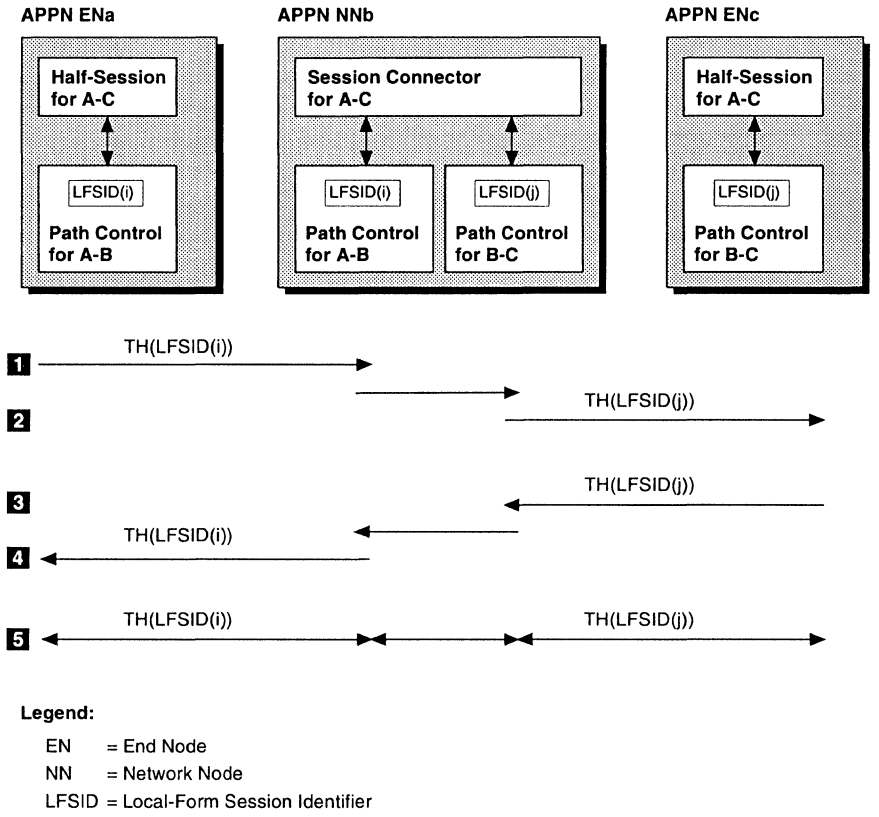


Figure 84. BIND Sets Up LFSID Swapping

The following comments correspond to the numbers in Figure 84:

1 The half-session in ENa is to be connected with the half-session in ENc to activate a session between LUa and LUc. A BIND goes from ENa to NNb, carrying a TH that contains an LFSID selected in ENa. In NNb, the BIND invokes upper-layer management components (address space manager (ASM) and session connector manager (SCM) and creates entries in the newly activated session connector and in both path control components (one entry for the incoming TG and one entry for the outgoing TG). NNb created a new LSFID(j) for the session stage to ENc.

2 The BIND continues to ENc, but with new address fields, representing LFSID(j), in the TH.

3 ENc accepts the BIND and returns a positive response. The LSFID values used in the TH are reversed for the return at each session stage.

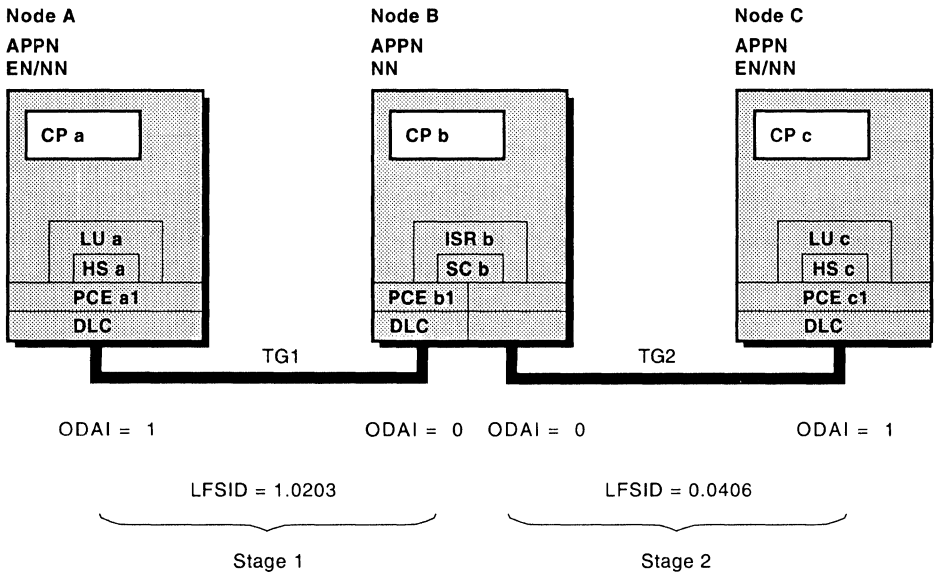
4 The response continues to ENa with swapped address values in the TH.

5 Now the rest of the PIUs on the session can flow through NNb without rising above the session connector layer. The session PIUs pass through the session connector layer for the pacing function and to switch path control components. The addresses in the THs are swapped as noted in accordance with the information stored at BIND time.

Half-session and session-connector initialization requires the setting of pacing window sizes for congestion control. The ***pacing window size*** in an APPN network is the number of consecutive messages that can be sent in one direction between adjacent nodes on a session stage before the receiving node acknowledges the messages. Adjacent nodes set pacing window sizes on each session

stage independently. On a given session stage, the initial pacing window sizes are negotiated through the exchange of the BIND request and response. Both the BIND request and response must be transmitted successfully before pacing window sizes are initialized in each half-session and session connector along the route. Session pacing is discussed further in Chapter 7, "Controlling Congestion in the Network."

Figure 85 on page 173 illustrates the assignment of LFSIDs and the building of session connectors and half-sessions for a session between two LUs. As shown in the diagram, the ODAI bit settings negotiated during activation of TG 1 were 1 for node A and 0 for node B. Likewise, the ODAI bit settings negotiated during activation of TG 2 were 0 for node B and 1 for node C.



Legend:

CP = Control Point
DLC = Data Link Control
HS = Half-Session
ISR = Intermediate Session Routing Component
LFSID = Local-Form Session Identifier
LU = Logical Unit
ODAI = Origin-Destination Assignor Indicator
PCE = Path Control Element
SC = Session Connector
TG = Transmission Group

Figure 85. LFSIDs and Session Connectors

Node A

1. LU *a*:
   a. Requests a PCID from CP *a*'s session services.
   b. Receives a PCID from CP *a*, creates an FQPCID, and sends a session-initiation request (for LU c) to CP *a*'s session services.
   c. Receives a Control Initiate from CP *a* (following a successful search for LU c) and requests an LFSID from CP *a*'s address space manager. Because the BIND request will originate at node A, the ODAI bit in the LFSID is set to 1.
   d. Begins initializing a half-session for the new session.
   e. Creates a BIND request containing the LFSID, FQPCID, and initial pacing window size, and sends it to CP *a*.
2. CP *a*'s address space manager routes the BIND request to PCE *a1*. PCE *a1* controls the first TG on the route as identified by the RSCV attached to the BIND request.
3. PCE *a1*:
   a. Saves the LFSID received from CP *a*.
   b. Transmits the BIND request on TG 1.

Node B

1. PCE *b1*:
   a. Receives the BIND request, extracts the LFSID, and saves it.
   b. Sends the BIND request to CP *b*.
2. CP *b*'s address space manager:
   a. Checks to see if the destination LU is local or remote.
   b. Because the destination LU is remote, CP *b* routes the BIND request to ISR *b*.
3. ISR *b*'s session connector manager:
   a. Begins initializing a session connector, and stores the LFSID and node A's initial pacing window size for session stage 1.
   b. Sends a session-initiation request to CP *b*.
   c. Receives a Control Initiate from CP *b*'s session services and requests an LFSID for the second session stage from CP *b*.
   d. Obtains an LFSID from CP *b*'s address space manager for session stage 2. Because the BIND request on stage 2 will originate at node B, the ODAI bit in the LFSID is set to 0.
   e. Places the new LFSID and the initial pacing window size for stage 2 in the BIND request, and sends it to CP *b*.
4. CP *b*'s address space manager routes the BIND request to PCE *b2*. PCE *b2* controls the second TG on the route as identified by the RSCV attached to the BIND request.
5. PCE *b2*:
   a. Saves the LFSID received from CP *b*.
   b. Transmits the BIND request on TG 2.

Node C

1. PCE *c1*:
   a. Receives the BIND request and stores the LFSID for the session stage 2.
   b. Sends the BIND request to CP *c*.
2. CP *c*'s address space manager:
   a. Checks to see if the destination LU is local or remote.
   b. Because the destination LU is local, CP *c* routes the BIND request to LU *c*.
3. LU *c*'s session manager:
   a. Creates a half-session for the new session.
   b. Stores node B's initial pacing window size for stage 2.
   c. Creates a BIND response containing the same LFSID as was received in the request, and its own initial pacing window size for stage 2.
   d. Sends the BIND response to CP *c*'s address space manager.
4. CP *c* routes the BIND response to PCE *c1*.
5. PCE *c1* transmits the BIND response on TG 2.

Node B

1. PCE *b2*:
   a. Receives the BIND response and validates the LFSID for session stage 2.
   b. Sends the BIND response to CP *b*.
2. CP *b*'s address space manager routes the BIND response to ISR *b*.
3. ISR *b*'s session connector manager:
   a. Completes initializing the session connector and stores node C's initial pacing window size for stage 2.
   b. Replaces the LFSID in the BIND response with the LFSID for session stage 1, and sets its own initial pacing window size for stage 1 in the BIND response.
   c. Sends the BIND response back to CP *b*.
4. CP *b*'s address space manager routes the BIND response to PCE *b1*.
5. PCE *b1* transmits the BIND response on TG 1.

Node A

1. PCE *a1*:
   a. Receives the BIND response and validates the LFSID for session stage 1.
   b. Sends the BIND response to CP *a*.
2. CP *a*'s address space manager routes the BIND response to LU *a*.
3. LU *a*'s session manager completes initializing the half-session and stores node B's initial pacing window size for stage 1. User data can now be routed on the session.

Routing in an APPN Node

When user data begins flowing on a session, the LFSIDs serve the function of identifying to a path control element (PCE) for which session a message unit is defined. The PCE can then determine to which half-session or session connector the message unit should be routed. Similarly, the PCE determines which LFSID information applies to the outgoing TH based on the source half-session or session connector. At an intermediate network node, the LFSID corresponding to the inbound session stage is replaced in the TH by the LFSID corresponding to the outbound session stage. Refer to Figure 85, together with the following narrative, for an illustration of how a message is routed on a session between two LUs.

Node A

1. LU *a*:
   a. Receives data from an end user.
   b. Creates a basic information unit (BIU).
   c. Sends the BIU on the appropriate half-session to PCE *a1*.
2. PCE *a1*:
   a. Creates a PIU from the BIU, and places the LFSID for session stage 1 in the transmission header (TH).
   b. Transmits the message unit on TG 1.

Node B

1. PCE *b1*:
   a. Receives the PIU and extracts the LFSID from the TH.
   b. Removes the TH from the PIU, thus re-creating the BIU.
   c. Using the LFSID, determines which session connector to route the BIU to.
   d. Routes the BIU to the appropriate session connector.
2. The session connector:
   a. Routes the BIU to PCE *b2* (assuming it is not awaiting a pacing response).
3. PCE *b2*:
   a. Creates a PIU from the BIU, and places the appropriate LFSID for session stage 2 in the TH.
   b. Transmits the message unit on TG 2.

Node C

1. PCE *c1*:
   a. Receives the PIU and extracts the LFSID from the TH.
   b. Removes the TH, thus re-creating the BIU.
   c. Using the LFSID, determines which half-session to route the BIU to.
   d. Routes the BIU to the appropriate half-session in LU *c*.
2. LU *c*:
   a. Receives the BIU from the half-session.
   b. Removes the RH from the BIU.
   c. Passes the data to the end user.

Functions of the Session Connector Manager (SCM)

The ***session connector manager (SCM)*** is responsible for session-activation and session-deactivation in intermediate network nodes. The SCM receives session-activation and session-deactivation requests and responses via the address space manager (ASM) from one path control and forwards them through ASM to another path control.

When the ASM of an APPN network node receives a BIND, it determines whether the secondary LU (SLU) resides within the local node. If the SLU is not local, ASM assumes that its node is to be an ***intermediate node*** for the session being established and passes the BIND to the SCM for processing. ASM also passes any associated RSP(BIND), UNBIND, and RSP(UNBIND) to the SCM for processing.

Session Activation: SCM's role in session activation includes the following:

- Keeps track of the maximum number of sessions that can concurrently use the ISR function at the local node.
- Transforms BINDs and RSP(BINDs) from extended to nonextended and vice versa; also, shortens BIND when BIND reassembly is not supported.

  Nodes supporting LEN protocols can send nonextended BIND and RSP(BINDs); nodes sending nonextended BINDs receive unextended RSP(BINDs). Nodes supporting LEN protocols can receive extended BIND (and ignore unrecognized fields and control vectors). SCM transforms LU

6.2 BINDs and RSP(BINDs) in order to support the attachment of nodes supporting LEN protocols to its APPN network.

- Negotiates RU sizes.

  A PLU specifies in the BIND both a PLU maximum send RU size and an SLU maximum send RU size. The SLU can negotiate these maximum RU sizes by modifying their values in the RSP(BIND) if the BIND is negotiable. SCM participates in maximum RU size negotiation for negotiable BINDs.

  SCM can have upper bounds on the maximum acceptable RU sizes, defined by the network operator. When SCM receives a BIND, it can reduce the specified maximum RU sizes based on the upper bounds before forwarding the BIND.

  SCM can also reduce the specified SLU maximum send RU size if the BIND indicates that the adjacent node on the primary stage does not support segment reassembly.

  SCM verifies the SLU maximum send RU size.

  SCM can reduce the PLU maximum send RU size before forwarding the RSP(BIND) if the size in the RSP(BIND) is larger than the upper bound defined by the network operator, or if the RSP(BIND) indicates that the adjacent node on the secondary stage does not support segment reassembly.

- Sets up session-level pacing for the session stages between its local node and the adjacent nodes.

- Performs session correlation.

  When an extended BIND is received, SCM uses the FQPCID in the BIND. When an nonextended BIND is received, SCM requests that session services assign an FQPCID. When a RSP(BIND) is received, it is correlated with the session in one of the following ways:

  - Matching the FQPCID in the RSP(BIND) in the extended case
  - Finding the local session record that includes the path control ID and matching LFSID for the secondary stage in the nonextended case.

- Reserves resources.

  When a BIND is received and successfully processed, the SCM reserves storage for the message units that will be sent.

  When the RSP(BIND) is received and successfully processed, the SCM reserves storage for receiving session data over both the primary and secondary stages before forwarding the RSP(BIND).

  If there is insufficient storage for the session, the session is deactivated.

Session Deactivation: SCM's role in session-deactivation includes the following:

- UNBIND and -RSP(BIND) processing.

  SCM participates in the deactivation of an LU-LU session when it receives an UNBIND or -RSP(BIND) for the session. In either case, SCM performs the necessary transform and forwards the UNBIND or -RSP(BIND) over the opposite session stage. (The transform consists of SCM changing an nonextended UNBIND to an extended UNBIND or converting a -RSP(BIND) into an extended UNBIND when the original BIND received over the primary stage was extended.)

- Error processing.

  SCM initiates session deactivation when it detects an error condition that prevents the continuation of the session. When an error is detected, SCM generates and sends UNBINDs over the primary and secondary stages to notify the other associated session-level components that the session is being deactivated.

Intermediate Session Routing for Dependent LU Sessions

Non-LU 6.2 sessions may be routed through an APPN network node. All of the previous SCM functions are applicable to ISR in non-LU 6.2 sessions with the following exceptions:

- No BIND and RSP(BIND) Extension

  SCM rejects an nonextended non-LU 6.2 BIND or RSP(BIND). Any node participating in the activation of a non-LU 6.2 session traversing an APPN network must extend the BIND or RSP(BIND) before sending it to the network.

- Maximum RU Sizes

  Nonnegotiable BINDs:

  Session activation is unsuccessful when the received BIND for a non-LU 6.2 session is nonnegotiable and either the PLU maximum send RU size or the maximum send RU size is larger than either its network operator-defined upper bound or the maximum send BTU size for the appropriate stage, minus 9, when whole BIUs are required on that stage.

  Negotiable BINDs:

  When no maximum size is specified in a negotiable BIND, SCM processes the BIND as if it had the maximum specifiable size. Otherwise, BIND negotiation is performed.

  Pacing Fields in Nonnegotiable BINDs

  SCM forwards a nonnegotiable BIND over the secondary stage without changing the pacing window sizes; SCM forwards a nonnegotiable +RSP(BIND) over the primary stage after setting the pacing window sizes to their values in the received BIND.

  When SCM receives a nonnegotiable BIND that indicates adaptive pacing is not supported and specifies a secondary receive window size that is larger than the maximum fixed receive window size defined by the network operator for the primary stage, the BIND is rejected.

  When SCM receives a nonnegotiable BIND that indicates adaptive pacing is not supported and specifies a secondary receive window size that is larger than the maximum fixed receive window size defined by NOF for the secondary stage, the session is deactivated.

  When a nonnegotiable BIND requests no pacing in the primary-to-secondary direction or a nonnegotiable +RSP(BIND) requests no pacing in the secondary-to-primary direction, SCM deactivates the session if the node does not support a receive pacing type of "none." Otherwise, SCM uses a NOF-defined pool size to create a nonreserved permanent buffer pool for that stage.

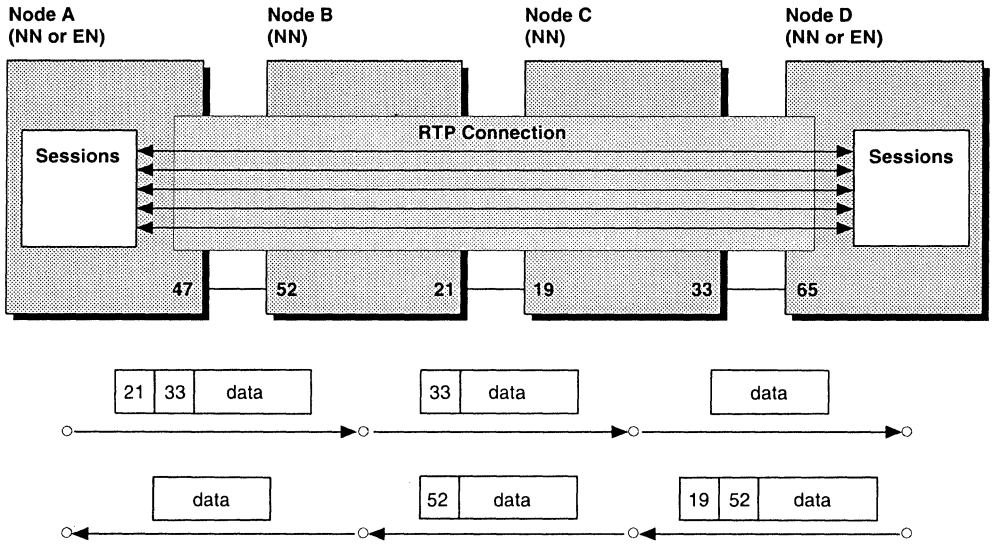
Automatic Network Routing (ANR) in HPR

Automatic Network Routing (ANR) is a low-level routing mechanism that minimizes cycles and storage requirements for routing packets through intermediate nodes. ANR represents significant increases in routing speed over basic APPN. ANR provides point-to-point transport between any two endpoints in the network. Intermediate nodes are not aware of SNA sessions or RTP connections passing through the node. ANR is designed for high-performance switching, since no intermediate node storage for routing tables is required and no precommitted buffers are necessary. When an RTP connection fails, a new RTP connection is activated over a different physical path. SNA sessions using this RTP connection are not interrupted and session data flows are automatically switched to the new RTP connection with no time-out or loss of data (nondisruptive path switch).

The originator of the packet explicitly defines in the routing field of the packet's network layer header the exact path on which the packet is to flow through the network; thus, ANR is a special implementation of a source-routing protocol. Each network layer packet (NLP) is routed through the network as a self-contained unit and is independent of all other packets. There is no table lookup or processing necessary at transit nodes such as the LFSID swapping procedure used by ISR.

HPR employs a route setup protocol in order to obtain ANR and RTP connection information about the selected path. Any processing of packets required at the network connection and transport connection sublayers is the responsibility of the origin and destination endpoints of the packets. Endpoint processing includes flow control, segmentation and reassembly, and recovery of lost packets.

A packet using ANR contains in the network layer header (NHDR) an ANR routing field composed of a sequence of ANR labels (ALs) that identify the transmission groups. Figure 86 illustrates automatic network routing.



Legend:

NN = Network node
EN = End node
RTP = Rapid-transport protocol
Data = Transport header and data

Figure 86. Automatic Network Routing

The routing information for every node on the path is contained in the packet header. As an NLP flows through a network, the ANR labels are stripped from the packet's ANR routing field as they are used by each node. Each node finds its routing information at a fixed placement in the header. Therefore, when a packet is forwarded across a transmission link, the first label in the routing field always indicates the physical link to be used by the next node.

Each TG connecting two nodes or subnodes is assigned two ANR labels, one at each end. When the TG is activated, each node assigns an ANR for the outbound direction. For example, in Figure 86, Node A assigns the label 47 for the direction from Node A to Node B; Node B assigns label 52 for the direction from Node B to Node A.

Deactivating Routes

In a subarea network, access methods and network control programs in subarea nodes deactivate explicit and virtual routes as needed. The network operator never enters commands to deactivate routes. Path control deactivates a route when the last session using it is deactivated. Path control deactivates explicit routes automatically when a PU or link along the path is deactivated or becomes inoperative.

In an APPN network, a session route interconnects the half-sessions and session connectors extending from the origin LU to the destination LU. A session route is deactivated when the session for which it was calculated is deactivated. Session deactivation begins when the session manager of one partner LU sends an UNBIND request to its local CP. The address space managers (ASMs) in all CPs along the route then route the UNBIND, just as the BIND was routed during session activation.

In processing the UNBIND request, the origin and destination LUs destroy their half-sessions for the session, and the session connector managers in the intermediate network nodes destroy their session connectors for the session. In addition, the PCE in the origin node handling the first session stage tells the local ASM to release the LFSID for the session. In processing the UNBIND response, all PCEs handling the outbound message unit likewise tell their local ASMs to release the LFSIDs for the session stages along the route. when the UNBIND response returns to the origin node, the route for the session has been dismantled.

Chapter 7. Controlling Congestion in the Network

This chapter explains the roles of transport network transmission protocols and flow control algorithms in moving data efficiently through the network.

The Need for Congestion Control

When the rate at which messages enter a network exceeds the rate at which they can be transmitted, congestion results. The rate at which data can be transmitted is called ***throughput***. When doing capacity planning, systems personnel can plan for normal peak traffic periods, but unexpected traffic demands or disruptions in component service levels can decrease network throughput.

When congestion occurs, response times lengthen and buffer resources are depleted. Severe or prolonged congestion in one part of the network affects other parts of the network, decreasing the network's efficiency, and overall throughput can actually decrease. When message buffers are full, new messages cannot be accepted. Messages might therefore have to be queued for long periods before finally reaching their destinations.

The transport of data through a network can be affected by many factors. For example, the number of end users sending and receiving data through a network changes over time. If the network attempts to handle more end users than it has capacity for, congestion occurs in nodes having the least throughput capacity. Some nodes are capable of processing a greater amount of data than others. For example, without congestion control, a logical unit in a T5 node might transmit data to a logical unit in a peripheral node faster than the peripheral logical unit could accept and process the data. Buffers in a T4 node interconnected between the peripheral node and the T5 node would begin to fill up because requests could not be delivered to the peripheral logical unit as quickly as they were being received from the T5 node.

Another factor affecting network throughput is network component failures. When a link fails, the traffic that it normally carries must be diverted over other links. An already active link might not be able to carry the additional traffic load, and the overloaded link may cause congestion in a node to which it is attached.

In an APPN network, any time the state of a node changes from "congested" to "not congested," or vice versa, a TDU is broadcast to all other network nodes. A node becomes congested when an upper fraction (90%) of the defined maximum number of intermediate sessions allowed for a node has been reached. A node is considered no longer congested when the number of intermediate session drops below a lower fraction (80%) of the maximum number of intermediate sessions allowed.

Approaches to Congestion Control

To optimize network throughput, SNA employs two categories of performance-related protocols: message repackaging and message pacing. ***Message repackaging*** is the combining of messages into larger units, or the subdividing of messages into smaller units, for transmission. Changing the sizes of message units improves the efficiency with which nodes or links can process them. ***Message pacing*** addresses congestion by controlling the rate at which messages are sent into a network. Both subarea and APPN networks use message repackaging and message pacing for congestion control.

Message Repackaging

Messages entering a network can be repackaged numerous times before arriving at their destinations. The first point at which a message can be repackaged is during the process of request/response unit (RU) creation. Subsequently, message units can be repackaged by path control protocols in order to accommodate the varying capacities of the nodes and links through which the message units must travel. The transport network uses two repackaging protocols: blocking and segmenting.

RU Creation

Messages submitted by an end user are repackaged by the sending LU into request/response units. An LU can package a message within one RU, subdivide it and transmit the parts within multiple RUs, or combine multiple messages into a single RU. The choice depends upon the maximum allowable RU size, which is determined by session start-up parameters.

The maximum RU size determines the size of the buffers used by presentation services to hold data being sent to, and received from, transaction programs, and by the half-session for data being sent to the partner LU. Unless the transaction program issues a verb that explicitly flushes the send buffer (such as FLUSH), presentation services (PS) waits until a buffer (typically set equal to the maximum RU size) is full before transferring it to the half-session. Similarly, PS holds an RU received from the partner LU in a receive buffer and dispenses it on demand and in the amount requested by the transaction program. The transaction program's interface with PS enables it to specify different amounts of data that the transaction program wishes to receive, such as a fixed number of bytes, a user-defined logical record, or one or more buffers.

The maximum RU size can be tuned for improved efficiency. For an end user that sends multiple small records into the network, for example, overhead can be reduced by grouping a number of the records into a single RU. To optimize an end user's storage allocation, the sizes of RUs sent to the end user can be matched to the end user's buffer sizes. Similarly, when segmenting is not supported by a lower-layer network component, PS can adjust the RU size to meet size restrictions on message units transmitted to that component.

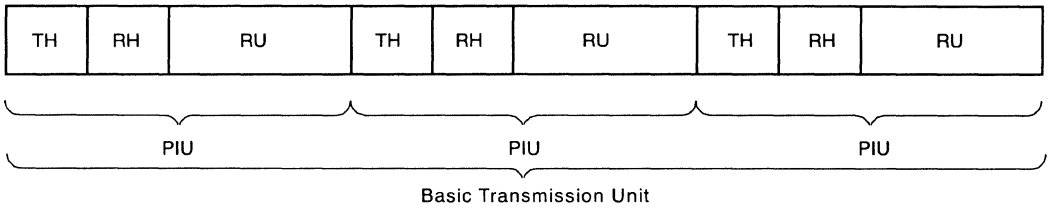
Blocking

Blocking is the combining of multiple PIUs into one basic transmission unit. A ***basic transmission unit*** (BTU) is a generic term for the information that data link control transmits over a link. A BTU can consist of one PIU or multiple PIUs. Recall from Chapter 3, "Data Formats" that a PIU framed by a link header and link trailer is called a ***basic link unit***. When PIUs are blocked, and framed by a link header and link trailer, the resulting message unit is also called a basic link unit.

Path control uses blocking protocols to combine multiple PIUs before giving the message units to data link control. Data link control then transmits one BTU (containing multiple PIUs) over a link. A maximum BTU length (the number of PIUs) for the network is specified during system definition. Once blocked message units are transmitted over the link, the original message units are separated.

Blocking increases the amount of information that data link control can transmit over a link at one time, thereby taking advantage of links with large transmission capacities. Path control currently implements blocking only on System/370 data

channels. Figure 87 shows a basic transmission unit that contains more than one PIU.



Legend:
PIU = Path Information Unit
RH = Request/Response Header
RU = Request/Response Unit
TH = Transmission Header

Figure 87. Blocking of Path Information Units

Segmentation

Segmentation is the division of one BIU into multiple BIU segments, each of which is contained in a separate PIU. Segmenting is an optional feature that is not supported in all nodes. Path control uses segmenting protocols to divide a single BIU into multiple BIU segments, then packages the segments in PIUs before giving them to data link control. Data link control then transmits each PIU (one BTU) separately over a link. A mapping field in the transmission header indicates whether the BTU contains the first, a middle, or the last segment of the BIU.

Segmentation accommodates links with large error rates or nodes with limited buffer space. Path control segments BIUs on a session basis between nodes so that the maximum BTU size allowed by the receiving node is not exceeded. Path control segments BIUs on virtual routes except for those BIUs destined for half-sessions attached to boundary functions; whole BIUs are sent to the boundary function, which may segment them to the peripheral node. In subarea and T2.0 nodes, BIU reassembly is performed by path control. In LEN or APPN nodes, reassembly of session BIUs is performed by a half-session or, in the case of an APPN intermediate routing node, a session-connector; reassembly of segmented session-activation and -deactivation BIUs is performed by the address space manager (ASM) component in the receiving node's control point.

Figure 88 on page 186 shows a BIU divided into BIU segments. Each BTU contains a transmission header and a segment of the BIU. A transmission header accompanies each BIU segment because it contains routing information that the receiving path control requires.

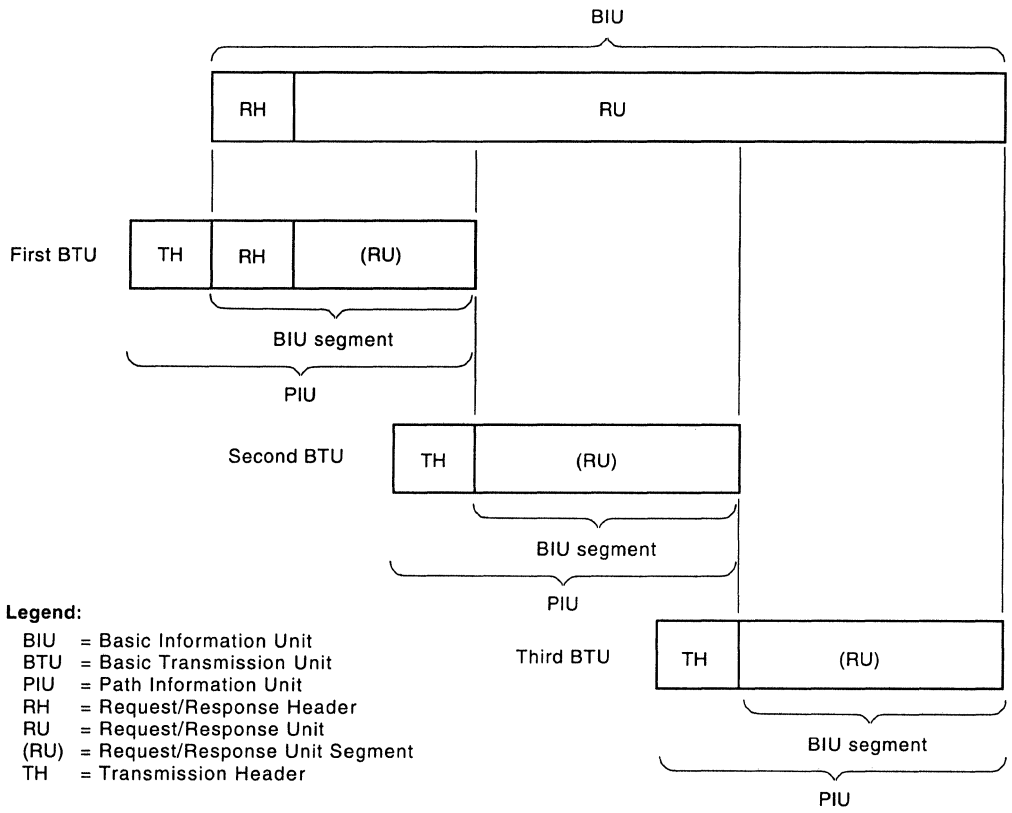


Figure 88. Segmenting of Basic Information Units

In HPR, PIUs are divided into PIU segments. Figure 89 on page 186 shows a PIU segmented into several segments that are transported in NLPs.

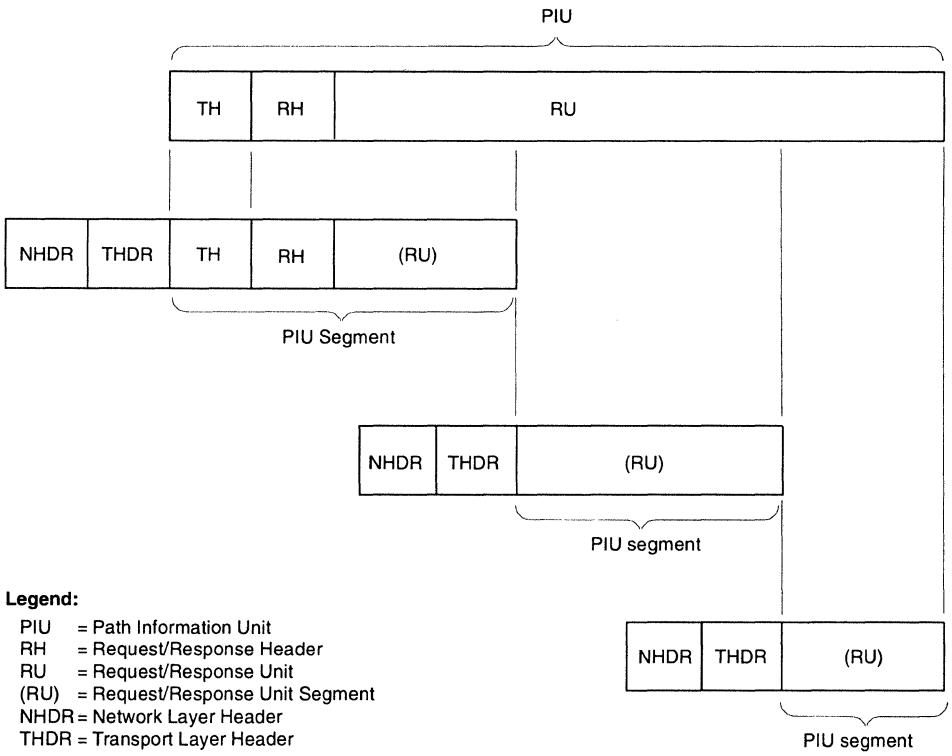


Figure 89. Segmenting of Path Information Unit in HPR

BIND Segmentation/Reassembly

APPN network nodes always support segmentation and reassembly of BIND requests and responses. APPN end node support of this function is optional. In an APPN end node that does not support segmentation and reassembly of BINDS, the address space manager (ASM) accepts only whole BIND requests and responses. It discards a segmented BIND request or response, and in the case of a request, returns a -RSP(BIND). In either case, ASM instructs configuration services to deactivate the TG. Because BIND message units are relatively lengthy, BIND segmentation is a matter of importance for nodes with low buffer capacities. A BIND message unit is segmented only if its length exceeds the maximum allowable BTU size on the link over which it is to be transmitted.

As with segmentation of other BIUs, path control segments the BIND message units and makes use of the mapping field in the transmission header of the PIU. BIND segmentation differs, however, in that the address space manager (ASM) in the control point of the adjacent node performs the BIU reassembly rather than a half-session or session connector. In LEN or APPN nodes, the ASM routes BIND messages between path control and logical units or (in intermediate routing nodes) intermediate session routing (ISR) components.

Message Pacing

Pacing is the control of the rate at which data enters and traverses a network. Pacing is based on pacing window techniques. A ***pacing window*** is the group of message units that a network component can send to another network component at one time. The ***pacing window size*** is the number of message units in the pacing window. Pacing permits only a certain number of sequential messages to be outstanding in the network between the origin and destination pacing partners before an acknowledgment is received. This acknowledgment indicates that the destination pacing partner is ready to accept an additional pacing window.

With message pacing, the destination component is responsible for managing its buffers, and it therefore controls the pacing window size. The first message unit in a pacing window contains a ***pacing request***, which solicits approval from the destination pacing partner for sending another window. Sometime after receiving a pacing request, the destination partner sends back a pacing response. A ***pacing response*** is a positive acknowledgment that indicates the destination's ability to receive an additional pacing window. While transmitting message units, the origin pacing partner keeps a running count of the number of message units that are left to send in a pacing window. The residual number of message units is called the ***pacing count***. The destination pacing partner withholds the pacing response until it is ready to receive an additional pacing window.

Several kinds of pacing techniques are used in SNA networks. These include virtual-route pacing, fixed session-level pacing, adaptive session-level pacing, and BIND pacing.

For detailed information on pacing protocols, refer to *SNA Format and Protocol Reference Manual: Architectural Logic* and *SNA LU 6.2 Reference—Peer Protocols*.

Virtual-Route Pacing

In the subarea routing network, ***virtual-route pacing*** operates on the group of sessions sharing a particular virtual route (VR). It enables an intermediate subarea node along a VR to inject window-size reduction requests into transmission headers passing through to the ends of the VR in order to prevent depletion of its buffers. Thus, a subarea node can influence the pacing parameters of a virtual route even though it is not aware of the individual sessions assigned to the VR.

Virtual-route pacing is implemented in subarea nodes at the path control layer. On a network-wide basis, subarea nodes continually monitor the amount of congestion in the network. If congestion occurs, these nodes limit the amount of data they send over virtual routes. Virtual-route pacing is automatic, requiring no action by end users or network operators.

Activation of a virtual route includes initializing the pacing window size to indicate the number of message units that a sender can initially transmit over a virtual route before a pacing response is received. The pacing window size for a virtual route fluctuates between a minimum (initial) value and a maximum value depending on the perceived need to use the transport network and the severity of network congestion. Subarea nodes along the route determine the severity of network congestion by monitoring the number of messages in their queues.

A virtual route can be paced in both directions. The pacing of requests flowing toward a subarea node is called ***inbound pacing*** for that node. Similarly, the pacing of requests flowing away from the node is called ***outbound pacing***.

Virtual-route pacing *adapts* to the level of congestion on a virtual route. SNA defines two degrees of congestion on virtual routes. Within the first, less severe level, the number of message units sent in each consecutive window is decreased by 1 until congestion clears. An intermediate node along the route signals its detection of this level of congestion by setting a ***change window indicator*** in the header of a message unit flowing toward the *destination* of the congesting traffic. The destination endpoint then decreases the window size in its next pacing response by 1, to accommodate either the received change window indicator or its current local congestion status.

Within the second, more severe level of congestion, the window size is immediately decreased to its minimum value. If the current (residual) pacing count exceeds the minimum window size, then the pacing count is also set to the minimum value. This level of congestion can be signaled by an endpoint or any intermediate node by setting the ***reset window indicator*** in the header of a message unit flowing toward the *source* of congesting traffic. A receiving endpoint node that detects such a severe level of congestion from some source may also withhold its next pacing response to that source until it frees the necessary buffer resources to do so. The sending source subarea node must, in turn, avoid severely depleting its own buffers by accepting too many message units that need to be sent over the congested virtual route. Congestion parameters for access methods and network control programs control the amount of buffer depletion by limiting the number of additional message units the sending subarea receives.

When the congestion clears, the destination endpoint can signal in its pacing response that the pacing window can be increased. The sending endpoint then increases the window size in each subsequent pacing request by 1 until the maximum size is reached.

Minimum and maximum window values can be specified by flow-control-threshold parameters during system generation, or by access-method exit routines. The minimum window size default is equal to the number of transmission groups that make up the explicit route that underlies the virtual route. The maximum window size default is equal to three times the minimum window value.

Gateway nodes always terminate virtual routes. Virtual routes between one network and a gateway node are paced independently of virtual routes between the gateway node and an interconnected network. Gateway nodes, like other subarea nodes, use virtual-route pacing to limit the amount of data they send and receive over virtual routes.

Session-Level Pacing

Session-level pacing involves each session-layer component in a session path. It is used by transmission control on LU-LU, SSCP-SSCP, and CP-CP sessions. The session path consists of one or more ***session stages***, where half-sessions at the origin and destination nodes and session connectors at intermediate nodes delimit the stages. A session connector can be a boundary function or gateway function component in a subarea node, or an intermediate session routing component in an APPN network node. Each stage is independently paced. Session-level pacing enables each session connector to participate in pacing based on awareness of its local node's congestion conditions.

Two communicating network accessible units frequently have an inherent saturation rate; at some point, incoming traffic exceeds a NAU's processing and buffer storage capacity. For this reason, each sender has a maximum number of message units that it can send before receiving a go-ahead pacing response from the receiver.

Session-level pacing is selected when a session is activated. The sending NAU initiates pacing by sending a pacing request in the first message unit of a pacing window. While transmitting message units, the sender maintains the pacing count of the number of message units that are left to send in the current pacing window. After having received the pacing request, the receiver returns a pacing response indicating that it is ready to accept an additional window. If it delays returning the pacing response, the receiver indicates that it is not ready to accept the next window. Upon receiving a pacing response, the sender first completes sending the number of message units remaining in the residual pacing count. It then sends the next pacing window.

Like a virtual route, a session can be paced in each direction independently. The window sizes do not have to be the same for the two directions. The maximum window size is 32,767.

Fixed Session-Level Pacing

Fixed session-level pacing requires a fixed window size. At session-activation, the fixed window size is carried in the BIND request. When a pacing response is received by a sending NAU, the sender increases its pacing count by the fixed window size.

With fixed session-level pacing, a pacing response can be returned to the sender in either of two ways: in the response header of a message unit, or in a stand-alone message called an ***isolated pacing response*** (IPR). Only one IPR can be sent for a pacing request. The IPR is needed for returning a pacing response when there is no regular response to be returned to the sending NAU.

Adaptive Session-Level Pacing

As with virtual-route pacing, ***adaptive session-level pacing*** adapts to the level of congestion by allowing a variable window size. Whereas VR pacing changes the window size in unit steps (or to the minimum value in the case of severe congestion), adaptive session-level pacing defines a more general variation of window size.

The window sizes are changed by use of isolated pacing messages. An ***isolated pacing message*** (IPM) not only authorizes the sending of the next window of message units, but, like the IPR, also specifies whether the window size is to be increased, decreased, or remain the same. With adaptive session-level pacing, nodes respond to pacing requests exclusively by use of the IPM, not by use of normal-flow message unit headers.

Under normal circumstances, an IPM is sent for the same reason as is an IPR. IPMs transmitted by a receiver to respond to the sender's solicitation for permission to send another window of message units are called ***solicited IPMs***. When a receiving node becomes congested, however, it can send an IPM regardless of whether there is an outstanding pacing request. IPMs transmitted to change the sender's pacing window because of congestion in the receiver are called ***unsolicited IPMs***. Because an unsolicited IPM is a request message unit, not a response, it requires a response. The response to an unsolicited IPM is a ***reset acknowledgment***.

When a data sender receives an unsolicited IPM with a nonzero window size, it completes sending the number of message units in its residual pacing count before changing the pacing window size. When the unsolicited IPM contains a window size of 0, however, that indicates a state of extreme congestion in the receiver. In that case, the sending node sets both its window size and its residual pacing count to 0. Another normal-flow message unit is not sent until the sender receives an IPM containing a nonzero window size.

Session-Level Pacing in Subarea Networks

In subarea networks, session stages are delimited by half-sessions and boundary function session connectors. (In interconnected subarea networks, session stages are also delimited by session connectors in gateway nodes (GWNs), but these do not perform session-level pacing.) Figure 90 illustrates two possible staging configurations within a stand-alone subarea network. ***One-stage pacing*** occurs when subarea LUs have a session with one another. Each LU paces messages received from the other LU on the session. In the one-stage pacing illustration, pacing occurs directly between the two LUs communicating over the session. ***Two-stage pacing*** occurs when a subarea LU has a session with a peripheral LU. The boundary function, as well as the LUs, participates in session-level pacing. In the two-stage pacing illustration, pacing occurs between one of the LUs and the boundary function, then between the boundary function and the other LU.

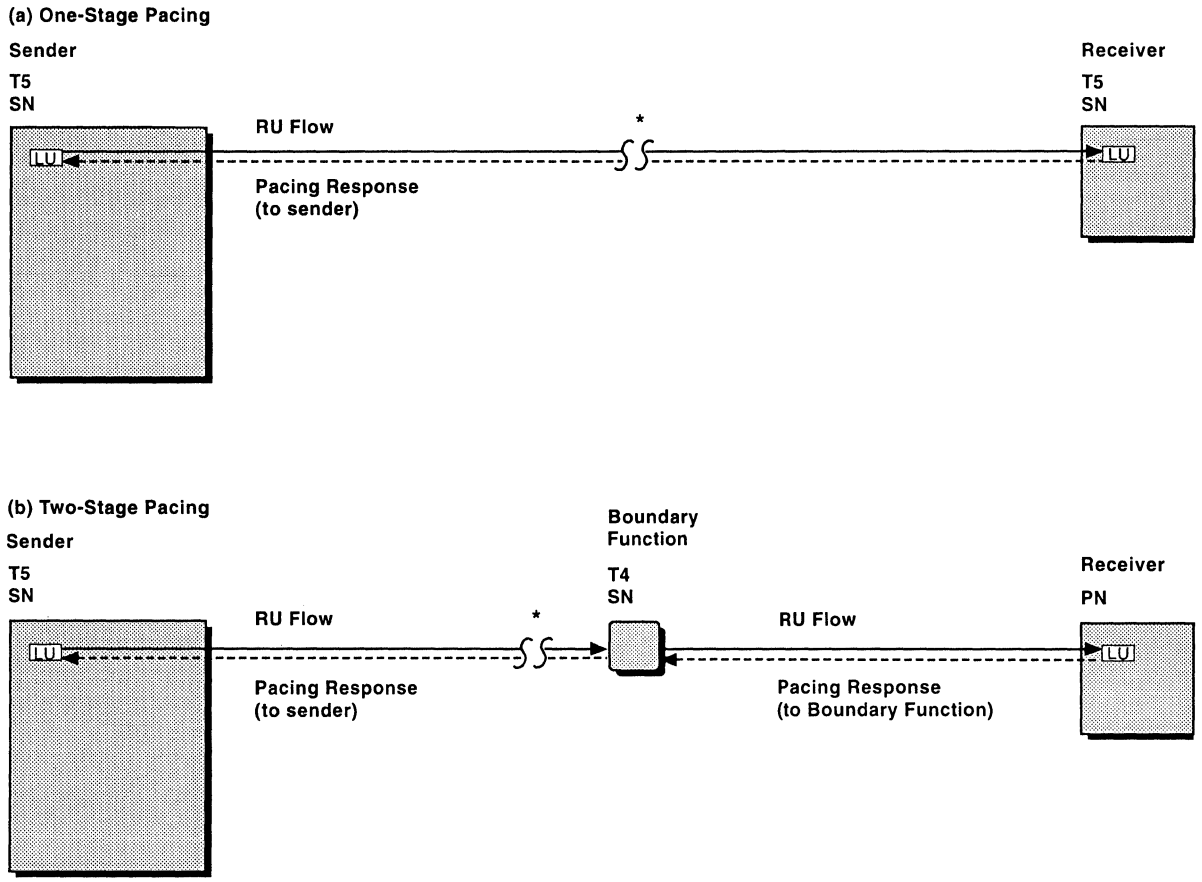
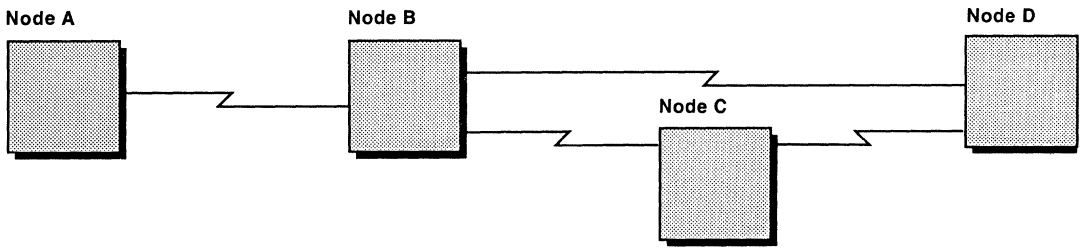


Figure 90. Session-Level Pacing

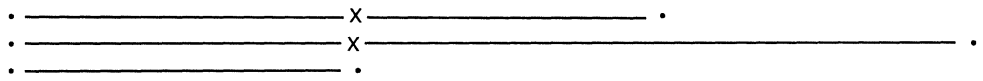
Session-Level Pacing in APPN Networks

In an APPN network, session stages are delimited by half-sessions and intermediate session routing (ISR) component session connectors in each adjacent LEN or APPN node. Through the flow of BIND request and response messages during session activation, adjacent half-session and session-connector components on a session route establish initial window sizes on each session stage independently. Then, as data flows on the session, the half-sessions and session connectors synchronize the rate of data flow over each link to prevent depletion of their buffers.

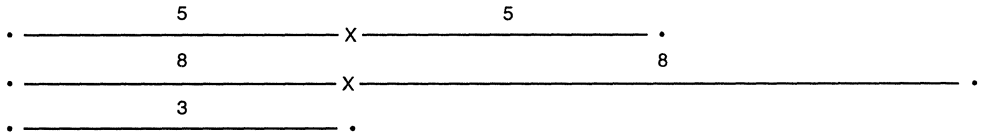
In an APPN network, all nodes in a session path participate in adaptive session-level pacing. Therefore, the double-tier pacing (session-level pacing and VR pacing) of the subarea routing network is not needed. Network-wide flow control is achieved by the combined effect of each node independently controlling congestion in its own buffers. The general lowering of traffic in the network is caused by *back-pressure*, whereby the lowering of pacing counts in one node causes lowering in the adjacent nodes from which data is being received. Figure 91 illustrates the process of back-pressure in lowering network-wide pacing counts in an APPN network.



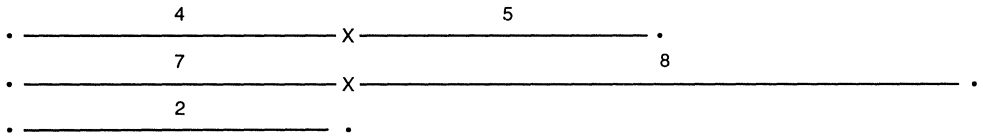
Three sessions exist in the network (X indicates session connector).



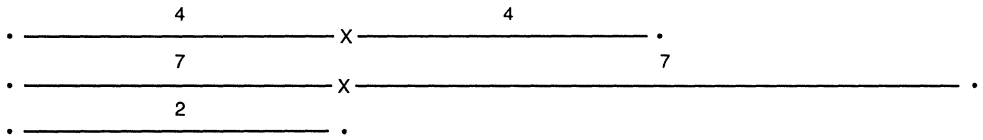
The current pacing counts with node A as receiver are as follows:



Node A becomes congested and reduces its windows.



Node B cannot buffer extra messages, so it reduces its windows.



The result is that all flows in the network are reduced.

Figure 91. Back-Pressure in an APPN Network

BIND Pacing

Congestion can occur when LUs in a network activate a large number of sessions in a short period of time. A condition called BIND standoff can occur when two nodes send a large number of BINDs to each other. ***BIND standoff*** is a condition in which each node allocates all of its resources for outgoing BIND requests and reserves none for servicing incoming BIND requests. Then, even though sufficient resources would be available for handling a slower pace of BIND requests, no sessions are activated because each node rejects the incoming BINDs because of a lack of resources. This problem is addressed in APPN nodes through the use of BIND pacing.

BIND pacing is a special case of adaptive pacing that applies to all BIND requests and responses exchanged between adjacent nodes. Like adaptive session-level pacing, BIND pacing allows message units to be paced at different rates in each direction, and pacing windows are altered dynamically through the use of IPM messages. Unlike adaptive session-level pacing, however, BIND pacing is not performed by half-sessions, because the half-sessions have not yet been initialized.

Instead, it is a function of the address space manager (ASM) in the control point. The ASM can concurrently pace BIND message units representing multiple sessions over the same transmission group. BIND pacing is useful to control heavy BIND traffic at node or network start-up time.

In HPR, the BIND-pacing mechanism is not performed over an RTP connection between two endpoint HPR nodes. This is due to a new flow/congestion control mechanism called *adaptive rate-based (ARB)* flow/congestion control.

Adaptive Rate-Based (ARB) Flow/Congestion Control in HPR Networks

The ***adaptive rate-based (ARB)*** flow/congestion control algorithm is designed to make efficient use of network resources by providing a congestion avoidance and control mechanism. The basic approach used in this algorithm is to regulate the input traffic in the face of changing network conditions. When the algorithm detects that the network is approaching congestion and the path gets saturated, resulting in increased delays and decreased throughput, it reduces the input traffic rate until these indications go away. When the network is sensed to have enough capacity to handle the offered load, the algorithm allows more traffic to enter the network without exceeding the rate the receiver can handle. ARB prevents unnecessary retransmissions, minimizes response time, and improves throughput efficiency. ARB also fosters network stability and consistent, predictable performance.

The ARB algorithm is based on information exchanged between the two endpoints of a connection. This information reflects the state of both the receiver and of the network. The sender, based on the information from the receiver, regulates the input traffic accordingly. The rate-based algorithm is applied independently in each direction, as in adaptive pacing. ARB "smooths" the sending of the available input to accommodate the target rate. The smoothing gives networks better characteristics than if the input is sent in a burst.

In HPR, multiple sessions with the same COS are multiplexed over a single RTP connection. The existing session-level pacing over an RTP connection is used to prevent one session from getting more than its share of buffers in the two RTP components. In the basic APPN context, the RTP connection is seen as one session stage (hop) on the session path. Figure 92 illustrates the difference between basic APPN and HPR in the way session-level pacing is handled.

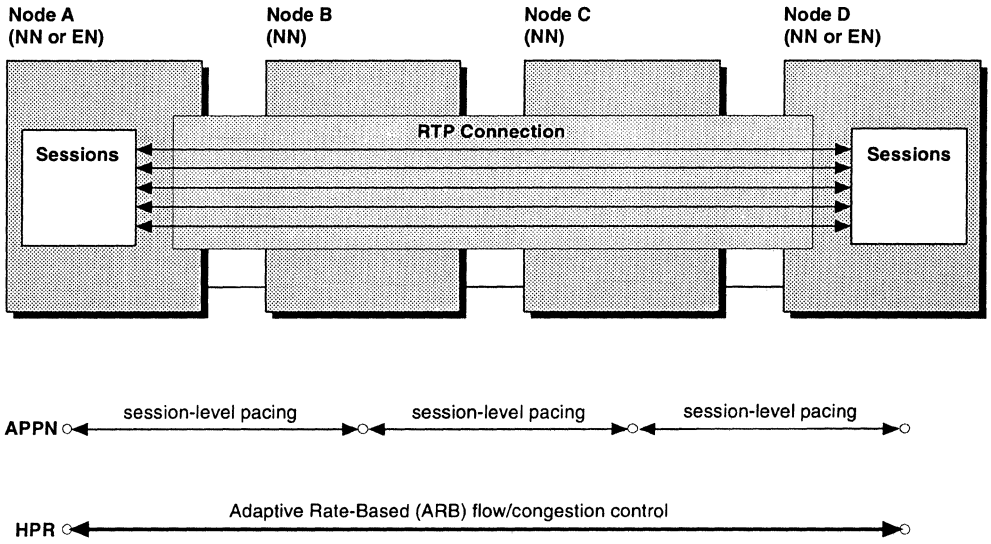


Figure 92. APPN/HPR Session-Level Pacing

The feedback information consists mainly of the minimum of two rates: one is the rate at which the receiver accepts arriving data from the network, and the other is the rate at which the receiver delivers data to the end user. The former indicates the state of the path in the network and is used by the sender to regulate the input traffic load to avoid overloading the path and therefore causing congestion. The latter is used by the sender to avoid overloading the receiver and thus results in end-to-end flow control. Based on the rate information received, the sender can determine when congestion is about to develop in the network and take appropriate actions to avoid it. If congestion does occur, the sender takes drastic measures to bring the network back to normal.

Chapter 8. Transporting Data Through the Network

This chapter discusses session protocols for controlling end-to-end data transport. The subject areas include data flow control protocols, LU-LU session control, security facilities, and protocols specific to LU 6.2.

Initiating LU-LU Sessions

End users (application programs and individuals) gain access to an SNA network through logical units and exchange information over LU-LU sessions. Once network resources are active, LU-LU sessions can be initiated.

LU-LU sessions can be initiated in several ways:

- Either of the participating logical units can initiate an LU-LU session.
- A network operator can initiate an LU-LU session.
- In some cases, a third logical unit can initiate an LU-LU session between two other logical units.
- System definition can specify that an LU-LU session be initiated automatically when certain resources become active.

Typically, one of the participating logical units initiates an LU-LU session. The two logical units that communicate with each other over a session are called ***session partners***. This section discusses the session-activation process when one of the session partners initiates the LU-LU session.

LU-LU session initiation generally begins when the session manager in a logical unit (LU) submits a ***session-initiation request*** to the appropriate control point. In a subarea network, it can be either the system services control point (SSCP) controlling the LU's domain or, in the case of a T2.1 peripheral node, the LU's local control point (CP). In an APPN network, it is always the LU's local CP.

A session-initiation request specifies the requested session partner's network name and a mode name. The ***mode name*** identifies which set of session parameters that the requesting logical unit chooses for the requested session. In a subarea network, the mode name is associated with the parameters through a ***mode table*** created during system definition. In an APPN network, the association is made through the network operator facility (NOF).

Using the specified set of session parameters, the control point builds a ***Bind image***. The control point transmits the Bind image in a ***Control Initiate request*** (CINIT request) to the primary logical unit. The ***primary logical unit*** (PLU) is the LU responsible for activating the session. The PLU ***activates*** the session by sending a ***Bind Session request*** (BIND request) (also called a ***session-activation request***) to the ***secondary logical unit*** (SLU). The SLU then returns a BIND response to the PLU.

Initiating LU-LU Sessions in Subarea Networks

In a subarea network, LU-LU session initiation is mediated by an SSCP, either by explicit request for dependent LUs or implicitly for independent LUs. Multiple SSCPs are involved when an LU-LU session spans multiple domains or networks. The SSCP determines (1) whether the logical units are authorized to communicate with each other, (2) the possible routes between the two logical units, and (3) a set of session protocols for the session. The SSCP obtains the session-initiation information from statically defined tables.

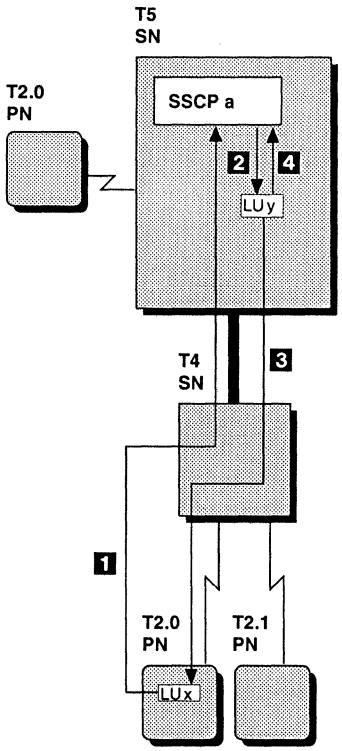
An LU in a peripheral node can be SSCP-dependent or SSCP-independent. The next three sections discuss LU-LU session initiation using dependent LU-LU session protocols. A dependent LU sends its session-initiation request to the

SSCP controlling the domain in which the LU resides. In this case, the LU must have an active session with the SSCP before it can request an LU-LU session. The SSCP then sends a CINIT request containing the necessary session parameters to the primary LU (PLU). The PLU activates the session by constructing a BIND request from the CINIT request and sending it to the secondary LU (SLU).

After a single-domain session is activated using SSCP-dependent protocols, the SSCP is notified by the PLU with a ***Session Started*** (SESSST) request. If the partner LUs reside in different domains, the SSCP in each domain is notified. If the SLU does not agree to the set of session parameters sent by the PLU, it returns a negative BIND response to the PLU, and the PLU sends a ***Bind Session Failure*** (BINDF) request to the SSCP (or SSCPs) instead of the SESSST request. The BINDF request indicates the reason that the LU-LU session was not activated.

Initiating Same-Domain LU-LU Sessions Using SSCP-Dependent Protocols

Figure 93 illustrates session initiation using SSCP-dependent protocols between two LUs in the same domain, LU *x* and LU *y*. When SSCP *a* receives the session-initiation request from LU *x*, it checks its directory and determines that LU *y* is in the same domain. SSCP *a* sends a Control Initiate (CINIT) request to LU *y*. LU *y* then activates the session by sending a BIND request to LU *x* and notifies SSCP *a* of the session activation.



1 Session-Initiation request from LU x to SSCP a (SSCP-LU session)

2 CINIT request from SSCP a to LU y (SSCP-LU session)

3 Session-Activation (BIND) request from LU y to LU x (LU-LU session)

4 Session-Started (SESSST) request from LU y to SSCP a (SSCP-LU session)

Legend:

LU = Logical Unit
SSCP = System Services Control Point

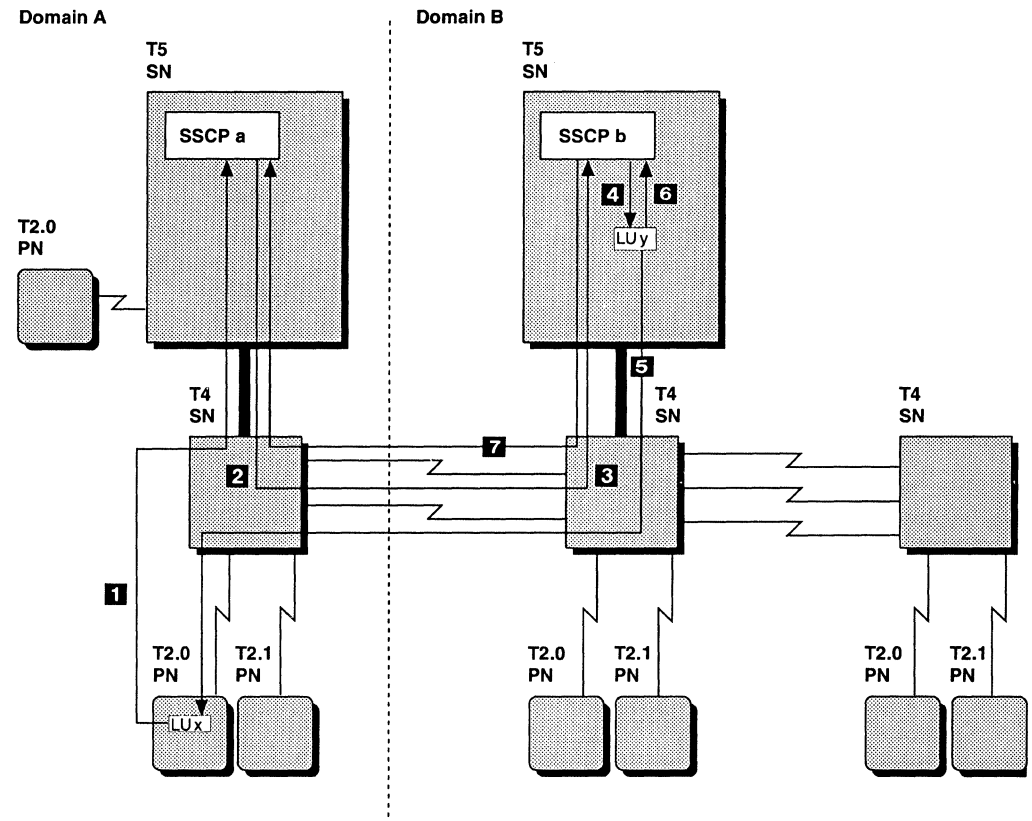
Figure 93. Initiating a Same-Domain LU-LU Session Using SSCP-Dependent Protocols

Initiating Cross-Domain LU-LU Sessions Using SSCP-Dependent Protocols

A ***cross-domain LU-LU session*** connects logical units in different domains. When two dependent logical units in a subarea network reside in different domains, each is under the control of a different SSCP. Because a different SSCP controls each of the logical units, an SSCP-SSCP session is necessary before either logical unit can activate the LU-LU session. The two SSCPs communicate over the SSCP-SSCP session to coordinate the cross-domain session initiation.

Figure 94 on page 200 illustrates the sequence for initiating a cross-domain session between two LUs in a subarea network, LU *x* and LU *y*. When SSCP *a* receives the session-initiation request from LU *x*, it checks its directory and determines that LU *y* resides in another domain, under the control of SSCP *b*. SSCP *a* sends a Cross-Domain Initiate (CDINIT) request to SSCP *b*. The CDINIT request and response identify the requesting LU (LU *x*) and the desired session partner (LU *y*), and they determine which LU will be the primary LU (PLU) on the session. In this sequence, LU *y* is designated the PLU and LU *x* is designated the SLU.

Next, the SSCP of the SLU, SSCP *a*, sends a Cross-Domain Control Initiate (CDCINIT) request to the SSCP of the PLU, SSCP *b*. The CDCINIT request passes information about LU *x* to SSCP *b* and requests that SSCP *b* send a Control Initiate (CINIT) to LU *y*. LU *y* then activates the session by sending a BIND request to LU *x*. Neither LU *x* nor LU *y* is aware that the LU-LU session crosses domain boundaries.



Legend:

LU = Logical Unit
SSCP = System Services Control Point

1. Session-Initiation request from LU x to SSCP a (SSCP-LU session)
2. CDINIT request from SSCP a to SSCP b (SSCP-SSCP session)
3. CDCINT request from SSCP a to SSCP b (SSCP-SSCP session)
4. CINIT request from SSCP b to LU y (SSCP-LU session)
5. Session-Activation (BIND) request from LU y to LU x (LU-LU session)
6. Session Started (SESSST) request from LU y to SSCP b (SSCP-LU session)
7. Cross-Domain Session Started (CDSESSST) request from SSCP b to SSCP a (SSCP-SSCP session)

Figure 94. Initiating a Cross-Domain LU-LU Session Using SSCP-Dependent Protocols

Initiating Cross-Network LU-LU Sessions

A ***cross-network LU-LU session*** connects logical units in different networks. In a subarea network, these sessions traverse one or more gateways[12] and are limited to SSCP-dependent protocols. A cross-network SSCP-SSCP session connects a gateway SSCP in one network to an SSCP in an interconnected network (the second SSCP may or may not be a gateway SSCP). Cross-network SSCP-SSCP sessions support the SSCP-mediation of cross-network LU-LU session initiation.

[12] Recall that a gateway consists of a gateway node and one or more gateway SSCPs.

Figure 95 on page 202 and Figure 96 on page 203 illustrate the sequence for initiating a cross-network session between LU *x* and LU *z*. The gateway consists of a gateway node and two gateway SSCPs. The gateway SSCPs send and receive requests and responses over an SSCP-SSCP session. The transfer of LU-LU messages by SSCPs between interconnected subarea networks is called ***SSCP rerouting***. To LU *x*, LU *z* appears to be in the domain of SSCP *a*, and to LU *z*, LU *x* appears to be in the domain of SSCP *b*. Neither LU *x* nor LU *z* is aware that the gateway SSCPs are rerouting their messages.

A cross-network LU-LU session-initiation sequence can be thought of as two series of messages. The first series establishes real and alias names and addresses that enable the gateway to perform the name and address *transforms*. The second series then activates the session.

Figure 95 illustrates the message exchanges needed to establish real and alias names and addresses in the gateway. When gateway SSCP *a* receives the session-initiation request from LU *x*, it checks its directory and determines that LU *z* resides in network B, under the control of SSCP *b*. SSCP *a* then sends a CDINIT request to SSCP *b* identifying the two session partners. Before doing so, however, SSCP *a* transforms network A's alias name for LU *z* into a real network-qualified name. The CDINIT request contains the network-qualified names of the origin LU, LU *x*; the destination LU, LU *z*; and LU *x*'s real network address.

SSCP *b* then sends a Request Network Address Assignment (RNAA) request to the PU in the gateway node, PU *c*, containing LU *x*'s real network address and requesting that PU c assign alias addresses for the two partner LUs. PU *c* stores LU *x*'s real network address and assigns the alias addresses. PU *c* now has three of the four addresses needed to perform the gateway address transform function. The fourth address, LU *z*'s real network address, is provided to PU *c* by SSCP *b* in the Set Control Vector (SETCV) request.

SSCP *b* then sends a CDINIT response to SSCP *a* containing LU *x*'s alias name in network B. This enables SSCP *a* to send a SETCV request to PU *c* containing the real and alias names for the partner LUs. PU *c* will need this information to perform the LU name transform for BIND messages exchanged during session activation. SSCP *a* also resolves the class-of-service name that was sent by LU *x* in the session-initiation request into a virtual route (VR) list, and sends the list in the SETCV request as well. SSCP *a* then sends a response to LU *x*'s session-initiation request back to LU *x*.

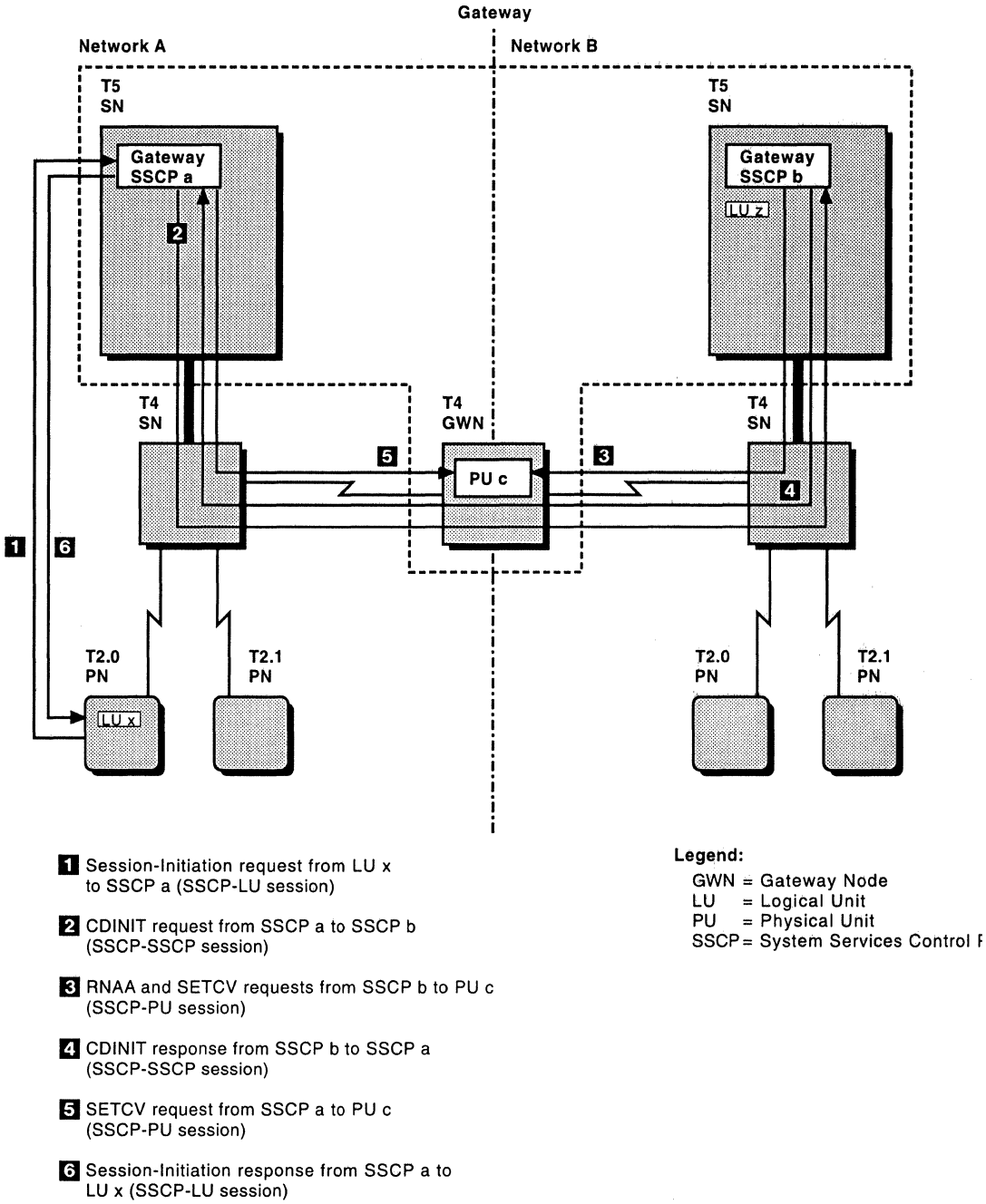
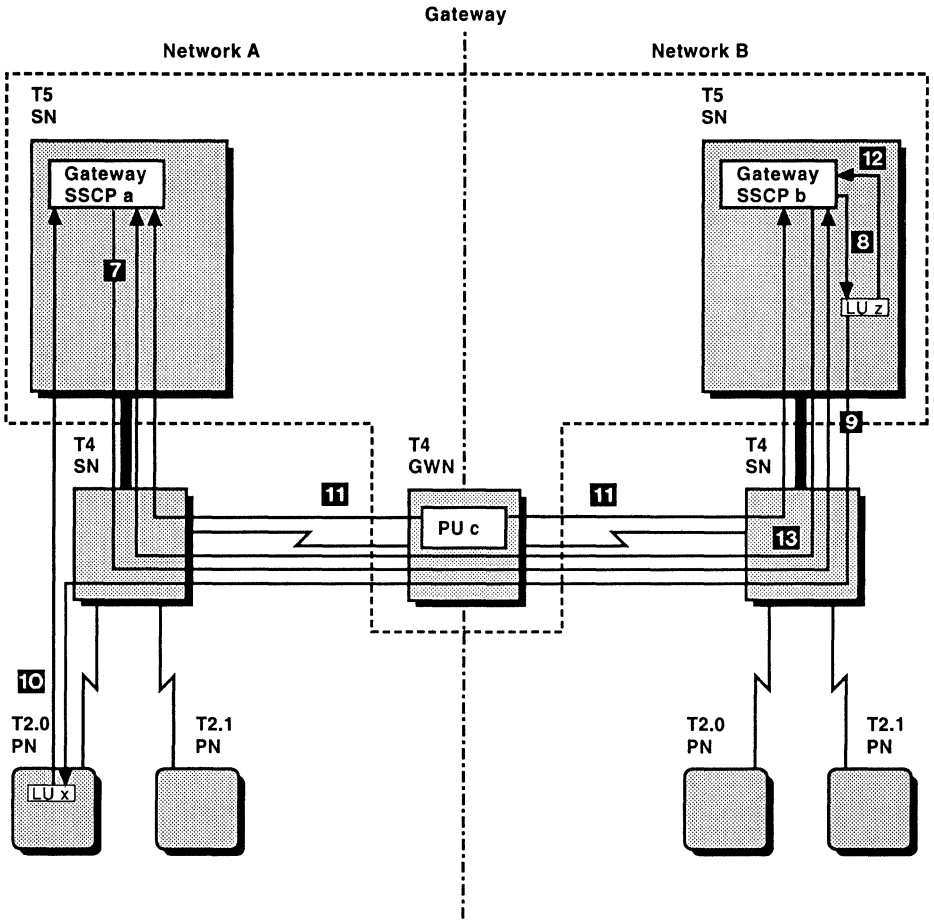


Figure 95. Initiating a Cross-Network LU-LU Session: Part I

Figure 96 illustrates the message exchanges for *activating* the cross-network LU-LU session. First, the SSCP for the secondary LU (SLU), SSCP *a*, sends a CDCINIT request to the SSCP for the primary LU (PLU), SSCP *b*, containing the Bind image for the session. As the request passes through the gateway node, the gateway function performs name and address transforms on the request. SSCP *b* then sends the Bind image to LU *z* in a CINIT request. To LU *z*, LU *x* is identified by its alias name.

The PLU, LU *z*, then sends a BIND request to the SLU, LU *x*, to activate the session. After LU *x* responds affirmatively to the BIND request, LU *x* and LU *z* each inform their SSCPs of the new session with a Session Started (SESSST)

message. PU *c* also notifies the gateway SSCPs with Notify messages. Finally, the SSCP of the PLU, SSCP *b*, sends a CDSESSST request to the SSCP of the SLU, SSCP *a*, so that the SSCPs can synchronize their *session awareness* records.



7 CDCINIT request from SSCP a to SSCP b
(SSCP-SSCP session)

8 CINIT request from SSCP b to LU z
(SSCP-LU session)

9 Session-Activation (BIND) request from LU z to LU x
(LU-LU session)

10 SESSST request from LU x to SSCP a
(SSCP-LU session)

11 Notify requests from PU c to SSCP a and SSCP b
(SSCP-PU sessions)

12 SESSST request from LU z to SSCP b
(SSCP-LU session)

13 CDSESSST request from SSCP b to SSCP a
(SSCP-SSCP session)

Legend:
GWN = Gateway Node
LU = Logical Unit
PU = Physical Unit
SSCP = System Services Control Point

Figure 96. Initiating a Cross-Network LU-LU Session: Part II

Initiating Same-Domain LU-LU Sessions Using SSCP-Independent Protocols

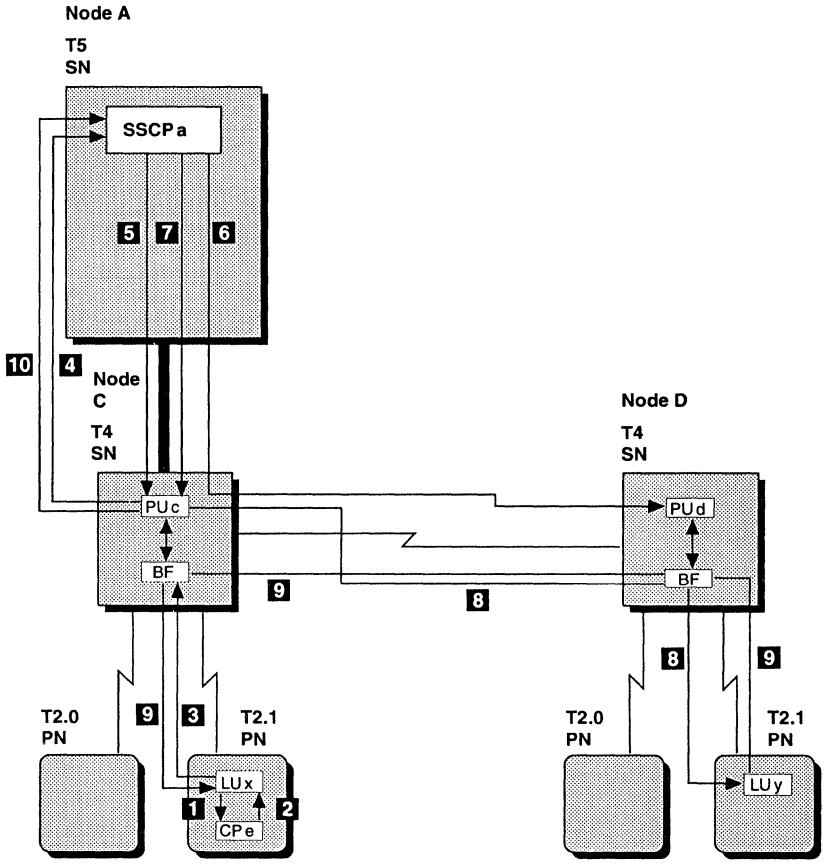
A session-initiation request from an independent LU (in a T2.1 peripheral node) is sent to its local CP. The CP then constructs a CINIT request, containing the BIND image, and returns it to the independent LU. The independent LU then becomes the primary LU (PLU). As with SSCP-dependent protocols, the PLU constructs a BIND request and sends it to the SLU.

In a subarea network, the BIND request from an independent LU in a T2.1 peripheral node does not flow directly to the SLU in another T2.1 peripheral node. Certain message exchanges must first take place between the appropriate T4 and T5 subarea nodes in order to prepare all T4 nodes along the session route to perform peer-session passthrough. Recall that independent LUs are not predefined to SSCPs. One objective of the subarea node message exchanges, therefore, is to assign network addresses to the independent LUs.

Figure 97 illustrates session initiation using SSCP-independent protocols between two LUs in the same domain, LU *x* and LU *y*. When CP *e* receives the session-initiation request from LU *x*, it builds a CINIT request and returns it to the PLU, LU *x*. LU *x* then attempts to activate the session by sending a BIND request to LU *y*. The request is intercepted, however, by PU *c*, which sends a Boundary Function Initiate (BFINIT) request to SSCP *a*.

In order to acquire network addresses for the partner LUs, SSCP *a* sends Request Network Address Assignment (RNAA) requests to the T4 physical units controlling the subareas of the partner LUs, PU *c* and PU *d*. The PUs assign network addresses for the LUs, store the addresses, and return them to the SSCP *a* in the RNAA responses. SSCP *a* then sends a Boundary Function Control Initiate (BFCINIT) request to the T4 PU from which it received the BFINIT, PU *c*.

After PU *c* receives the BFCINIT request, PU *c* forwards the BIND request to LU *y*. When LU *y* receives the BIND request, it sends a BIND response back to LU *x*. As the BIND request flows through subarea nodes C and D to LU *y*, and the BIND response flows back from LU *y* to LU *x*, the BFs in nodes C and D perform the necessary network address transforms in the message transmission headers. After the BIND response flows back to LU *x*, PU *c* sends a Boundary Function Session Started (BFSESSST) to the SSCP from which it received the BFCINIT, SSCP *a*.



Legend:

BF	= Boundary Function
BF.LU x	= BF for LU x
BF.LU y	= BF for LU y
CP	= Control Point
LU	= Logical Unit
PU	= Physical Unit
SSCP	= System Services Control Point

1. Session-Initiation request from LU x to CP e
2. CINIT request from CP e to LU x
3. Session-Activation (BIND) request from LU x to LU y, intercepted by BF.LU x
4. BFINIT request from PU c to SSCP a (SSCP-PU session)
5. RNAA request from SSCP a to PU c (SSCP-PU session)
6. RNAA request from SSCP a to PU d (SSCP-PU session)
7. BFCINIT requests from SSCP a to PU c (SSCP-PU session)
8. Session-Activation (BIND) request from BF.LU x to LU y via BF.LU y
9. Session-Activation (BIND) response from LU y to LU x via the BFs in nodes D and C (LU-LU session)
10. BFSESSST from PU c to SSCP a (SSCP-PU session)

Figure 97. Initiating a Same-Domain LU-LU Session Using SSCP-Independent Protocols

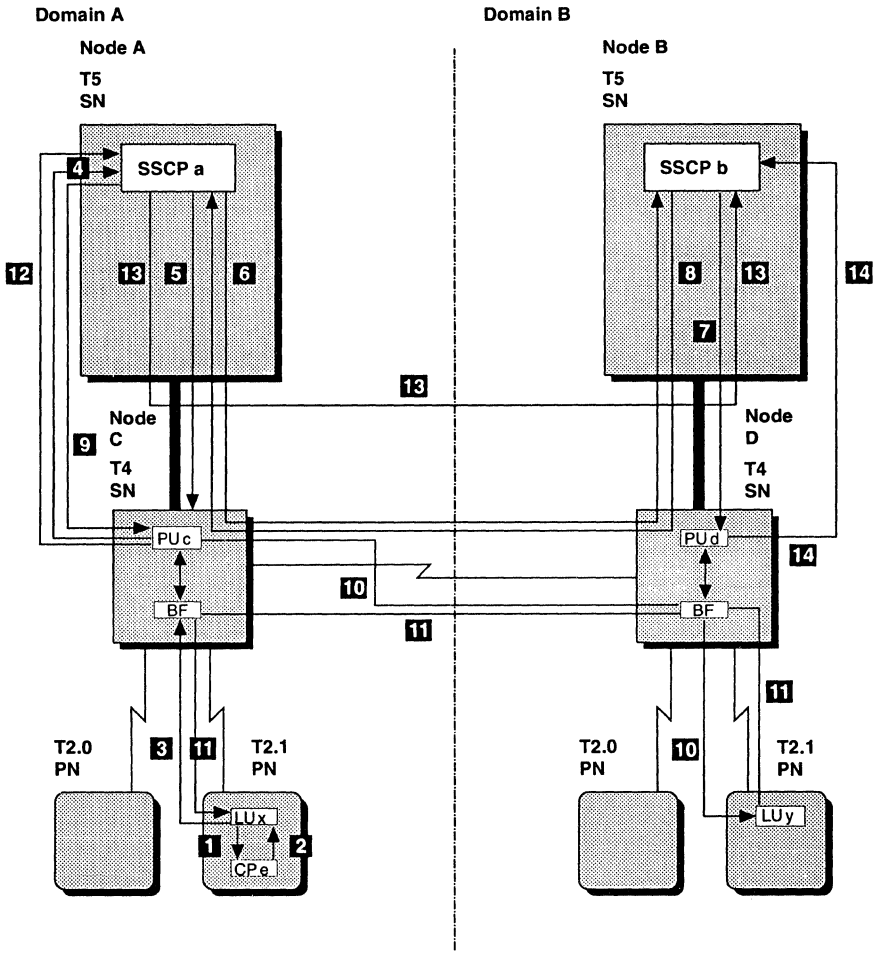
Initiating Cross-Domain LU-LU Sessions Using SSCP-Independent Protocols

As with same-domain LU-LU session-initiation using SSCP-independent protocols, with cross-domain LU-LU session-initiation a series of message exchanges between the T4 and T5 subarea nodes prepares the T4 nodes for peer-session passthrough. In the cross-domain environment, however, two SSCPs, those controlling the origin and destination domains, participate in the message exchanges.

Each SSCP requests a network address for the LU in its domain, from the T4 node controlling the subarea of the LU.

Figure 98 on page 207 illustrates session initiation using SSCP-independent protocols between two LUs in different domains, LU *x* and LU *y*. The flow of the session-initiation and BFINIT requests is the same as in the same-domain scenario. After SSCP *a* receives the BFINIT request, however, it sends only a single RNAA request to the T4 PU in its own domain, PU *c*. PU *c* assigns a network address for LU *x* and returns it to SSCP *a* in the RNAA response. SSCP *a* then sends a Cross-Domain Initiate (CDINIT) request to SSCP *b* identifying the destination LU, LU *y*. SSCP *b*, in turn, exchanges an RNAA request and response with the T4 PU in its own domain, PU *d*, to obtain a network address for LU *y*. SSCP *b* then sends the address to SSCP *a* in a Cross-Domain Control Initiate (CDCINIT) request.

Having received network addresses for both LUs, SSCP *a* now sends a Boundary Function Control Initiate (BFCINIT) request to the T4 PU from which it received the BFINIT, PU *c*. As in the same-domain scenario, PU *c* then forwards the BIND request to LU *y*, and LU *y*, returns the BIND response to LU *x*. After the BIND response flows back to LU *x*, PU *c* sends a Boundary Function Session Started (BFSESSST) to the SSCP from which it received the BFCINIT, SSCP *a*. SSCP *a* then sends a Cross-Domain Session Started (CDSESSST) to the SSCP from which it received the CDCINIT, SSCP *b*. PU *d* sends a BFSESSST to SSCP *b*.



Legend:

CP = Control Point
LU = Logical Unit
PU = Physical Unit
SSCP = System Services Control Point

1. Session-Initiation request from LU x to CP e
2. CINIT request from CP e to LU x
3. Session-Activation (BIND) request from LU x to LU y, intercepted by BF.LU x
4. BFINIT request form PU c to SSCP a (SSCP-PU session)
5. RNAA request from SSCP a to PU c (SSCP-PU session)
6. CDINIT request from SSCP a to SSCP b SSCP-SSCP session)
7. RNAA request from SSCP b to PU d (SSCP-PU session)
8. CDCINIT request from SSCP b to SSCP a (SSCP-SSCP session)
9. BFCINIT requests from SSCP a to PU c (SSCP-PU session)
10. Session-Activation (BIND) request from BF.LU x to LU y via BF.LU y
11. Session-Activation (BIND) response from LU y to LU x via the BFs in nodes D and C (LU-LU session)
12. BFSESSST from PU c to SSCP a (SSCP-PU session)
13. CDSESSST from SSCP a to SSCP b (SSCP-SSCP session)
14. BFSESSST from PU d to SSCP b (SSCP-PU session)

Figure 98. Initiating a Cross-Domain LU-LU Session Using SSCP-Independent Protocols

Initiating LU-LU Sessions in APPN Networks

In an APPN network, LU-LU session initiation typically involves exchanges between multiple control points (CPs). The CPs cooperate in providing partner LU location and route information to the initiating LU. The location and route information is derived from dynamically updated databases such as the distributed network directory and the network topology database.

As in a subarea network, the process of activating an LU-LU session in an APPN network begins when an LU submits a session-initiation request to its control point (CP). The session-initiation request identifies the two logical units that are to participate in the LU-LU session and specifies a mode name for the session. As in a subarea network, the CP uses the mode name to obtain a set of session parameters that the requesting logical unit can support, builds a Bind image, and transmits it in a CINIT request to the primary logical unit (PLU).

In APPN networks, it is typical for LU-LU sessions to connect logical units in different domains. As with cross-domain LU-LU session-initiation in subarea networks, the two control points serving the partner LUs must exchange control information before the sessions are activated. Unlike subarea networks, however, the CPs serving the partner LUs do not need to establish a session with one another. The two endpoint CPs are already indirectly connected through the preexisting CP-CP sessions in the network.

Figure 99 illustrates the sequence for initiating a session between two LUs, LU *a* and LU *c*, in an APPN network. LU *a* initiates the session by sending a session-initiation request to CP *a* specifying LU *c* as the session partner. When the CP in end node A, CP *a*, receives the session-initiation request, it checks its directory and determines that LU *c* is not a local resource. CP *a* directory services conducts a one-hop search to the CP in its network node server, CP *b*, for LU *c*.

In CP *b*, session services handles the search request. Session services first calls upon directory services to obtain the name of the node containing LU *c*. This could take the form of either a directed or broadcast search. Session services then calls upon topology and routing services to calculate a route for the session. (For further information on directory search message exchanges and route calculation, see Chapter 6, "Establishing Routes Through the Network.")

After locating the partner LU and calculating a route for the session, CP *b* returns a search reply to CP *a* containing the route selection control vector (RSCV). CP *a* then sends a CINIT message to LU *a*, enabling LU *a* to activate the session. LU *a* activates the session by sending a BIND request with the session-route RSCV to CP *a*. The address space managers in CPs *a*, *b*, and *c* use the transmission group (TG) vectors in the RSCV to route the BIND request along consecutive TGs to LU *c*. As the BIND flows from node to node, half-sessions are initialized in the partner LUs, and session-connectors are initialized in the ISR component of the intermediate routing node, Node B. The half-sessions and session-connectors then participate in routing the BIND response from LU *c* back to LU *a*.

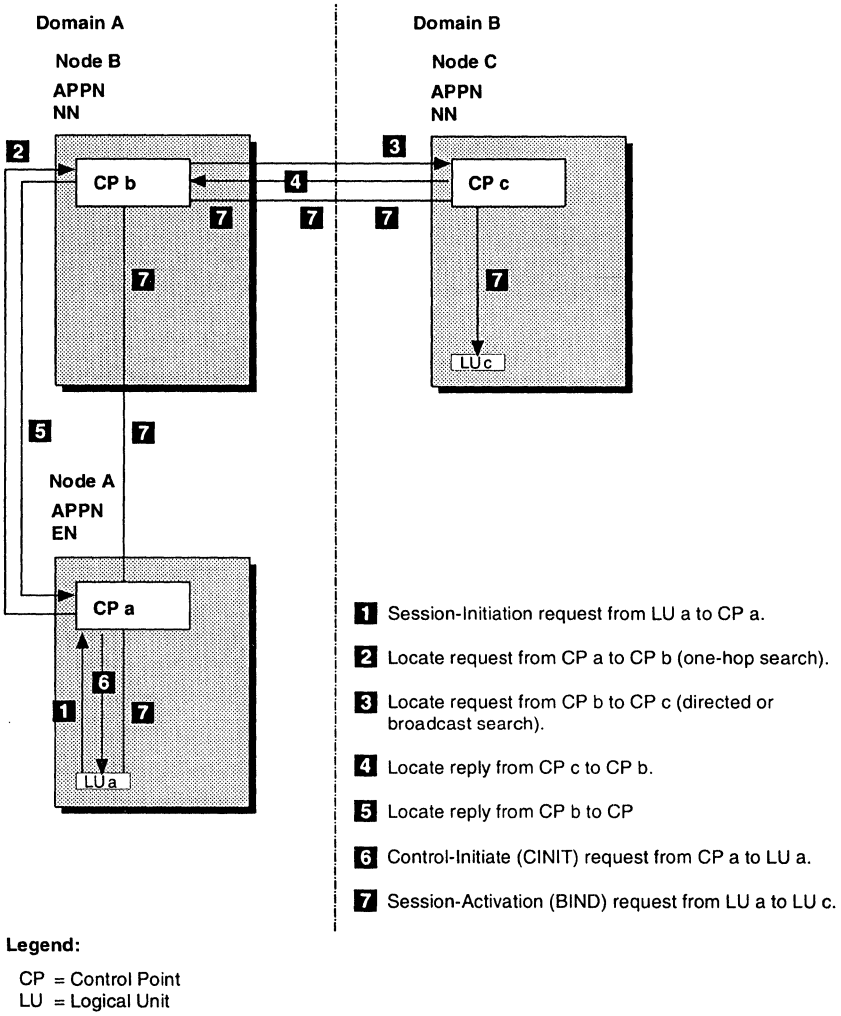


Figure 99. Initiating an LU-LU Session in an APPN Network

Initiating LU-LU Sessions in HPR Networks

Initiating an LU-LU session in an HPR network is done as in basic APPN with the following exceptions:

- A regular APPN directed search is performed to locate the target LU.
- The RSCV is calculated using the basic APPN algorithm. If HPR routing information is needed for this desired route, then a route setup protocol is performed. The route setup protocol obtains routing information such as forward and reverse ANR labels associated with this route.
- An RTP connection is activated to carry the LU-LU session traffic. A connection setup message is sent together with the BIND (in the transport header of the data portion in the NLP) to the partner. Each partner endpoint of the RTP connection generates an identifier for the other endpoint to use in sending data over the RTP connection. An already active RTP connection may be used for the LU-LU session, in which case it is not necessary to perform the route setup protocol and establish the RTP connection. The BIND is simply sent as normal data over the existing RTP connection.

Rapid-Transport Protocol (RTP) in HPR

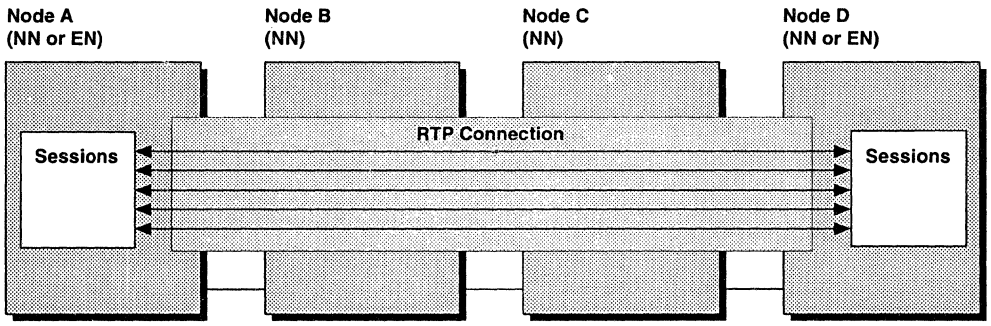
Rapid-transport protocol (RTP) provides high-performance, reliable, connection-oriented service with full-duplex delivery of arbitrary-length messages and automatic path-switching without disrupting sessions. RTP is used to establish an RTP connection between session endpoints (which can be in either the base-APPN subnetwork or the HPR portion of the network) and to transport APPN LU-LU session data over this connection in such a way that intermediate nodes are not aware of SNA sessions, or even of the transport connection itself.

Data delivery is reliable for APPN; however, RTP is designed to be implemented over networks that are not completely reliable and have fixed maximum length messages; thus, the RTP protocol performs message segmenting, reassembly, sequence checking, retransmission, and in-order delivery at session endpoints. Since RTP provides reliability/error recovery, in-order delivery at session endpoints, and end-to-end flow/congestion control, it eliminates the need for these protocols on each hop along the end-to-end path. This improves the overall throughput performance of HPR.

RTP provides ***end-to-end flow control and congestion control*** called ***adaptive rate-based (ARB)*** flow/congestion control. For more information on ARB, see "Adaptive Rate-Based (ARB) Flow/Congestion Control in HPR Networks" in Chapter 7, "Controlling Congestion in the Network." The physical path used by the RTP connection satisfies the class of service (COS) associated with the sessions routed over it. Traffic from many sessions requesting the same COS can be routed over a single RTP connection. RTP connections are used to:

- Transport CP-CP session traffic
- Transport LU-LU session traffic
- Carry route setup requests required to establish LU-LU sessions over a series of RTP connections.

Figure 100 illustrates rapid-transport protocol.



Legend:

NN = Network node
EN = End node
RTP = Rapid-transport protocol

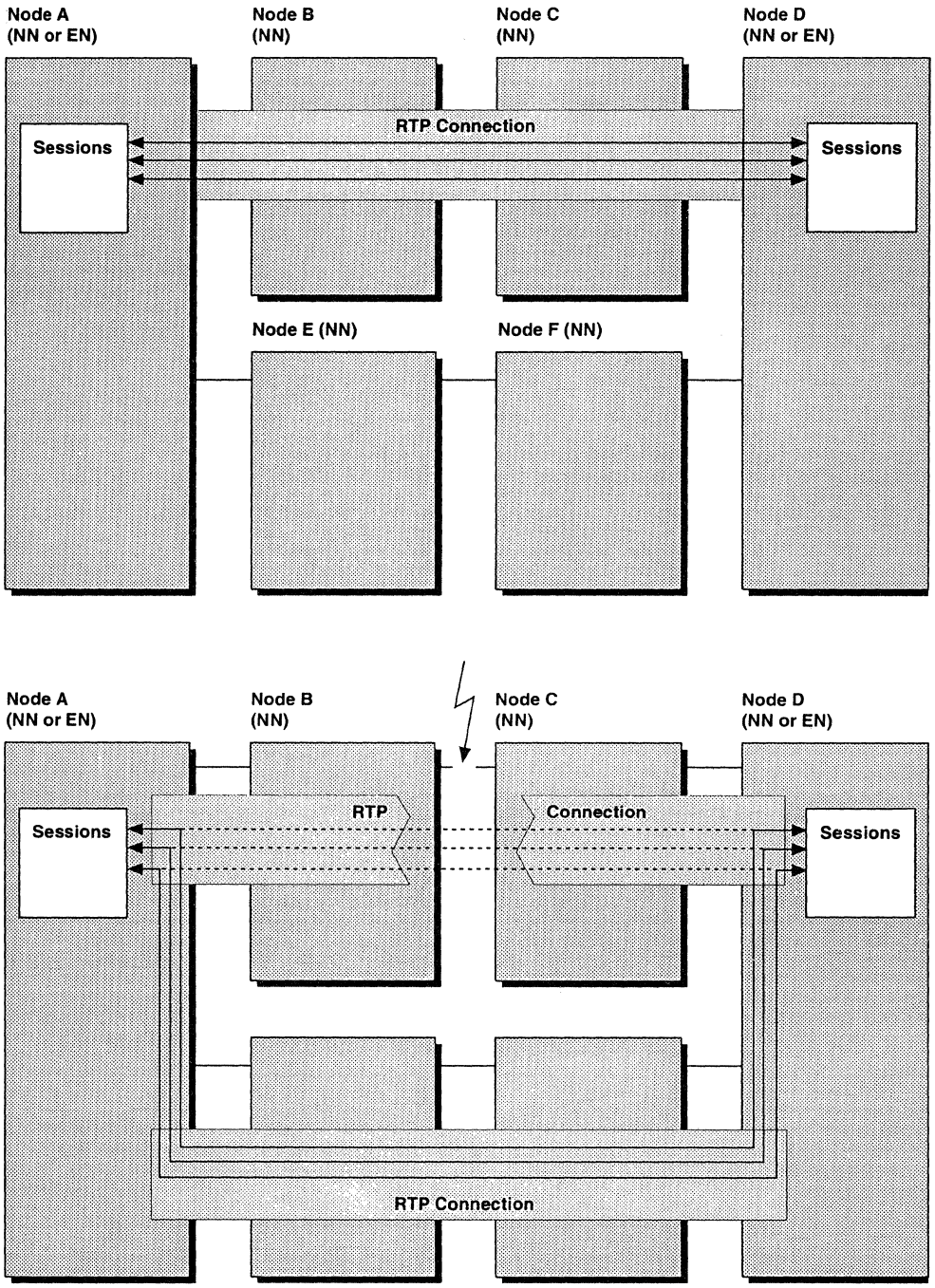
Figure 100. Rapid-Transport Protocol

An RTP connection has the following features:

- It transports data at very high speeds by using lower-level intermediate routing (ANR) and by minimizing the number of flows over the links for error recovery and flow control. These functions are performed at the session endpoints rather than at each hop along the path. For more information on ANR, see

"Automatic Network Routing (ANR) in HPR" in Chapter 6, "Establishing Routes Through the Network."

- It can be switched automatically to reroute data around a failed node or link without disrupting the LU-LU sessions. This is called ***nondisruptive path switch*** because LU-LU sessions survive link failures. Nondisruptive path switch within the HPR portion of the network automatically occurs in an HPR subnet to bypass link and node failures if an acceptable alternate path is available. However, all the nodes on both the failed path and the new path must be HPR-capable. This function does not operate within or across a basic APPN subnet. Figure 101 illustrates nondisruptive path switch.



Legend:

NN = Network node
EN = End node
RTP = Rapid-transport protocol

Figure 101. Nondisruptive Path Switch

- It provides transport of data in a reliable manner. RTP guarantees delivery of all data and performs the necessary end-to-end error recovery. HPR uses selective retransmission, which eliminates the need for error recovery at each hop. Selective retransmission transmits only packets that were lost, rather than retransmitting the lost packet and all the following packets.

- It provides rate-based flow/congestion control. HPR uses the adaptive rate-based (ARB) flow/congestion algorithm. This eliminates the need of flow control at each hop.
- It performs segmentation and reassembly. RTP performs the necessary segmenting and reassembly of messages based on the minimum link BTU size on the path between the two endpoints of the RTP connection.

Route Setup Protocol

The route setup protocol is initiated whenever it is necessary to obtain information about a route over which an RTP connection will be established. The protocol consists of a route setup request and a route setup reply. Requests and replies stop at each intermediate node along the path to gather information such as ANR labels.

When the origin node receives the route setup reply, it has all the information it needs to establish an RTP connection and ensures that all links along the path are active and operational.

The route setup request and reply is forwarded hop by hop from route setup component to route setup component in the nodes along the path over the TGs specified in the RSCV for the desired route. The route setup requests and replies are transported reliably over the appropriate link using a previously established RTP connection.

Properties of RTP Connections

RTP connections are used to transport session data between HPR nodes operating within an HPR subnet. An RTP connection has the following properties:

Single COS per connection: Each RTP connection transports session data for a single COS as specified in the BIND. An RTP connection is not used for more than one COS. For example, the COS may specify that the path for the RTP connection must be an all-HPR path; this is necessary in order to take advantage of nondisruptive path switching.

Connections can be used for both directions: An RTP connection is activated by one node to a destination node to carry sessions originating at this node. This connection can also be used by the partner node for sessions originating at the partner node. All traffic from an individual session flows over a single RTP connection, but many sessions can be multiplexed over a single RTP connection. All sessions requesting the same COS and following the same path through the HPR subnet are transported over a single RTP connection between the HPR nodes containing the session endpoints.

RTP Connection Activation/Deactivation

An RTP connection is activated when a node wants to activate a session and a suitable RTP connection does not already exist. An RTP connection is deactivated when no sessions are using it.

RTP Error Recovery

RTP employs ***end-to-end error recovery*** to improve throughput. With end-to-end error recovery, no link-level error recovery is required and selective retransmission is possible. The RTP connection endpoints perform the error recovery. In the past, error recovery on each hop of a network route was necessary because of the high link-error rates. However, improvements in link-error rates make it feasible and desirable to provide end-to-end error recovery in place of error recovery on each hop. HPR provides this capability by:

- Using existing links and DLCs so that they bypass link-level error recovery. For example, for SDLC, data can be sent as unnumbered information frames.
- Using RTP to perform end-to-end error recovery on RTP connections. RTP retransmits only those packets that failed to reach the receiver (selective retransmission).

Figure 102 illustrates the difference between basic APPN and HPR in the way error recovery is handled.

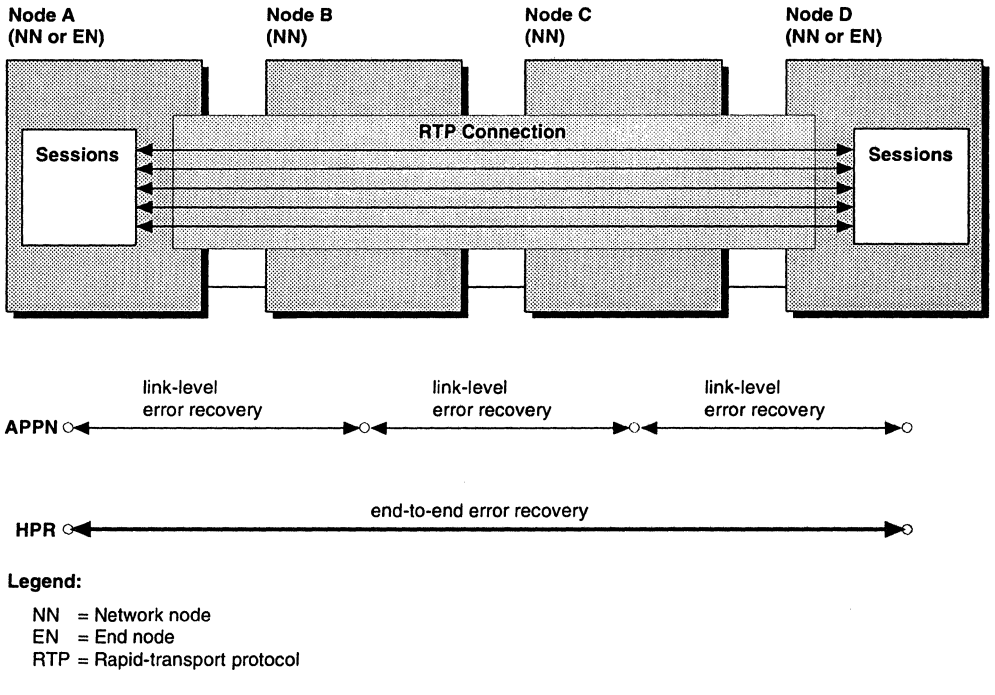


Figure 102. APPN/HPR Error Recovery

Initiating LU-LU Sessions in Networks with Both HPR and APPN Nodes

HPR uses CP-CP sessions just as in basic APPN, with the exception that formats are different. All HPR traffic (including CP-CP) is carried in network layer packets (NLPs). A FID5 transmission header is used for all session traffic that flows over RTP connections. For more information on network layer packet, see "Network Layer Packet (NLP)" in Chapter 3, "Data Formats."

BIND Negotiation

A BIND request specifies, as parameters, the protocols that the primary and secondary LUs are to observe when communicating with each other over an LU-LU session. In both subarea and APPN networks, there are two kinds of BIND requests: nonnegotiable and negotiable.

Nonnegotiable BIND

Upon receiving a ***nonnegotiable BIND request***, the secondary LU (SLU) can accept or reject the parameters of the BIND. If the SLU accepts the session parameters, it returns a positive response to the primary LU (PLU). The LU-LU session is then active, and both logical units conform to the parameters as specified in the BIND request.

The SLU rejects the BIND request if the session parameters are unacceptable. The SLU rejects the BIND request by returning a negative response to the PLU. An LU-LU session for which a BIND request has been rejected cannot be activated.

Negotiable BIND

LU types 4, 6.1, and 6.2 can *negotiate* BIND parameters. Upon receiving a ***negotiable BIND request***, the SLU determines the acceptability of the BIND parameters. Certain parameters, called ***negotiable parameters***, can be changed by the SLU. For example, if the BIND request designates the PLU as the contention winner for the session, but installation-defined specifications require that the SLU be the contention winner, the SLU can change the parameter to indicate that it will be the contention winner.

For each negotiable BIND parameter that is unacceptable to the SLU, the SLU returns an alternate session parameter to the PLU in the (positive) BIND response. If the PLU accepts the alternate set of session parameters, the LU-LU session becomes active. Both logical units conform to the alternate set of parameters. If the alternate set of session parameters is unacceptable to the PLU, the LU-LU session is deactivated by the PLU sending an immediate UNBIND request to the SLU.

Data Flow Control Protocols

A session-activation request carries a function management profile identifier. Function management profiles define protocol options relevant to the data flow control layer for the session. These protocols control message sequencing and message response requirements and are used by logical units in both subarea and APPN networks. Data flow control protocols fall into the following categories:

- Bracket Protocols
- Response Protocols
- BIU Sequencing Protocols
- Request and Response Mode Protocols
- Send and Receive Mode Protocols.
- Chaining Protocols.

Bracket Protocols

Logical units use brackets to separate groups of related request chains and responses from other groups of related request chains and responses. A ***chain*** is a series of requests treated as a single transfer of data. (For further information on chains, see "Chaining Protocols" on page 220.) A ***bracket*** delimits a sequence of request chains and responses exchanged in either or both directions between two logical units. During session activation, BIND parameters specify whether brackets will be used during the session.

The sequence of exchanges within a bracket is considered to comprise a ***transaction***. Between two type 6.2 LUs, the dedicated use of a session for executing a transaction is called a ***conversation***. A conversation supports the execution of a transaction by partner transaction programs. A transaction program (TP) begins a conversation by issuing an ***ALLOCATE*** verb. The LU supporting the TP allocates the conversation to a session by sending an ***Attach*** FM header (FM header type 5) to the partner LU over the session. The Attach names the partner transaction program and begins a bracket. Another conversation cannot be started on the session until the current conversation is ended. A transaction program ends a conversation by issuing a ***DEALLOCATE*** verb. The LU supporting the TP deallocates the conversation by ending the bracket. A session is *serially reused* by multiple conversations, and each conversation is delimited by a bracket. (For further information on conversations and verbs, see "LU 6.2 Protocols" on page 224.)

A bracket includes the first request header through the last response unit of related request chains and their responses. A bracket can be a monolog or a dialog. In a ***monolog*** (also referred to as a ***one-way bracket***), a bracket delimits a unidirectional chain or series of chains sent from one session partner to the other. In the context of an LU 6.2 conversation, a one-way bracket is called a ***one-way conversation***. In a ***dialog***, a bracket delimits a bidirectional series of chains exchanged between the session partners. In both cases, a bracket identifies a sequence of related requests and responses that flow between session partners.

To identify the beginning of a bracket, a logical unit sets the ***begin bracket*** (BB) indicator in the request header of the first request in the first chain of the bracket. To identify the end of a bracket, a type 6.2 logical unit sets the ***conditional end bracket*** (CEB) indicator in the request header of the last request in the last chain of the bracket. The CEB indicates that bracket termination is conditional upon the return of a positive response by the receiving LU. Logical units other than type 6.2 use the ***end bracket*** (EB) indicator, not the CEB, to identify the end of a bracket. They set the EB in the request header of the *first* request in the last chain of the bracket.

For additional information on the protocols that govern the use of bracket indicators (BB, CEB, and EB), refer to *SNA LU 6.2 Reference—Peer Protocols* and *SNA Format and Protocol Reference Manual: Architectural Logic*.

Response Protocols

When a logical unit sends a request to its session partner, it sometimes needs to know if its session partner received the information. A session partner acknowledges the receipt of a request by returning a response to the sender. Indicators in the request header (RH) identify one of three SNA response protocols requested: definite response, exception response, or no response.

Definite Response

If a request specifies that a ***definite response*** is requested, the request receiver must return either (1) a positive response to accept the request or (2) a negative response to reject the request. The request sender sets definite response indicators (DR1I, DR2I) in the request header to identify the definite response protocol. DR1I is set by a sending LU to request a definite response by the receiving LU. DR2I is set by a sending LU, at the request of a sending transaction program, to request a definite response by the receiving transaction program. DR2I is used in confirmation and sync point processing by type 6.2 LUs. For information on confirmation and sync point processing, see "Synchronization Processing" on page 235.

Exception Response

If a request specifies that an ***exception response*** is requested, the request receiver returns a negative response for any unacceptable requests. The receiver does not return any positive responses. The request sender sets the exception response indicator (ERI), along with the definite response 1 indicator (DR1I), in the request header to identify the exception response protocol.

No Response

If a request specifies that ***no response*** is requested, the request receiver returns neither a positive nor a negative response. Requests sent by type 6.2 logical units do not use the no-response protocol; they always specify either a definite or an exception response.

BIND parameters specify which type of response protocols logical units will abide by while communicating over an LU-LU session. If definite response protocols are specified for all LU-LU sessions, unnecessary link traffic can result. If an LU-LU session requires transmission accuracy, the session should not use the no-response protocol. Using the no-response protocol, the request sender does not know if errors have occurred or if data has been lost. There is no response to indicate whether the request was acceptable or unacceptable.

BIU Sequencing Protocols

SNA session protocols require that a logical unit receive related requests and responses in the same order in which they were sent by its session partner. A logical unit assigns a sequence number to each request (BIU) that it sends over an LU-LU session. Session partners assign sequence numbers independently of one another. Data flow control in the sending LU assigns the sequence numbers, and transmission control in the receiving LU verifies them.

A logical unit assigns the sequence number 1 to the first request that it sends after session activation. Then the logical unit increases the sequence number by 1 for every subsequent request that it sends to its session partner. The maximum sequence number is 65,535; after reaching the maximum it wraps back to 0. The receiving logical unit assigns responses the same sequence number as their associated requests. Sequence numbers enable logical units to identify which response is associated with which request.

Request and Response Mode Protocols

Request and response mode protocols control when logical units can send and receive requests and responses over an LU-LU session. The BIND command specifies the request and response modes. The modes that the BIND indicates for traffic in one direction are independent of the modes that it indicates for traffic in the reverse direction. SNA defines four request and response modes: immediate request mode, delayed request mode, immediate response mode, and delayed response mode.

Immediate Request Mode

The ***immediate request mode*** requires that after sending a request, the sending logical unit wait for a response from the receiver before sending any additional requests. If the sender groups the request units into chains, it must wait for the receiver to return a response to the last chain sent before it can send any additional chains. The immediate request mode applies only to requests that specify definite responses. It does not apply to a request that specifies an exception response or no response.

Delayed Request Mode

The ***delayed request mode*** allows the sending logical unit to send additional requests without waiting for any responses. A type 6.2 logical unit does not use the delayed request mode; it always requires a response to its last request before it sends another.

Immediate Response Mode

The ***immediate response mode*** requires that the receiving logical unit return responses in the same order in which it received the requests. For example, if request A is received before request B, the response to request A must be returned before the response to request B. Responses need not follow their respective requests immediately, but they must be returned in the same sequence as that in which the requests were received.

Delayed Response Mode

The ***delayed response mode*** allows the receiving logical unit to return responses in any order. For example, if request A is received before request B, the response to either request can be returned first. Responses need not follow their respective requests immediately and they can be returned to the sender in any sequence. A type 6.2 logical unit does not use the delayed response mode; it always sends responses in the same sequence as the sequence in which their corresponding requests were received.

Send and Receive Mode Protocols

Send and receive mode protocols determine when each LU-LU session partner is allowed to send data on the session, and they determine how to resolve contention between the partners. Contention occurs when both partners try to send data at the same time. In a contention situation, the LU that wins the right to transmit is referred to as the ***contention winner***; the partner LU is referred to as the ***contention loser***. SNA defines three send and receive modes: half-duplex contention, half-duplex flip-flop, and full-duplex.

Do not confuse send and receive mode protocols with transmission medium protocols. Transmission media are either full duplex or half duplex. A ***full-duplex***

transmission medium refers to the capability of the medium to transmit data in two directions simultaneously. A ***half-duplex transmission medium*** refers to the capability of the medium to transmit data in only one direction at a time. LU send and receive modes are independent of the physical properties of the network.

Half-Duplex Contention

On a session using ***half-duplex contention*** send and receive mode protocol, contention occurs when partner LUs simultaneously send requests to each other. When contention occurs, it is always resolved in favor of the secondary LU (SLU). If the SLU receives a request from the primary LU (PLU) after having just sent a request to the PLU, the SLU rejects the request and the PLU must enter *receive state* (be ready to receive and respond to requests from the SLU).

Half-duplex contention mode is not a peer-oriented protocol; the SLU is always deemed the contention winner because of its presumed processing limitations with respect to the PLU. For example, between a workstation and an application program using half-duplex contention mode, the workstation is normally designated the secondary LU. It is assumed that whereas the program can buffer data to be sent to the workstation, the workstation cannot normally buffer data to be sent to the program.

Half-Duplex Flip-Flop

On a session using ***half-duplex flip-flop*** send and receive mode protocol, one session partner is designated by BIND parameters as the first speaker and the other as the bidder. The ***first speaker*** begins in the send state; it can begin a bracket without asking for permission from its session partner, the bidder. The ***bidder*** begins in the receive state; it must ask for and receive permission from the first speaker to begin a bracket.

Contention can occur immediately following session activation, or when the session is between brackets. It occurs when the bidder requests permission to begin a bracket just as the first speaker transmits a message unit. When contention occurs, it is always resolved in favor of the first speaker. If the first speaker receives a bid from the bidder after having just sent a request to the bidder, the first speaker rejects the request and the bidder must remain in receive state.

Half-duplex flip-flop mode is more peer-oriented than half-duplex contention mode. When the session is between brackets, no contention occurs because the partner LUs alternate send and receive state on the session. This is done by use of the ***change direction*** (CD) indicator carried in the request header. When the logical unit in send state has finished sending data, it enters receive state and allows the other LU to enter send state by setting the change direction indicator (CDI) in the request header of the last request unit sent. The two logical units thus continually *flip-flop* between send and receive states until the session is terminated.

Type 6.2 logical units use the half-duplex flip-flop mode in support of conversation protocols. As a conversation alternates between send and receive state, the underlying session likewise alternates between send and receive state. Examples of verbs that cause a conversation to switch from send state to receive state include RECEIVE_AND_WAIT, PREPARE_TO_RECEIVE, and FLUSH.

Because a type 6.2 LU can support multiple sessions, it can be a contention winner on some sessions and a contention loser on others. Sessions on which it is designated the contention winner are called ***contention-winner sessions***; sessions on

which it is designated the contention loser are called ***contention-loser sessions***. See "Session Limits" on page 232 for information on how limits on the number of contention-winner and contention-loser sessions for a type 6.2 LU are specified.

Full-Duplex

A session using the ***full-duplex*** send and receive mode protocol uses the same protocol to resolve contention while starting a bracket as ***half-duplex flip-flop***. When the session is between brackets, no contention occurs because both LUs are allowed to send data simultaneously on the session.

Type 6.2 logical units use the full-duplex mode in support of conversation protocols. Logical units using the full-duplex mode support both full-duplex and half-duplex conversations. Full-duplex conversations allow sending and receiving data concurrently. However, half-duplex conversations use the change direction indicator to control which program is allowed to send data.

Chaining Protocols

A ***chain*** is a sequence of BIUs that constitute a single, unidirectional transfer of data. Logical units use chains to transmit related request units (RUs) as a single entity or to establish a series of requests as a single unit of error recovery. An LU may need to transmit a series of related RUs, rather than a single RU, to conform to the BIND request for the session. The ***RU-size parameter*** limits the size of a request unit (RU) that two logical units can send to each other. In order to send a request that contains more information than can fit into one request unit, the sending LU divides the information into a series of separate requests. Even though a chain is a series of separately transmitted requests, the receiving logical unit either accepts or rejects the chain as a whole; it returns only one response to the sending logical unit.

There are both single-element and multiple-element chains. A single-element chain consists of (1) a command RU, (2) a single data RU, or (3) a response RU. A multiple-element chain consists of multiple data RUs. Every RU belongs to a chain.

A logical unit sets indicators in request headers to identify the beginning and end of a chain for the receiving logical unit. A logical unit sets the begin chain indicator (BCI) in the request header of the first request to identify the beginning of a chain; it sets the end chain indicator (ECI) in the request header of the last request to identify the end of a chain. A logical unit sets both the BCI and ECI in the request header of a single-element chain.

In an LU 6.2 conversation, a transaction program (TP) can issue consecutive send-type verbs before issuing a receive-type verb. The LU supporting the TP transmits the resulting series of requests as a chain. The chain is ended when the TP issues a verb that causes a transition to receive state. Any requests then received from the partner LU are also received as a chain.

The following sections explain how the sending logical unit specifies a response protocol (definite response, exception response, or no response) for a multi-element chain and how the receiving logical unit returns the specified response to its session partner.

Definite Response Chain

The sending logical unit sets the exception response indicator in the request headers of all but the last request in the chain; it sets the definite response indicator in the request header of the last request in the chain.

After receiving the last element of the chain, the receiving logical unit returns a positive response to acknowledge that the entire chain is acceptable. If any one of the chain elements is unacceptable, the receiving logical unit returns a negative response for the entire chain.

Exception Response Chain

The sending logical unit sets the exception response indicator in the request headers of all the requests in the chain.

The receiving logical unit never returns a positive response for the chain. However, it returns a negative response for the entire chain if any of the requests in the chain is unacceptable.

No-Response Chain

The sending logical unit indicates no-response requested in each request header of the chain. The receiving logical unit never returns a response for the chain. Because type 6.2 logical units do not use the no-response protocol, they do not send no-response chains.

Extended Recovery Facility Sessions

The extended recovery facility (XRF) is an SNA enhancement that increases the availability of online database/data-communication transaction processing in a subarea network. It is available for Information Management System/Virtual Storage (IMS/VS) and Customer Information Control System/Multiple Virtual Storage (CICS/MVS). XRF is incorporated in the boundary function and in host-node logical unit types 1, 2, and 4.

Network designers have employed a variety of approaches to improve system availability; all involve cost justification of the added hardware and software components that are required. SNA employs an approach in which an *active* system and a *backup* system share access to a common database. The active system processes the online work while the backup system is available in case of failure. With this approach, the backup system does not represent a wasted resource, since it can be used to perform other work until it is called upon to take over for the active system.

The extended recovery facility uses hardware and software components to back up sessions in the active system through the backup system, in order to maintain end-user access to the database in the event of certain system failures. Service disruption is minimized and end-user involvement in the recovery process is simplified. End users may often, in fact, be unaware of the disruption.

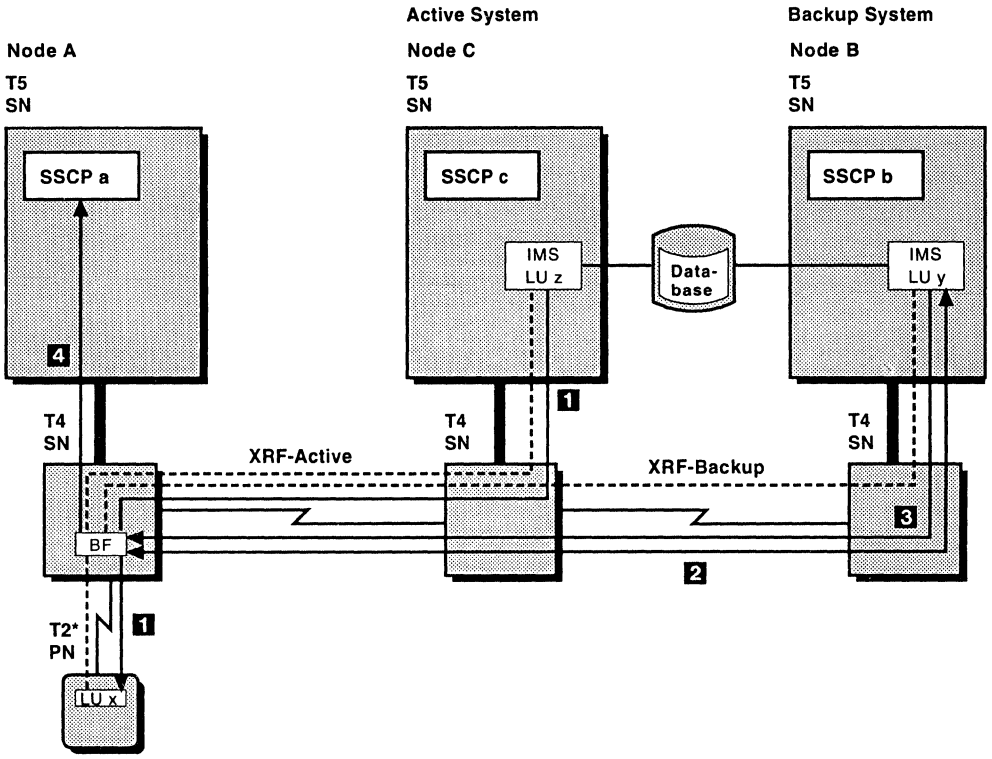
Initiating Extended Recovery Facility Sessions

The extended recovery facility establishes an ***XRF-backup session*** for each ***XRF-active*** session. This minimizes the time required to recover from a failure. The XRF-backup session remains inactive unless the XRF-active session fails, at which time XRF may deactivate the failed session and activate the XRF-backup session.

The XRF-backup session extends from the backup primary logical unit (PLU) to an ***XRF switchpoint*** located in the boundary function for the secondary logical unit (SLU). The XRF-active session extends from the active PLU to the SLU, passing through the switchpoint.

Figure 103 on page 223 illustrates an XRF configuration with an **XRF-active** session from LU *z* to LU *x* and an **XRF-backup** session from LU *y* to the XRF switchpoint in the boundary function of LU *x*. (In this figure the paths of the sessions are shown by dashed lines.)

An XRF-active session is established in a manner similar to that for a non-XRF session between secondary LU *x* and XRF-active primary LU *z*, with the exception that the switchpoint recognizes the session as an XRF session and collects the session-state data required for subsequent resynchronization of the session state when takeover occurs. LU *y* learns via the shared data that a backup session is required with LU *x* and sends an INIT request to SSCP *b* indicating that it wants to establish an XRF-backup session with LU *x*. SCCP *b* then uses its session with SSCP *a* to perform the normal cross-domain session setup actions; these actions are not shown in the figure.



Legend:

BF = Boundary Function (with XRF Switchpoint)
IMS = Information Management System
LU = Logical Unit
SSCP = System Services Control Point
----- = Session

Figure 103. Initiating an XRF-Backup Session

The following steps are involved in establishing the XRF sessions and in subsequently causing the takeover of a failed XRF-active session by its corresponding XRF-backup session.

1 LU *z* initiates an XRF-active session to LU *x* via normal cross-domain session services. (When a primary LU wants to provide XRF support for a secondary LU, the PLU includes a **session correlation identifier** in the BIND request it sends to initiate the backup session. The XRF-active logical unit assigns the identifier when it builds its BIND, and the XRF switchpoint uses the identifier to associate the subsequent XRF-backup BIND with the corresponding XRF-active session.)

2 LU *y* learns, via an implementation-defined signal from node C to node B (using the shared database), that an XRF-backup session is required and initiates the XRF-backup session stage between LU *y* and the boundary function for LU *x*. The other session stage, between the boundary function and LU *x*, is unaffected by the initiation of the backup session, and LU *x* is unaware of the existence of the backup session.

After a failure of the XRF-active session:

3 Recognizing the failure of LU *z*, the XRF facility associated with the backup system causes LU *y* (a) to send a SWITCH request to the boundary function of LU *x* requesting that the XRF-backup session become XRF-active and (b) to receive the current session-state data from the boundary function. The boundary function switchpoint is responsible for switching traffic to the XRF-backup session. The boundary function sends a positive acknowledgment containing the current session-state data to the XRF-backup PLU, making it the new XRF-active PLU, and sends an UNBIND to the old XRF-active PLU to end the old session.

4 The boundary function notifies the SSCP that owns the SLU (SSCP *a*) that an XRF switch has occurred. The SSCP then updates its session information to reflect that the new session pair has become XRF-active.

LU 6.2 Protocols

The rapid growth in recent years of networks of small and mid-range distributed processors has increased the demand for networking protocols that reduce reliance solely on mainframe-based services. The ways in which the type 2.1 node has evolved in this direction, from XID negotiation and adaptive session-level pacing to the directory and topology services of the APPN node control point, have been discussed in prior sections of this publication. Just as the type 2.1 node has been the focus for developing peer protocols in node architecture, the type 6.2 logical unit has been the focus for developing peer protocols in the logical unit. LU 6.2, which provides ***advanced program-to-program communication*** (APPC), is better suited to the needs of distributed processing than are earlier LU types.

Functions of LU 6.2

Examples of LU 6.2's peer-oriented capabilities have been cited previously. The ability of type 6.2 LUs in LEN and APPN nodes to set up sessions independently of SSCPs and to take on either the primary or secondary role on a session, are two such examples. However, an even more important peer-oriented aspect of LU 6.2 is its symmetric appearance at the two ends of a session—in terms of both the protocols governing the exchanges between the partners and the application program interface presented to the using transaction programs. These are the functions of LU 6.2 that are more apparent to an end user. They fall into various categories, such as interface verbs and options, connectivity, and security. These functions are discussed in the sections to follow.

The entire range of LU 6.2 functions is not implemented by all products. Compatible subsets of functions have been defined that reduce development costs by allowing some products to implement more limited functions than others. The functions are grouped into a ***base set***, which is implemented by all products, and various ***option sets***. Product developers can choose to implement a given option set or not, but only *whole* option sets can be implemented.

Various prerequisite relationships exist among the option sets, as illustrated in Figure 104. In this illustrative structure, option sets A and D have no dependent option sets. Option set E, however, is a prerequisite for F, and both B and C are prerequisites for G. This is the general way options are structured in LU 6.2 and other SNA architectures. For a detailed discussion of the LU 6.2 option sets, see *SNA Transaction Programmer's Reference Manual for LU Type 6.2*.

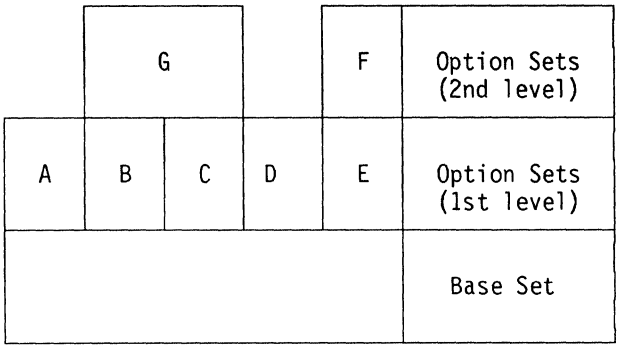


Figure 104. Concept of Base and Option Sets

CPI-C and LU 6.2 Protocol Boundary

SNA defines a generic ***application program interface*** (API) for developers of transaction programs called the LU 6.2 protocol boundary. It is *generic* in that implementing products can adapt the functional (semantic) definition of the API to their own programming language (syntactic) environments. Each product publication describes the relationship of its own syntactical representation of the LU 6.2 API to that defined by SNA.

The LU 6.2 ***protocol boundary*** is a conceptual dividing line between a transaction program (TP) and the type 6.2 LU, whereby the TP is effectively shielded from the details of the underlying implementation within the LU. As shown in Figure 105, the LU 6.2 protocol boundary shields transaction programs from the underlying details of session protocols. Those details are handled by presentation services and the lower layers of the LU.

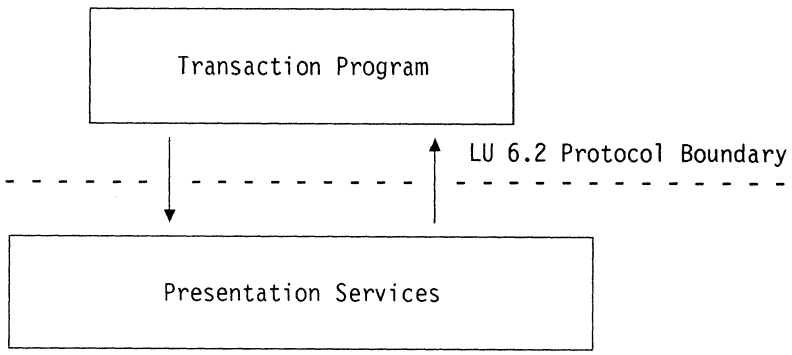


Figure 105. LU 6.2 Protocol Boundary

The LU 6.2 API is defined as a set of ***verbs*** used by a TP to establish connection and then to communicate with a partner TP. This level of API definition is unique to the type 6.2 LU. The Common Programming Interface for Communications (CPI-C) is an even higher-level API, which implementing products can map to their underlying LU 6.2 and other APIs.

Common Programming Interface for Communications

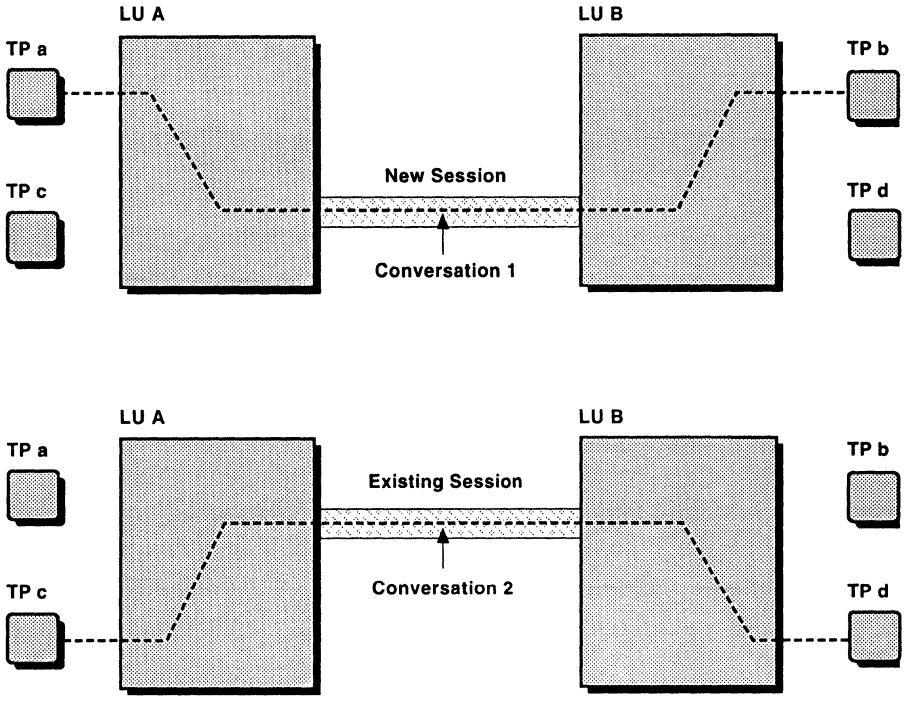
The ***Common Programming Interface for Communications (CPI-C)*** allows applications to be developed easily, without the developer having to learn the details of the LU 6.2 API for each product platform that is used. Using CPI-C, portable applications may be written; portable applications can be moved from one product platform to another with few if any changes. CPI-C applications can protect an organization's investment in application programming because CPI-C applications are designed to be run on any product platform supporting it. For more information on CPI-C, refer to *Systems Application Architecture: Common Programming Interface Communications Reference* and *Common Programming Interface for Communications Specification.*

Conversations

A ***conversation*** is a logical connection between a pair of transaction programs for serially sharing a session between type 6.2 logical units. Just as a session is a temporary logical connection between two network accessible units, a conversation is a temporary logical connection between two transaction programs. While a conversation is allocated, it has exclusive use of an LU-LU session as delimited by a distinct set of brackets.

Once a conversation is allocated to a session, a send-receive relationship is established between the programs that are connected on the conversation. The programs may now exchange data either alternating sending (half-duplex) or both sending at the same time (full-duplex). Programs need not be concerned with the protocols of the underlying session, only with the conversation protocols.

Successive conversations can use the same session, thus minimizing the allocation and initialization of session resources in the LU. Figure 106 illustrates two pairs of transaction programs communicating on two successive conversations, with each conversation using the same session.



Legend:

LU = Logical Unit
TP = Transaction Program

Figure 106. Conversations

In Figure 106, transaction programs *a* and *b* allocate a conversation between them, and LUs *a* and *b* activate a session for the conversation. The TPs then execute their transactions. When the conversation is complete, the TPs deallocate the conversation between them, but the LUs do not terminate the session. Then, when conversation *2* is allocated between TPs *c* and *d*, a new session need not be activated for the conversation.

Conversation Types

LU 6.2 defines two types of conversation, corresponding to two levels of the protocol boundary: the basic conversation and the mapped conversation. The conversation type is specified by the allocating transaction program.

The ***basic conversation*** protocol boundary provides a low-level interface suitable for service transaction programs or application transaction programs that require the more privileged functions available on basic conversations. Transaction programs using basic conversations are required to manage the details of the data stream exchanged on the conversation, and are responsible for more of their own error recovery.

The ***mapped conversation*** protocol boundary is intended for application transaction programs. It enables the programs to exchange messages of arbitrary format, regardless of the underlying data stream. System-defined or user-defined *mappers* can perform data transformation for the application transaction programs.

Figure 107 compares the basic and mapped levels of the conversation protocol boundary.

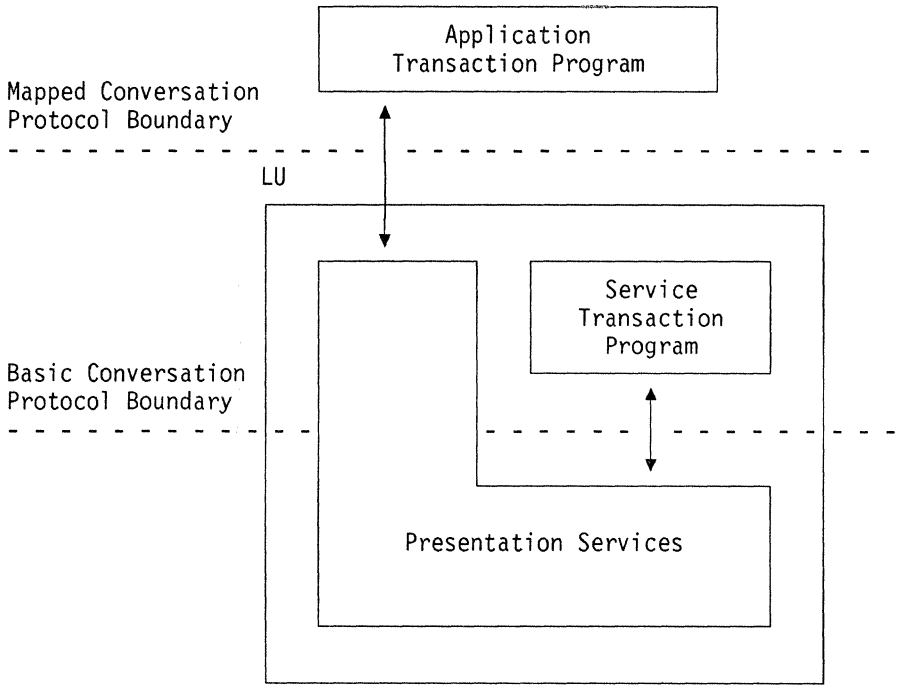


Figure 107. Mapped and Basic Conversation Protocol Boundaries

Conversation Verbs

The LU 6.2 protocol boundary is specified by a set of ***verbs*** that are described in the *SNA Transaction Programmer's Reference Manual for LU Type 6.2*. Transaction programs issue verbs to communicate across the LU 6.2 protocol boundary. For example, a transaction program (TP) requests a conversation with a remote TP by issuing an ALLOCATE verb. The LU for the requesting TP sends the LU for the remote TP an Attach request naming the remote TP, to invoke the TP and initiate the conversation. The two transaction programs are then connected by a conversation and can communicate with one another.

Verbs are issued with accompanying ***parameters*** specifying information needed for execution of the verb. For example, the ALLOCATE verb is accompanied by parameters specifying the partner transaction program name, the partner LU name, the mode name, the synchronization level, and security parameters. (LU 6.2 synchronization level and security are discussed in sections to follow.)

As noted earlier, the intent of the LU 6.2 API is to provide a *semantic* rather than a *syntactic* foundation for transaction program development. Whether or not products implement the exact syntax of the LU 6.2 verbs within their APIs, it is important that products provide the intended functions of the verbs. Then, despite the syntactic differences between their APIs, individual products can ensure that they are functionally compatible by mapping their individual APIs to the LU 6.2 protocols. As long as the product verbs and parameters map functionally to the LU 6.2 protocols, compatibility is ensured.

LU 6.2 verbs fall into four categories:

1. Basic conversation verbs
2. Mapped conversation verbs

3. Type-independent verbs
4. Control-operator verbs.

Basic conversation verbs are intended for use by service transaction programs using basic conversations. They provide the complete functionality of the LU 6.2 API. Because programs written to handle basic conversations must consider the structure of the underlying data stream, some parameters accompanying basic conversation verbs refer to the GDS variable. For example, a parameter provided by the RECEIVE_AND_WAIT and RECEIVE_IMMEDIATE verbs allow the issuing transaction program to receive data one logical record at a time (as defined by the LL field of the GDS variable), or one receive buffer at a time.

Mapped conversation verbs are intended for use by application transaction programs using mapped conversations. They shield the transaction programmer from the complexities of the underlying data stream. For example, the RECEIVE_AND_WAIT and RECEIVE_IMMEDIATE verbs assume that the LL and ID fields of the GDS variable have been removed, and that only the data portion is moved into the receive buffer. In addition, they can provide a MAP_NAME parameter that specifies the format of incoming data, enabling it to be mapped by system-defined or user-defined mappers for presentation to the transaction program.

Type-independent conversation verbs are intended for use with both basic and mapped conversations. Their functions do not reflect the type of conversations on which they are issued. For example, the ***WAIT*** verb waits for posting to occur on any of a list of conversations for which asynchronous receives have previously been issued.

Control-operator verbs define the protocol boundary for a control-operator transaction program that performs definition and control functions for LU 6.2, possibly in concert with a human operator. The functions pertain to the definition and control of logical unit and session attributes. The logical unit attributes include their network-qualified names, parallel-session support, security information, and map names. The session attributes include pacing counts, maximum RU sizes, synchronization level, and security information. The concepts of parallel sessions, synchronization level, and LU 6.2 security are discussed under "LU 6.2 Sessions."

Conversation States

The LU 6.2 architecture conceives of each end of an LU 6.2 conversation as a finite-state machine. A ***finite-state machine*** (FSM) is a combination of processing and memory, the memory portion of which exists in one of a finite number of possible states. The ***state*** of the FSM determines the actions that the FSM can take at any given time.

Half-Duplex

In the case of a half-duplex LU 6.2 conversation, the state of a conversation partner determines the verbs that the partner can issue. Examples of conversation states include receive state and send state. A conversation partner can only receive data on the conversation (issue receive-type verbs) in ***receive state***, and only send data on the conversation (issue send-type verbs) in ***send state***. The states of multiple conversations, allocated on multiple sessions supported by the same LU, however, can vary. One conversation can be in receive state and another in send state, simultaneously.

The two ends of a conversation are normally in different, but complementary states. If one end is in send state, the other is in receive state. When the sending program finishes sending data, it transfers control to the receiving program. The two programs then reverse their send and receive states, and user data flows in the opposite direction.

Note that this message exchange discipline is reminiscent of the half-duplex flip-flop data flow control protocol. As mentioned under "Send and Receive Mode Protocols," a session between type 6.2 logical units underlying a conversation between two transaction programs uses the half-duplex flip-flop protocol or the full-duplex protocol.

Full-Duplex

On a session using full-duplex send and receive mode protocol, both session partners can send data simultaneously. A conversation that permits simultaneous transmission of independent records in both directions is a ***full-duplex conversation***. In order to provide a full-duplex conversation where send and receive operations occur simultaneously, ***nonblocking*** verbs are used. Nonblocking verbs are those that need not be complete when control is returned to the issuing application transaction program. This allows both a nonblocking send request and a nonblocking receive request to be outstanding at the same time. The send and receive flows of a conversation can be completely independent, in that the data being sent by one transaction program can be totally independent of the data being received by the same program. The application program can choose to relate the information on its send and receive flows or to use the send and receive flows independently.

A full-duplex LU is an LU that contains the necessary function to support both full-duplex and half-duplex conversations on a particular session. A single LU 6.2 session may support, sequentially, both full-duplex and half-duplex conversations. LU 6.2 is full-duplex at the session level when full-duplex LU support is negotiated on the BIND RU.

Nonblocking Support

A ***blocking verb*** performs a synchronous operation. A blocking operation reaches a point where it cannot proceed because a resource it needs is unavailable; it waits for the resource to become available. In doing so, control is voluntarily given to a lower-priority process. When the TP blocks waiting for the completion of an issued verb, there is a loss of efficiency for the TP and the potential danger of deadlock exists.

A ***nonblocking verb*** performs an asynchronous operation. A nonblocking operation does not voluntarily give control to a lower-priority process. Instead, the LU will return to the TP with the indication that the operation is incomplete. The operation proceeds in an asynchronous manner and the TP is eventually notified of its completion. The TP may continue processing other conversations or performing other functions while waiting for the nonblocking operation to complete. A nonblocking operation that is not yet completed but from which control has already been returned to the TP is termed an ***outstanding operation***. Nonblocking is a prerequisite for both full-duplex conversations and expedited data. Nonblocking can also be used by a TP running half-duplex conversations to handle all its conversations concurrently with greater efficiency.

Expedited Data

Expedited data allows user control information to be exchanged. Expedited data can be up to 86 bytes and is neither parsed nor formatted by the LU, nor is it segmented by the network. Expedited data has no logical field lengths or GDS variables, nor is it partially buffered by the LU or partially delivered by the transaction program.

LU 6.2 Sessions

There are many considerations for sessions between type 6.2 LUs that are unique to that LU type. Such considerations include multiple sessions, pooling, selection, session limits, and polarity.

Multiple Sessions

Only one pair of transaction programs at a time can use a particular session. To enable concurrent communication between multiple pairs of transaction programs, the type 6.2 LU supports multiple concurrent sessions. This capability enhances efficiency within the node because fewer LUs need be allocated. Applications for multiple-session capability include multiprogrammed processors and multiple-user workstations.

When multiple sessions exist between the same pair of LUs, they are called ***parallel LU-LU sessions***. The ability to support parallel sessions is optional for type 6.2 LUs. LUs that can support parallel sessions are called ***parallel-session LUs***, and those that cannot are called ***single-session LUs***. A single-session LU can support more than one session concurrently, but each must be with a different partner LU. The protocols for sessions between two parallel-session LUs are different from those between two single-session LUs, or between a parallel-session LU and a single-session LU. Therefore, a single session between two parallel-session LUs is still called a ***parallel session***.

Session Pools

When a transaction program deallocates a conversation, the underlying session is not terminated by the LU. Instead, the LU places the session in a ***session pool*** containing active sessions ready to be allocated to new conversations. The efficiency gained from avoiding the overhead of repeated session initiation and termination for single-session LUs is proportionately greater for parallel-session LUs.

A session pool in a type 6.2 LU is partitioned by partner LU name and mode name. A group of sessions pertaining to the same partner LU and mode pair is referred to as an ***LU-mode session group***. When a conversation is allocated, the LU checks whether an available session already exists in the session pool for the specified partner LU name and mode name. If so, the conversation is allocated to it. If no session is left in the LU-mode session group, a new session is activated for the conversation. The activation of a new session is subject, however, to the session limit placed on the group.

Session Selection

Transaction programs do not control the attributes of sessions directly. They are able to control session attributes indirectly, however, by use of the mode name. The ALLOCATE verb contains a MODE_NAME parameter, which maps to a set of session characteristics.

For LU 6.2, a mode determines the characteristics of a single session, or a group of parallel sessions with one partner LU. A maximum number of sessions is allowed for a given partner LU and mode pair (the LU-mode session group). That maximum is set by the ***INITIALIZE_SESSION_LIMITS*** control operator verb, which is issued before any conversations are allocated specifying that mode.

Figure 108 illustrates single and parallel sessions within LU-mode session groups. In this example, LU a and LU c have parallel-session capability, while LU b supports only a single session. Sessions within an LU-mode session group can be activated by either partner LU.

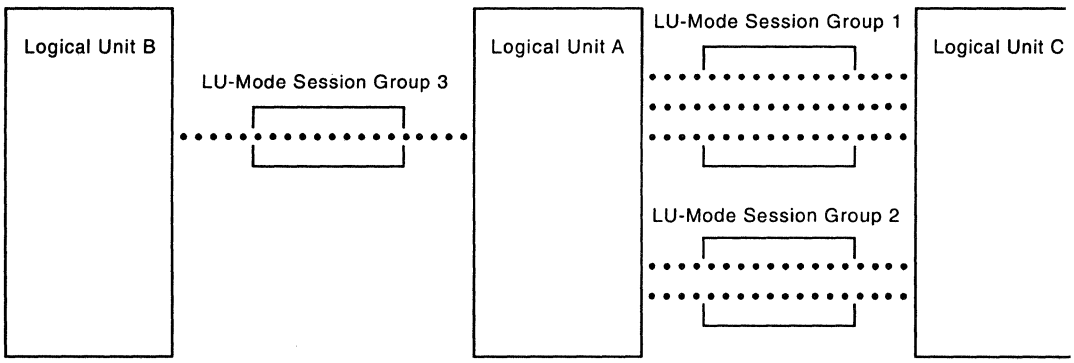


Figure 108. Single and Parallel Sessions

Session Limits

To facilitate storage allocation at the LU, limits are placed on the numbers of sessions in the LU. The limits are specified through a combination of the control operator verbs and system definition. The ***total LU session limit*** is the maximum number of sessions that can be active at one time at the LU. The ***LU-mode session limit*** is the maximum number of sessions that can be active at one time at the LU for an LU-mode session group. The ***automatic activation limit*** is the maximum number of sessions that are automatically activated for a particular LU-mode session group before conversations are allocated. Automatically activated sessions constitute the ***initial session pool***. Finally, the ***local-LU minimum contention-winner limit*** and the ***partner-LU minimum contention-winner limit*** determine the minimum number of sessions within the LU-mode session group for which the local LU and partner LU, respectively, will be the contention winner.

Contention Polarity

When the LUs at each end of a session both try to start a conversation at the same time, a *contention* situation is created. To resolve the contention, each session is assigned a ***contention polarity*** that determines beforehand which LU will be the *contention winner* on the session. A control-operator transaction program can use control operator verbs to specify the minimum numbers of contention-winner and contention-loser sessions at the LU.

When a transaction program allocates a conversation, the LU allocates it on a contention-winner session for the LU-mode pair. If the maximum number of contention-winner sessions has been reached, the LU attempts to allocate it on a contention-loser session.

Communicating on Conversations

Figure 109 on page 233 depicts a half-duplex conversation message flow, from allocation to deallocation.

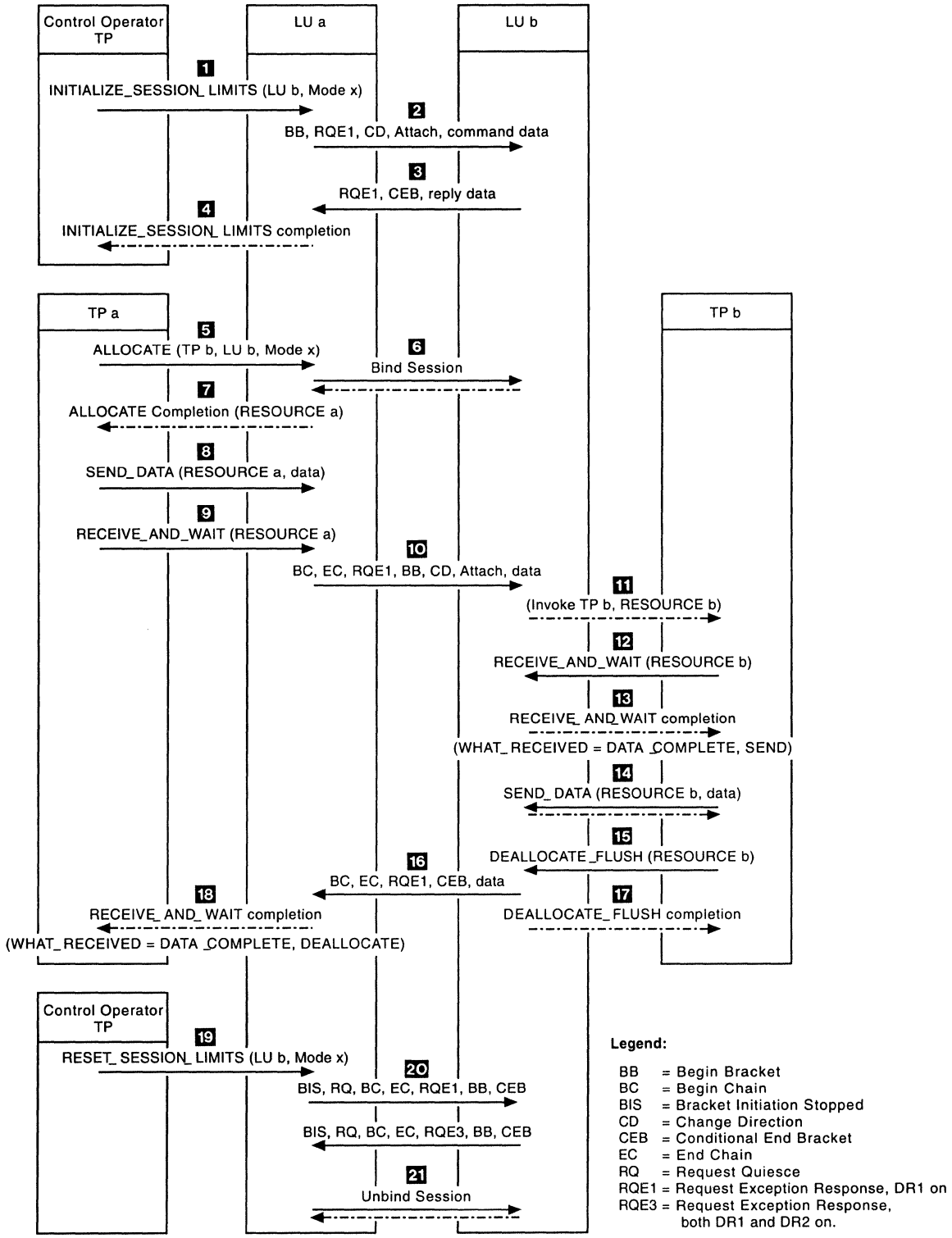


Figure 109. Communicating on a Half-Duplex Conversation

The following comments correspond to the numbers in Figure 109.

1 The control operator TP issues an INITIALIZE_SESSION_LIMITS verb, telling LU *a* to raise session limits for the LU-mode session group represented by LU *b* and mode *x*. The control operator TP waits on completion of the verb.

2 LU *a* allocates session pool storage for the potential sessions and sends an Attach to LU *b* to invoke the Change Number of Sessions service transaction program (CNOS STP) in LU *b*. The Attach begins a conversation between the control operator TP and the CNOS STP. The command data in the message is then passed on the conversation to the CNOS STP.

3 LU *b* allocates session pool storage to support the LU-mode session group. LU *b* then responds affirmatively to LU *a*. The conditional end bracket (CEB) ends the conversation.

4 LU *a* completes the INITIALIZE_SESSION_LIMITS verb issued by the control operator TP.

5 TP *a* issues an ALLOCATE verb telling LU *a* to allocate a conversation to TP *b* at LU *b* with mode *x*. TP *a* waits on completion of the ALLOCATE.

6 LU *a* activates a session with LU *b*. An Attach for the conversation is not yet sent, however. In the interest of efficiency, data does not normally flow on a conversation unless certain conditions are met. The queuing of enough data for the partner to exceed the user-defined minimum RU size, or a change in conversation state are two such conditions. (A FLUSH verb issued by a TP, for example, will cause queued data to be sent to the partner immediately.)

7 LU *a* responds affirmatively to TP *a* by completing the ALLOCATE and passing a ***resource ID*** for the conversation to TP *a* that is unique to LU *a*. TP *a* will use the resource ID to identify the conversation to LU *a* when issuing conversation verbs for that conversation.

8 Because TP *a* allocated the conversation, TP *a* is the first speaker on the conversation. TP *a* issues a SEND_DATA verb transferring end-user data to LU *a* for transmission to TP *b*.

9 Because TP *a* has completed sending data, it reverses the send/receive state of the conversation by issuing a RECEIVE_AND_WAIT. This places TP *a* in a wait state.

10 The change in the conversation from send to receive state causes LU *a* to reverse the send/receive mode of the session and transmit the queued data to LU *b*. LU *a* sends a message to LU *b* that includes an Attach, telling LU *b* to invoke TP *b*, and to pass the end-user data to LU *b* once it is invoked.

11 LU *b* invokes TP *b*, passing a resource ID for the conversation to TP *b* that is unique to LU *b*. TP *b* will use the resource ID to identify the conversation to LU *b* when issuing conversation verbs for that conversation.

12 TP *b* then issues a RECEIVE_AND_WAIT verb, which places TP *b* in a wait state.

13 LU *b* passes the end-user data to TP *b*, thus completing TP *b*'s outstanding RECEIVE_AND_WAIT. The WHAT_RECEIVED parameter of the RECEIVE_AND_WAIT verb tells TP *b* two things: first, that data has been transferred to its input buffer, and second, that it has been placed in send state by TP *a*.

14 After processing the data, TP *b* issues a SEND_DATA verb (assuming TP *b* has something to send). LU *b* queues the data and completes the SEND_DATA verb. The message exchange sequence illustrated thus far can be continually repeated. To reverse the conversation state and cause the queued data to be transmitted to TP *a*, TP *b* would now issue a receive type verb.

15 In this illustration, however, TP *b* completes the conversation by issuing a DEALLOCATE_FLUSH verb. The DEALLOCATE-FLUSH causes the queued data to be sent before the conversation is deallocated.

16 LU *b* transmits the data to LU *a* with a CEB, thus ending the conversation,

17 LU *b* then completes TP *b*'s outstanding DEALLOCATE_FLUSH verb. After the verb is completed, TP *b* might shut down if it has no further work to do.

18 LU *a* then completes TP *a*'s outstanding RECEIVE_AND_WAIT verb. The WHAT_RECEIVED parameter tells TP *a* two things: first, that data has been transferred to its input buffer, and second, that the conversation has been deallocated by TP *b*. When TP *a* learns that the conversation has been deallocated, it might shut down if it has no further work to do. Whether it does or not, however, LU *a* does not terminate the session with LU *b*. The session might be reused by subsequent conversations.

19 When sessions are no longer needed between LUs *a* and *b*, the control operator TP issues a RESET_SESSION_LIMITS verb, which lowers the LU *a*'s session limit to 0 for the LU-mode session group represented by LU *b* and mode *x*.

20 LU *a* sends a one-way bracket to LU *b* indicating that LU *a* wishes to terminate the session with LU *b* for mode *x*. LU *b* returns a one-way bracket to LU *a* in acknowledgment.

21 LU *a* and LU *b* deactivate the session by exchanging an UNBIND request and response.

Note that even though only a single conversation is depicted in the figure, multiple concurrent conversations could exist between the LUs. If the LUs are parallel-session capable, and the specified session limit is greater than 1, then parallel sessions can be activated. Each session would support a separate pair of TPs communicating on a conversation.

It is important for system designers to understand the implications of conversation allocation and deallocation for system efficiency and resource utilization. Between the time a conversation is allocated and the time it is deallocated, the conversation has exclusive use of its underlying session. If the communication between a pair of TPs is characterized by intermittent exchanges, then a conversation should be allocated and deallocated for each exchange. A conversation allowed to remain in a wait state represents wasted resources for both the conversation and its underlying session, and sessions will be activated unnecessarily for conversations between other pairs of TPs at the same LU pair. On the other hand, if the communication is characterized by continual exchanges, then a conversation should remain allocated for the duration of the conversation traffic. Otherwise, continually allocated and deallocated conversations represent unnecessary overhead for the system, because the session will not be freed for any significant period of time for use by other TP pairs.

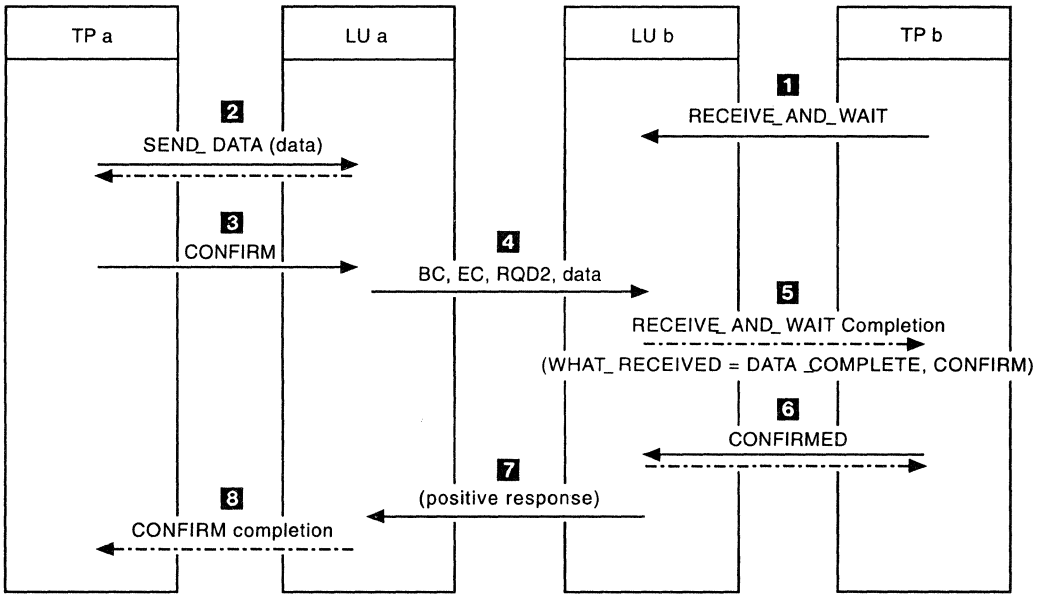
Synchronization Processing

While processing transactions, transaction programs in different locations need to keep their resources synchronized. A banking transaction, for example, might need to be posted in the processors at both ends of a conversation. To aid in resource synchronization, LU 6.2 defines two levels of synchronization processing capability: a base level known as confirmation processing, and a higher level known as sync point processing. When a transaction program allocates a conversation, it indicates the ***synchronization level*** of the conversation in the SYNC_LEVEL parameter of the ALLOCATE verb.

Confirmation Processing

Confirmation processing enables a transaction program (TP) to solicit acknowledgment from a partner program of a message sent to the partner program. It applies to half-duplex conversations. It is invoked when a conversation is allocated with a synchronization level of CONFIRM. The TP requesting confirmation does so by issuing a ***CONFIRM*** verb, and the partner TP acknowledges the request by issuing a ***CONFIRMED*** verb. In processing the CONFIRM and CONFIRMED verbs, the LUs build and exchange message units that accomplish the confirmation protocol.[13] When the LU message exchange is complete, the originating LU completes the CONFIRM verb with a return code indicating to the requesting TP whether or not CONFIRMED was received.

Figure 110 on page 236 shows a sample confirmation processing flow between transaction programs (TPs) *a* and *b*.



Legend:

BC = Begin Chain
EC = End Chain
RQD2 = Request Definite Response, DR2 on

Figure 110. Confirmation Processing

The following comments correspond to the numbers in Figure 110.

1 TP *b* issues RECEIVE_AND_WAIT, which places TP *b* in a wait state.

2 TP *a* issues SEND_DATA, transferring end-user data to LU *a* for transmission to TP *b*. LU *a* queues the data and completes the verb.

3 TP *a* issues CONFIRM, requesting confirmation of the last message sent. Note that after issuing CONFIRM, TP *a* does ***not*** issue a RECEIVE_AND_WAIT in order to receive acknowledgment. Because confirmation processing does not require the sending and receiving of confirmation messages by transaction programs, neither does it require them to change their send/receive states.

[13] The LU uses the DR2 indicator in the RH for the confirmation protocol.

4. LU *a* creates a confirmation request message with definite response requested and transmits it to LU *b* along with the end-user data.
5. LU *b* passes the data to TP *b*, thus completing TP *b*'s outstanding RECEIVE_AND_WAIT. The WHAT_RECEIVED parameter indicates that data has been transferred to TP *b*'s receive buffer and that TP *a* has requested confirmation of receipt by TP *b*.
6. After processing the data, TP *b* issues a CONFIRMED verb, acknowledging receipt of TP *a*'s message. LU *b* then completes the verb.
7. LU *b* creates a confirmation acknowledgment and transmits it to LU *a*.
8. LU *a* completes TP *a*'s outstanding CONFIRM, thus notifying TP *a* of TP *b*'s acknowledgment.

The action to take upon receiving the acknowledgment to a CONFIRM is entirely application-dependent. Any action necessitated by the failure of a CONFIRM request is the responsibility of the transaction programs, not the LUs. For example, if a temporary failure of transmission facilities resulted in a negative response to CONFIRM, the sending program would most likely retry the SEND_DATA. The partner transaction program would cooperate in this recovery procedure by, perhaps, checking user-defined sequence numbers so as not to reprocess the retransmitted data.

Different kinds of transactions can have different implications for resource synchronization. For example, the transmission of account balances for informational purposes might not require confirmation, whereas notification of a completed monetary transaction such as a cash withdrawal might require confirmation. Because transmitting and processing confirmation messages entails some overhead for the LUs, transaction programmers should be selective as to which kinds of transactions should be confirmed and which should not.

Sync Point Processing

Sync point processing enables transaction programs that are processing a distributed transaction to synchronize their resources at user-specified points during the transaction called ***synchronization points***, or ***sync points***. A ***distributed transaction*** involves the cooperative execution of multiple programs at multiple locations to achieve some user-defined processing function. Distributed transactions are made possible by the multiple-session capability of the type 6.2 LU in all nodes. Whereas confirmation processing applies to only the two TPs at either end of a conversation, sync point processing links two or more TPs, and possibly other protected resources such as databases, at one or more LUs involved in a single distributed transaction.

LU 6.2 provides optional synchronization services to aid transaction programs in recovery from failures. The "sync point manager (SPM)" is the component of the node that provides synchronization services.

Sync point processing is invoked when a TP allocates a conversation with a synchronization level of SYNCPT. When either the allocating TP or the partner TP allocates subsequent conversations with the same synchronization level, the TPs and supporting LUs become participants in a distributed transaction. The participating TPs and LUs are said to be connected in an ***allocation tree***. The TP that allocates the first conversation becomes the *root* of the allocation tree. The conversations that interconnect the participating TPs form the *branches* of the tree.

Figure 111 illustrates an allocation tree involving four TPs and four LUs. The tree could have been constructed as follows: TP *a* allocated a conversation with TP *b* with a sync level of SYNCPT, and TP *b* allocated subsequent conversations with TPs *c* and *d*, also with sync levels of SYNCPT. In this example, TP *a* is the root of the tree. The three branches of the tree are represented by the conversations between TPs *a* and *b*, between TPs *b* and *c*, and between TPs *b* and *d*.

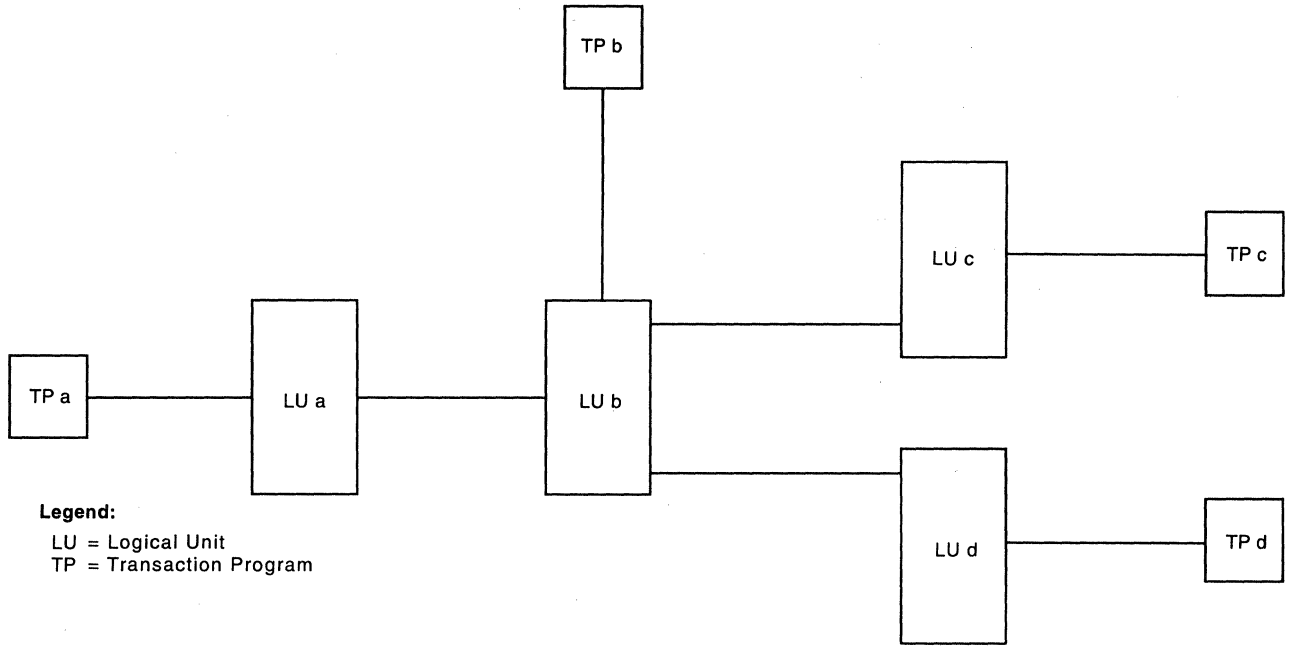


Figure 111. An Allocation Tree

The processing performed by transaction programs that takes place between sync points is called a ***logical unit of work*** (LUW). Sync point processing protects both conversation resources and implementation-defined resources, such as databases, that pertain to an LUW. Any changes to these resources are logged by the logical units so that the changes can be either ***committed*** (made permanent) if the LUW is completed as expected, or ***backed out*** (reversed) if any transaction program detects an error with the LUW.

A sync point is initiated when any transaction program in the allocation tree issues a ***SYNCPT*** verb. The LU supporting the TP initiated the sync point operation is called the ***initiator***; all other LUs processing the LUW are called ***agents***. The initiator and its agents propagate the sync point notification messages throughout the tree in a cascading fashion, similar to the propagation of Locate Search messages throughout an APPN network during a broadcast search. The message flow is described as forming a ***sync point tree***. The TP that issues a SYNCPT becomes the root of the sync point tree; the paths taken by the sync point notification messages form the branches of the tree. Because any TP in an allocation tree can become the root of a sync point tree, the two trees may or may not have the same root-branch structure.

A sync point involves verbs exchanged between the participating TPs and commands exchanged between the participating LUs. The TPs have different responsibilities from those of the LUs in processing a distributed transaction and in executing a sync point. The TPs are responsible for executing the transaction, deciding when to execute sync points, and recognizing conditions that merit

backout of an LUW. The LUs are responsible for controlling communication resources, as called for by the TPS.

The SPM component of the LU is responsible for coordinating the sync point operation over the different resources, including local resource such as databases and file systems.

A sync point involves a ***two-phase commit protocol***, which permits updates to one or more protected resources to be committed or backed out automatically. In the first phase, the sync point initiator sends a Prepare command, requesting that all agents vote by responding Request Commit or Backout, indicating whether the logical unit of work should be committed or backed out. All agents must vote to commit if the transaction is to be committed. When all the votes are collected, the second phase begins. In this phase, the initiator informs the agents to commit or back out. At various times, the sync point participants write state information to nonvolatile storage so that the protected resources can be resynchronized if any failures occur during the two-phase commit processing.

Presumed nothing is a variation of the two-phase commit protocol in which a sync point initiator force-writes logical-unit-of-work state information (to nonvolatile storage) before sending the Prepare command, and an agent force-writes a record of the end of the commit processing before acknowledging a Backout message. The initiator is always responsible for initiating recovery processing if any failure occurs during the two-phase commit processing.

Presumed abort is another variation of the two-phase commit protocol in which a sync point initiator need not force-write logical-unit-of-work state information before sending the Prepare command, and an agent need not force-write a backout record before acknowledging a Backout message. If a prepare record is found on its log after a crash, an agent initiates recovery processing with its initiator. If the initiator has no information about the transaction, it presumes that the transaction aborted and tells the subordinate to abort.

The partner LUs negotiate the two-phase commit protocol they support in the Exchange Log Name GDS variables they exchange at cold start or respecify the support at warm start (following a crash).

Figure 112 and Figure 113 illustrate verbs and commands issued by the TPs and LUs in a sync point tree during a successful sync point. In this example, the sync point tree is identical to the allocation tree in Figure 111.

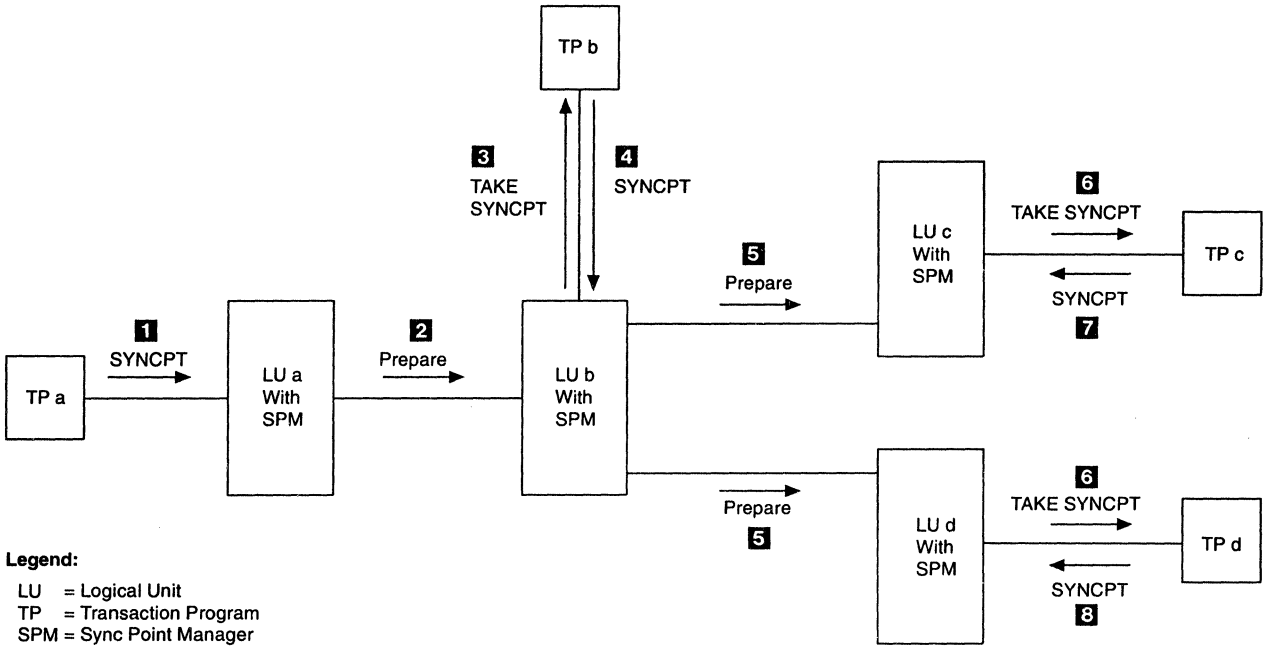


Figure 112. A Committed Sync Point Sequence: Part I

The following comments correspond to the numbers in Figure 112.

1. TP a begins a sync point by issuing a SYNCPT verb.
2. The initiator, LU a, then sends a Prepare command to the first agent, LU b, to tell it to prepare to commit its protected resources.
3. LU b signals its TP with a TAKE_SYNCPT indication set on completion of any receive-type verb, such as RECEIVE_AND_WAIT.
4. TP b issues SYNCPT.
5. Once TP B issues SYNCPT, LU b propagates the Prepare command to the other agents, LU c and LU d.
6. LUs c and d signal their TPs with a TAKE_SYNCPT indication set on completion of any receive-type verb, such as RECEIVE_AND_WAIT.
7. TP c issues SYNCPT.
8. TP d issues SYNCPT.

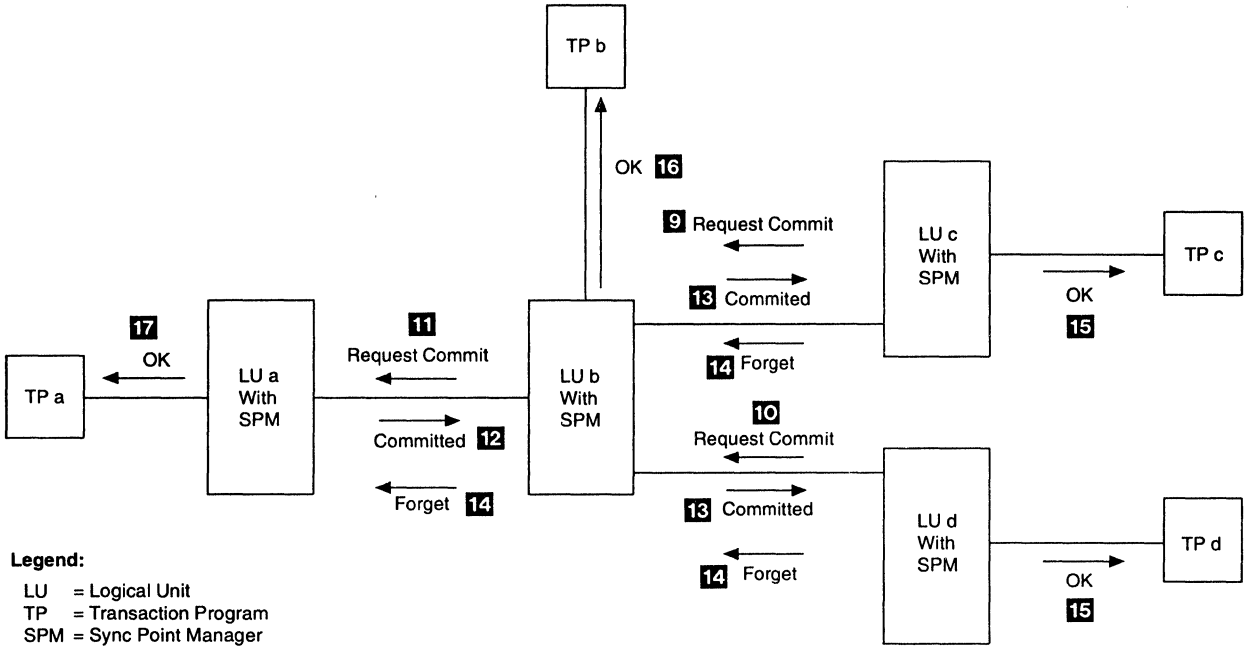


Figure 113. A Committed Sync Point Sequence: Part II

The following comments correspond to the numbers in Figure 113. Figure 113 assumes that the sequence in Figure 112 has taken place.

9 The agent, LU c, then sends a Request Commit command upstream to its initiator, LU b. A Request Commit indicates that an agent has prepared all its protected resources to be committed. (An agent may also send Forget, which indicates that the agent has made no resource changes for this LUW.) Each agent sends a Request Commit to the agent from which it received a Prepare command.

10 The agent, LU d, then sends a Request Commit command upstream to its initiator, LU b.

11 The agent, LU b, sends a Request Commit upstream to its initiator, LU a, indicating that LU b and its subordinate agents (LU c and LU d) are prepared. When the initiator, LU a, receives a Request Commit from LU b, that indicates that all agents are prepared to commit their resource changes.

12 LU a then sends a Committed command to LU b telling LU b to commit its resources.

13 LU b commits the local resources and propagates the Committed command to LUs c and d.

14 LUs c and d each issue Forget to LU b. LU b issues a Forget to LU a, indicating that the Committed was executed successfully. (When LU a received Forget from LU b, that indicates that LU b's subtree has Committed successfully.)

15 LU c sends OK to TP c in response to the SYNCPT issued in step 7 (from TP c to LU c) and LU d sends OK to TP d in response to the SYNCPT issued in step 8 (from TP d to LU d). This indicates that the sync point operation was successfully completed.

16 LU b sends OK to TP b in response to SYNCPT sent in step 1.

17 LU a sends OK to TP a in response to the SYNCPT issued in step 1 (from TP a to LU a).

In an unsuccessful sync point, at least one TP in a distributed transaction initiates a backout. After receiving a TAKE_SYNCPT indication on completion of a receive-type verb, a TP determines for some reason that it is necessary to back out the changes corresponding to the current LUW. It initiates the backout by issuing a ***BACKOUT*** verb. Figure 114 illustrates verbs and commands issued during an unsuccessful sync point from the point at which the TPs in Figure 113 respond to the TAKE_SYNCPT indication.

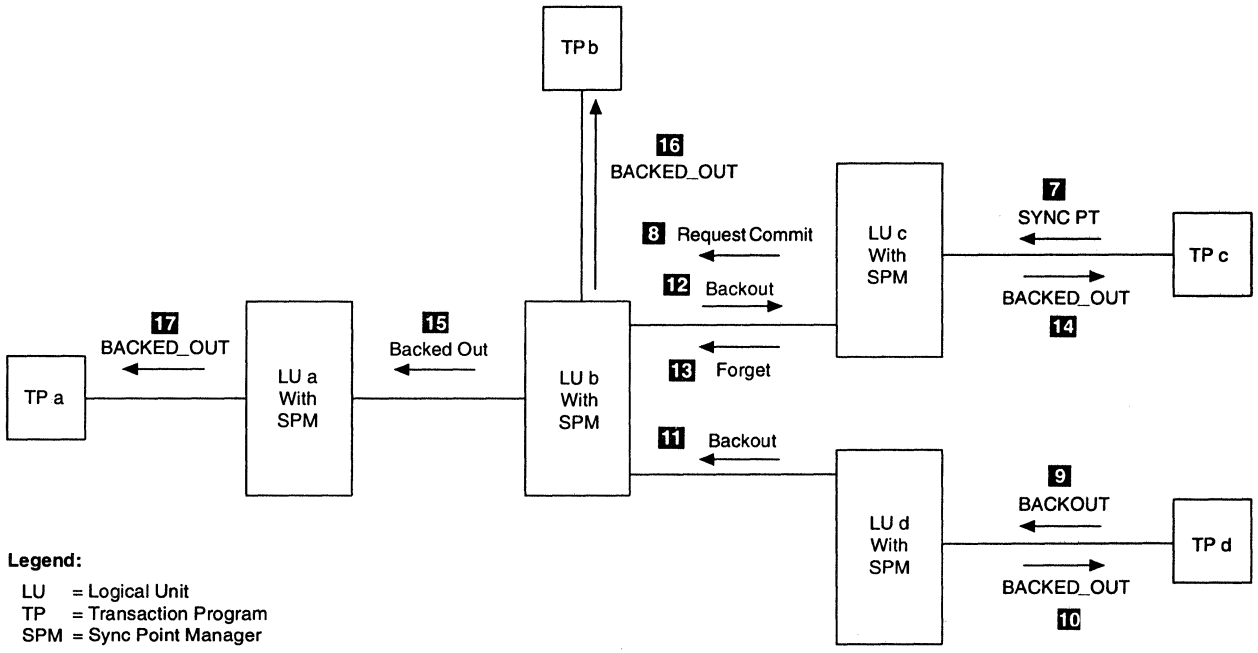


Figure 114. A Backed-Out Sync Point Sequence

The following comments correspond to the numbers in Figure 114. Figure 114 assumes that the sequence in Figure 112 has taken place through step 7.

7 TP c issues SYNCPT.

8 LU c issues a Request Commit command, indicating that its protected resources are prepared.

9 TP d determines that the current LUW must be backed out and issues a BACKOUT verb to LU d.

10 LU d backs out its resource changes for the current LUW and sends a return code of BACKED_OUT to TP d.

11 LU d issues a Backout command to LU b.

12 The Backout command is propagated throughout its sync point tree, from LU b to LU c. LUs b and c also back out their resource changes for the current LUW.

13 LU c issues a Forget to LU b.

14 LU c backs out its resources and sends a return code of BACKED_OUT to TP c in response to the Request Commit command sent in step 8.

15 LU b issues a return code of Backed Out to LU a.

16 LU b issues a return code of BACKED_OUT to TP b.

17 LU a issues a return code of BACKED_OUT to TP a, in order to complete TP a's outstanding SYNCPT verb (sent in Figure 112).

There are certain failures for which backout processing is ineffective in resynchronizing resources. Those failures include the loss of LUs, sessions, or conversations that interrupt sync point processing. If such a failure occurs, the ***resynchronization*** process restores protected resources to a consistent state. This is done by the ***resync*** service transaction program. When an appropriate session is available, the resync STP in one LU attaches its counterpart in another. The partner LUs then communicate to validate the integrity of their logs, determine the sync point status, and complete the sync point protocol. Resynchronization does not restart the failed transaction.

Refer to *SNA Transaction Programmer's Reference Manual for LU Type 6.2* for additional information about the conversation protocol boundary. Refer to *SNA LU 6.2 Reference—Peer Protocols* for additional information about LU 6.2 protocols.

Data Security Protocols

Data security has become an increasingly important issue for today's networks, which have become pathways for such confidential data as automated banking transactions. Logical units provide three functions to assist an installation in providing security: session-level cryptography, LU-LU verification, and end-user verification. For session-level cryptography and LU-LU verification, SNA makes use of the data encryption standard algorithm.

The Data Encryption Standard Algorithm

The ***data encryption standard*** (DES) algorithm is a public-domain algorithm used widely for security applications. It disguises confidential data by *enciphering* it under the control of a cryptography key. After it is enciphered, the data can then be *deciphered* into its original form under the same key. The ***cryptography key*** is an 8-byte value used by the DES algorithm in the enciphering or deciphering process. The security of the DES algorithm lies in the secrecy of the key, not the algorithm. The algorithm enciphers 8 bytes of *clear data* at a time. To encipher data longer than 8 bytes the algorithm can be applied repeatedly.

Figure 115 illustrates data enciphering and deciphering using the DES algorithm.

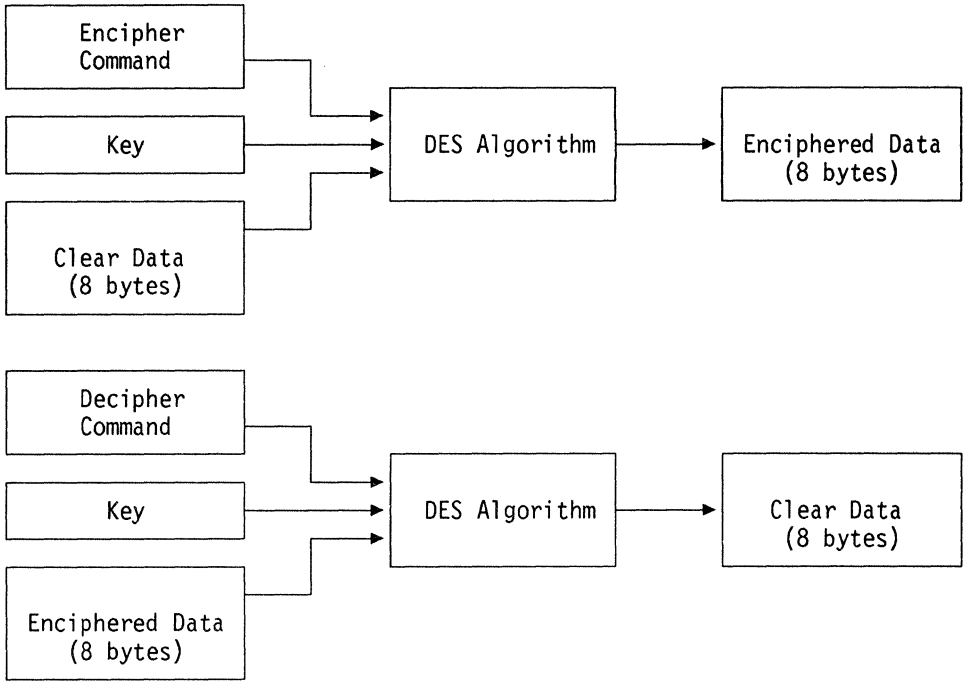


Figure 115. DES Encipherment and Decipherment

Session-Level Cryptography

Session-level cryptography disguises confidential session data in order to prevent its disclosure to unauthorized end users (persons or programs). Session-level cryptography disguises the data by *enciphering* it in a process that uses the DES algorithm. This approach to data security relies on the secrecy of the keys used in the cryptographic process.

Logical units can provide mandatory data cryptography, selective data cryptography, or no data cryptography. In a ***mandatory cryptographic session***, a logical unit enciphers all outbound data RUs and deciphers all inbound data RUs. In a ***selective cryptographic session***, a logical unit enciphers only the data RUs specified by the sending transaction program (TP). The sending LU signals that the RU is enciphered by setting the ***enciphered data*** (ED) indicator in the request header. By checking the EDI, the receiving LU can tell which RUs to decipher before passing them on to the receiving TP.

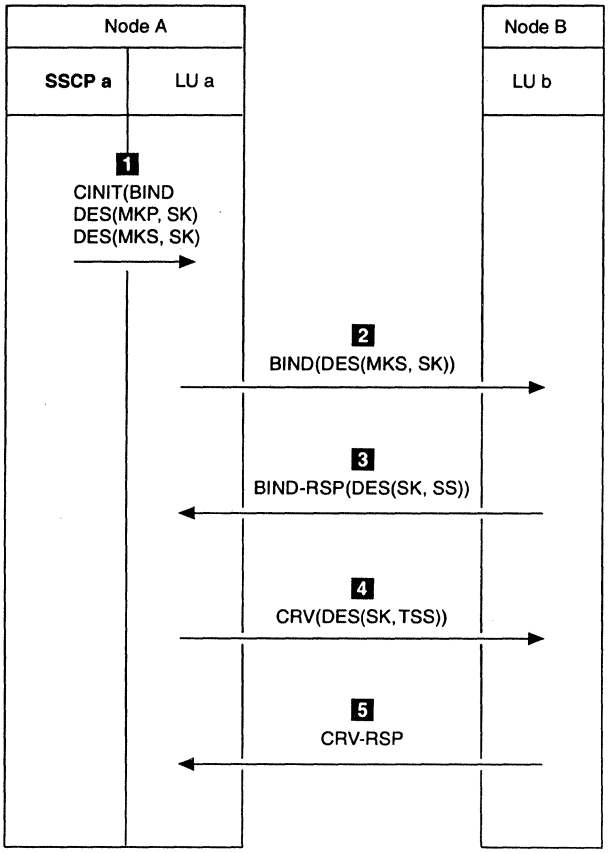
LUs prepare for data cryptography on a session by exchanging information contained in the CINIT, in the BIND, and in ***cryptography verification*** (CRV) messages. Session-level cryptography start-up flows establish two values: the session cryptography key and the session cryptography seed.

The ***session cryptography key*** (SK) is the cryptography key used to encipher and decipher data on the session. It is unique to the session and is randomly generated by the system services control point (SSCP) of the primary LU. Before transmitting the SK to an LU, the SSCP maintains the secrecy of the SK by first enciphering it, using the DES algorithm, under a master key known to the receiving LU.

A ***master key*** (MK) is a cryptography key used to encipher and decipher session keys. A master key is unique to an LU. It is stored both at the LU and at the LU's SSCP in an implementation-dependent manner, and can be changed only by network security personnel. When cross-domain sessions employ session-level cryptography, the SSCPs must exchange key-management messages.

The ***session cryptography seed*** (SS), an 8-byte value, is used in two ways. First, it is used as test data during session activation to ensure that the partner LUs are sharing the same session key. Second, it is used during the session by the cryptographic algorithm to help encipher data. Like the SK, it is unique to the session. It is randomly generated by the secondary LU and transmitted to the primary LU in the BIND response. The secrecy of the SS is maintained by transmitting it enciphered under the SK.

Figure 116 on page 246 illustrates the session-level cryptography start-up flows for a same-domain session between LUs *a* and *b*.



Legend:

BIND	= BIND Request Unit
BIND-RSP	= BIND Response Unit
CINIT	= Control Initiate
CRV	= Cryptography Verification
CRV-RSP	= Cryptography Verification Response
LU	= Logical Unit
MKP	= Master Key of PLU
MKS	= Master Key of SLU
SK	= Session Key
SS	= Session Seed
SSCP	= System Services Co
TSS	= Transformed Session Seed
DES(key,data)	= Encrypt 'data' using 'key' as encryption key

Figure 116. Session-Level Cryptography Start-up Flow

The following comments correspond to the numbers in Figure 116.

1 SSCP *a* supplies two versions of the SK to LU *a* in the CINIT, one enciphered under the MK for LU *a*, and the other enciphered under the MK for LU *b*.

2 LU *a* transmits the SK intended for LU *b*, the one enciphered under LU *b*'s MK, to LU *b* in the BIND request. After LU *b* deciphers the SK, both LUs have a copy of the SK for the session.

3 LU *b* randomly generates an SS and transmits it, enciphered under the SK, to LU *a* in the BIND response.

4 LU *a* deciphers the SS, then uses it as test data to verify that LU *a* is using the same session key as LU *b*. LU *a* first *transforms* the deciphered SS by inverting the first four bytes (turning 1's to 0's and 0's to 1's). It then sends the transformed SS (TSS) back to LU *b* in a CRV request, with the TSS enciphered under the SK.

5 LU *b* deciphers the TSS, transforms it back, and compares the result to the original SS. If they compare equal, then the test is successful and LU *b* returns a positive response to LU *a* in the CRV response. Otherwise, LU *b* terminates the session.

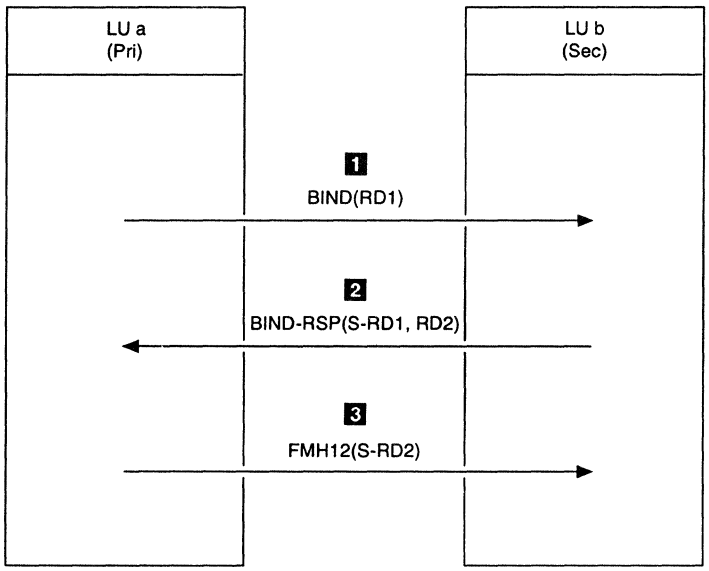
In a session employing session-level cryptography, transmission control performs the cryptographic functions. Before passing an RU to path control, transmission control enciphers the RU 8 bytes at a time. The RU is first padded to a length equal to a multiple of 8, then each 8-byte segment is enciphered, one segment at a time. This is done with a technique referred to as ***block chaining with cipher text feedback*** which proceeds as follows. First, the left-most 8 bytes of the RU are exclusive-ORed with the session seed (SS). The SS used in this manner is referred to as the ***initial chaining value***. Second, the result is enciphered under the session key using an 8-byte block chain algorithm that is in accordance with the DES algorithm. Finally, each subsequent 8-byte segment is exclusive-ORed with the previously enciphered segment. The deciphering process is simply the inverse of the enciphering process.

For additional information about cryptography, refer to *Data Security Through Cryptography*.

LU-LU Verification

LU-LU verification is a session-level security protocol intended to verify the identity of each LU to its partner. LUs verify their partners' passwords during the process of session activation. The passwords are called ***LU-LU passwords*** because they are established on a partner LU basis: one LU-LU password is established between each LU pair. LU-LU passwords are established by implementation-defined methods outside of SNA.

Figure 117 on page 248 illustrates the LU-LU verification message flow.



Legend:

BIND = BIND Request Unit
BIND-RSP = BIND Response Unit
S-RD = Security Reply Data
FMH12 = Function Management Header 12
LU = Logical Unit
RD = Random Data

Figure 117. LU-LU Verification Message Flow

The following comments correspond to the numbers in Figure 117.

1 The primary LU sends a BIND request that contains random data to the secondary LU. (A user-data subfield in the BIND contains this random data.)

The secondary LU uses its LU-LU session password to generate the security reply data.

2 The secondary LU then returns the generated security reply data to the primary LU in the BIND response. It also returns additional random data for the primary LU in the BIND response.

The primary LU uses its password to transform the random data that it sent in the BIND. The security reply data can be either the random data enciphered using the DES algorithm or a DES Message Authentication Code Value computed using both random data values and the secondary LU's name. Then it matches the resulting information with the security reply data that it received from the secondary LU in the BIND response. If the information matches, the primary LU knows that the secondary LU has the same LU-LU password as it does, and the secondary LU is authorized. If the information does not match, the primary LU terminates the session.

3 The primary LU then uses its LU-LU session password to generate another security reply data response. This response is sent in a security function management header (FMH-12). The response security reply data is either the random data, received in the BIND response and enciphered using the LU-LU session password, or a DES Message Authentication Code Value computed using the two random data values and the LU-LU session password.

The secondary LU uses the same procedure to transform its LU-LU session password and one or both random data values. Then it matches the resulting information with the security function management header. If the information matches, the secondary LU can now communicate over the session. If the information does not match, the secondary LU terminates the session.

End-User Verification

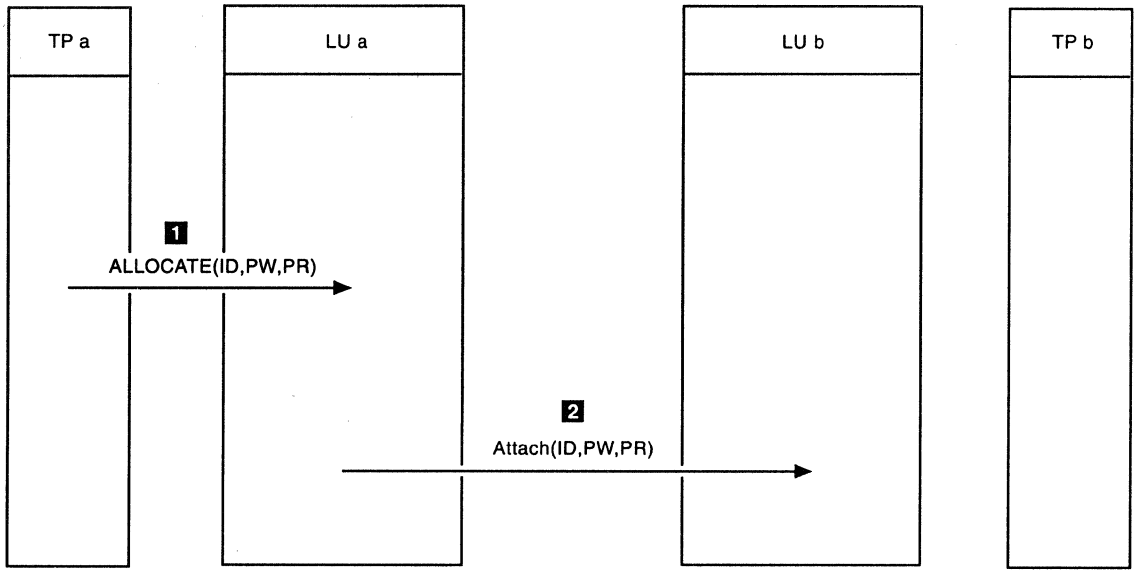
End-user verification is a conversation-level security protocol intended to verify the identity of a partner end-user. A type 6.2 LU can verify an end-user's password during the process of conversation allocation. In the context of an LU

6.2 conversation, the end user is the partner transaction program.

PGM parameter: The LU 6.2 conversation protocol boundary provides three security information fields in the PGM parameter of the ALLOCATE verb: password, user ID, and security profile. The PGM parameter specifies to use access security information that the local transaction program provides on this parameter. These fields are passed to the remote LU in the FMH-5. The user ID and password are verified by the remote LU. How they are verified and enforced is implementation-defined. If the end user has not supplied a correct user ID and password combination, the request is rejected and the target transaction program is not attached.

The ***security profile*** provides an additional authorization criterion. The profile is intended to be used by the remote LU to determine which remote programs and resources the local program is allowed to access.

Figure 118 illustrates the acquisition and transmission of the security fields used in end-user verification. This flow applies to the SECURITY(PGM) parameter of the ALLOCATE verb.



Legend:

ID = User ID
LU = Logical Unit
PR = Security Profile
PW = User Password
TP = Transaction Program

Figure 118. End-User Verification Message Flow

The following comments correspond to the numbers in Figure 118.

1. The end-user security fields are provided by TP *a* in the ALLOCATE verb.
2. The security fields are then mapped by LU *a* into the Attach function management header (FMH-5). LU *a* sends the FMH-5 to allocate the conversation, and LU *b* then uses the fields to verify the end user.

SAME parameter: The SAME parameter of the ALLOCATE verb specifies that access security information is to be sent, which will cause the remote LU to create an equivalent security environment. If the remote LU trusts the local LU to verify user IDs and passwords, then the user ID and security profile (if present) currently associated with the executing program are used. The user ID is indicated as being already verified. If the remote LU does not trust the local LU to verify user IDs and passwords, or the local program does not have a user ID and security profile associated with it, then the local LU, in conjunction with a local security manager, can attempt to determine user ID, password, and security profile values to send to the remote LU. This determination is implementation specific. If values cannot be determined, then access security information is omitted on this allocation request.

Data Compression

Data compression applies to logical units in either an extended BIND or nonextended BIND environment. The session partners can negotiate for each direction whether to use data compression and then what type of data compression to use.

Run-length encoding (RLE) compression replaces strings of identical bytes with shorter encoded strings. Lempel-Ziv-like (LZ) compression is an adaptive dictionary-based compression algorithm similar to Lempel-Ziv algorithms that are

described in the technical literature, and is applicable to a wide range of data types. It uses tables that adapt dynamically to match the data being sent or received. LZ compression replaces the original data with a set of compression codes (indexes into the adaptively built dictionaries). Each set represents one or more bytes. Following are the variations of LZ compression:

- LZ (small table)—This type uses 9-bit compression codes.
- LZ (medium table)—This type uses 10-bit compression codes.
- LZ (large table)—This type uses 12-bit compression codes.

With LZ compression, each partner begins a session with an identical table for a given direction. The sender of RUs (in each direction) updates its send table as data is compressed. The receiver of the RU makes identical updates to its receive table as data is decompressed. This method keeps both ends of the session identical without exchanging table data between nodes.

At certain times, when a satisfactory compression ratio has been achieved, the dictionary may be frozen by the sender to improve throughput by reducing compression processing overhead. If the compression ratio becomes less satisfactory, the sender may unfreeze the table and continue the adaptive updating.

Session Services Extensions

APPN end nodes and network nodes can optionally support ***session services extensions***. These extensions allow LU-LU sessions among dependent LUs to be set up across an APPN network using APPN protocols. They allow an APPN network to have some of the same functions as a subarea network. These protocols are transparent to ordinary intermediate network nodes. A network node on the path between two network nodes that support session services extensions can find itself routing dependent LU-LU session traffic between a primary LU (PLU) and secondary LU (SLU). An APPN end node supporting session services extensions requires a network node server that also supports the extensions, in order to make use of any of the functions. These new protocols support existing application programs' invocation of functions that are available in subarea networks, such as third-party-initiated sessions, SLU-initiated sessions, queuing for partner LU availability, monitoring session limits, requesting that an LU release a session, and session-status notification.

SLU-Initiated Sessions

An SLU must be able to initiate sessions, and a node that attaches dependent LUs must be able to provide mode names, COS names, BIND images, and device characteristics, when required.

Queuing

Session queuing is the process of suspending the establishment of an LU-LU session until a needed resource (either an LU or a session with that LU) is available. There are two basic reasons for queuing of a session initiation request; it can be queued for either or both of the following reasons:

Queue for LU Enabled: A session initiation request can be queued because the PLU or SLU is not enabled for a session; for example, a printer is powered off or an application program is not initialized. Queuing for enabled LU is performed once the destination LU is found.

Queue for Session Limit: When an LU has the capability of having only a limited number of sessions and has reached the limit, subsequent session initiation requests may be queued. Once a current session is terminated, the node containing that LU dequeues the first session initiation request that was queued for session limit.

In APPN, session initiation requests that are queued require that the Locate chain be maintained between the nodes of the session partners as long as the request is queued. Once the required resource becomes available, the node managing that LU dequeues pending session initiation requests for that LU and resumes the network flows needed to establish the sessions.

Session requests indicate the queuing position for the request, should it become queued. Normally, requests are queued FIFO (first in, first out). This allows session requests to be dequeued in the order they are received. However, to support VTAM's version of third-party initiate, LIFO (last in, first out) is used to ensure that the SLU is directly passed from the current PLU to the next PLU indicated in the request.

Third-Party Initiation

Third-party initiation is a function, limited to PLUs, that allows the LU to establish a session between the LU it is currently in a session with and a third-party LU. The LU initiating the session setup request can be a menu server, a help function, or some other application program that might have reasons to end its session with the SLU and, in its place, initiate a session between the SLU and some other application program.

Session-Release Request

A PLU can initiate a session with an SLU and indicate in the request that, if the SLU is at its session limit, the current PLU should be notified that another PLU would like a session with the SLU. The PLU that sends the new session initiation request must indicate that the request can be queued. If the SLU is enabled and is not at its session limit, the session will be initiated. If the SLU is at its session limit, the session request will be queued and the current PLU will be notified. The current PLU can terminate its session with the SLU or ignore the request.

This function is normally used to improve the availability of printers shared by different application programs. The PLU receiving the session-release request terminates its session, for example, if no output is queued for the (printer) SLU or the current listing is finished.

Request LU Status

This function allows an OLU node to request the status of a destination LU. The DLU node, if it supports this function, will report its LU status in its reply.

Dependent LU Requester/Server

The ***dependent LU server (DLUS)*** is a product feature of a T5 (VTAM) network node supporting session services extensions. The DLUS function enables VTAM to have SSCP services for dependent LUs located in remote APPN end nodes or network nodes, which act as the ***dependent LU requester (DLUR)***. The DLUS provides SSCP services through standard SSCP-PU and SSCP-LU session flows

that are encapsulated and sent over LU 6.2 sessions. The DLUS function enables the user to continue using dependent LUs while also taking advantage of APPN benefits. A DLUR is a function of an APPN end node or network node that owns dependent LUs but obtains services from a DLUS. The DLUR function provides a remote boundary function for dependent LUs; that is, it removes the current restriction requiring that a node supporting dependent LUs be adjacent to a subarea boundary node. A DLUS can serve multiple DLURs; a DLUR can have multiple DLUSs.

The benefits of using the DLUS/DLUR function are:

- Fewer system definitions for VTAM via dynamic definition
- Minimized dependence on a single point of failure, due to minimized dependence on the NCP for network routing
- Ability of SSCP to own dependent LUs that are single or multiple hops away from either VTAM or NCP
- Network management and full downstream node visibility for problem determination
- Optimal routing from the PLU to the SLU without impacting nodes that do not support the DLUS
- Ability of LUs to move throughout the network without requiring VTAM definition changes
- Support of dependent LUs on network nodes and end nodes.

The SSCP-PU and SSCP-LU flows used to set up a dependent LU-LU session are encapsulated inside LU 6.2 session flows that are established with a mode name of CPSVRMGR. The endpoints of the sessions are the CPs of the DLUR and DLUS. The LU 6.2 session carrying encapsulated SSCP-PU and SSCP-LU session flows (from the DLUS to the DLUR) is active only when SSCP-PU and SSCP-LU sessions are needed. The dependent LU can reside in the dependent LU requester node itself or in an externally attached peripheral node. In the latter case, the requester strips off the LU 6.2 encapsulation and forwards standard SSCP-PU and SSCP-LU flows to the dependent node. The dependent LU-LU session is not constrained to traverse the same path as the encapsulated SSCP-PU and SSCP-LU sessions used for its setup.

Figure 119 illustrates APPN network nodes that provide intermediate session routing for LU-LU sessions between dependent LUs. The dependent LUs are not adjacent to a node where SSCP services are provided. The node where the SSCP resides must support the DLUS product feature. The dependent LU must be either in or adjacent to a node supporting the DLUR function.

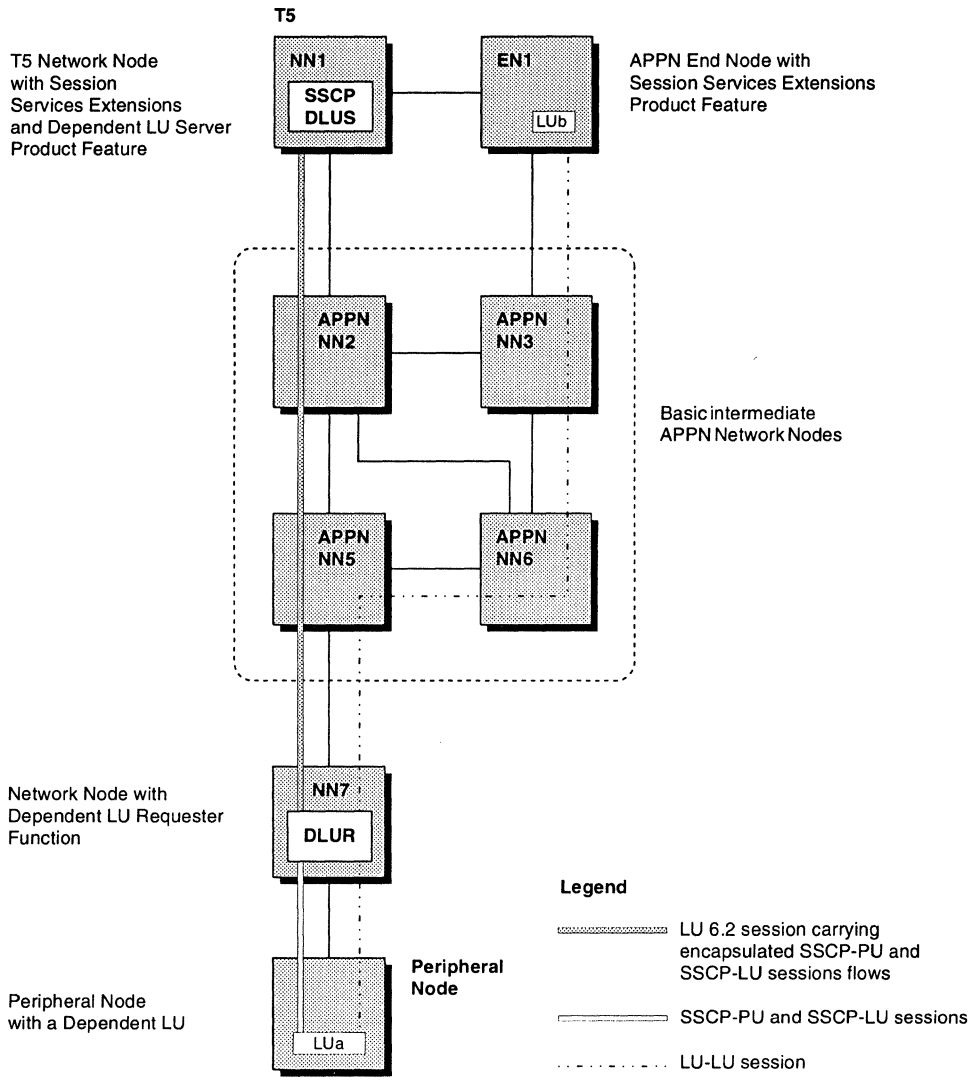


Figure 119. Dependent LU Server and Dependent LU Requester

Limited-Resource Connections

A limited resource is a switched link station that is deactivated when no sessions are using it and activated when sessions are established using that link station. When all the sessions using a switched link become inactive, it will automatically be deactivated. The purpose of limited-resource connections is to reduce the cost of keeping the switched connection active when it is not needed. This is possible in an APPN network because all nodes (end and intermediate) have session awareness. This is different in HPR networks; HPR end nodes have session awareness and can deactivate links based on session usage. However, HPR intermediate nodes have no session awareness, so they cannot deactivate links automatically based on session awareness; however, an intermediate link is deactivated if there is no session traffic on the link for a user-specified amount of time.

Nonswitched link stations cannot be defined as limited resources, whereas switched and shared-access transport facilities (SATF) can be. Link station connections through connection networks are always defined as limited resources.

VTAM User Variables

VTAM provides the ability to define "user variables" (or USERVARs) to associate a generic name with a collection of identical application programs (for example, CICS can be used to generically refer to CICS1, CICS2, ... CICS5.) The USERVAR is maintained by the node operator and is dynamically associated with the real name of the currently active instance of the generic application program (for example, CICS=CICS4). Terminals and application programs may then request sessions by specifying the USERVAR name (CICS), rather than having to know the real name of the instance of the application program that is currently active. If the currently active instance fails, the node operator simply changes the USERVAR value to a newly activated instance of the application program (for example, CICS=CICS2). Subsequent logon requests that specify the USERVAR name will be directed to the new instances of the application program.

This function is used by Information Management System (IMS) and Customer Information Control System (CICS) to map user logons to the IMS or CICS application that is currently active. USERVARs can also be used to facilitate migration from one application program release to another.

The VTAM application program interface (API) allows application programs to use USERVAR names across the API in place of LU names. Because USERVAR can be used across the API, many application programs that participate in sessions with an XRF-capable application program (such as IMS or CICS) can remain unchanged when USERVARs are used. Also, because VTAM determines whether a name refers to an LU or to a USERVAR and performs the appropriate translation automatically, the user does not have to code an interpret table to specify that a name used in a particular logon is actually a USERVAR.

USERVARs can be used to help with workload balancing. A group of applications can be given a common or generic name. VTAM uses this generic name, known as a USERVAR, to associate a logon request with the currently active member of this group. This capability can be used for workload balancing and preparing for the planned takedown of a host. USERVAR values can be managed by the user as well as by VTAM.

Terminating LU-LU Sessions

An LU-LU session remains active until the end users communicating over the session are finished exchanging data. When the end users are finished communicating on a session, one LU terminates the session. The session termination process differs depending on whether the LUs are dependent or independent.

Session Termination by Dependent LUs

Between dependent LUs, only the primary LU (PLU) can terminate a session. The secondary LU (SLU), however, can request that its SSCP assist in terminating a session.

Secondary LU Requests Session Termination

An SLU requests termination of an LU-LU session by sending a session-termination request to its SSCP over the SSCP-LU session. The LU sends a Terminate Self (TERM-SELF) request to ask for the SSCP's assistance in terminating the LU-LU session. The SSCP then sends a Control Terminate (CTERM) request to the PLU over the SSCP-LU session. The CTERM notifies the PLU that the SLU has requested session termination.

If the PLU is not finished communicating with the SLU, it continues to communicate with the SLU over the LU-LU session. When the PLU is finished communicating with the SLU, it sends the SLU an UNBIND request to deactivate the session. The SLU returns an UNBIND response to the PLU. Both the PLU and the SLU send the SSCP a SESSEND message to notify it that the session has ended.

Primary LU Requests Session Termination

Between dependent LUs, session termination is typically requested by the SLU. When the PLU requests that an LU-LU session be terminated, it sends a Shutdown (SHUTD) request to the SLU. This request ensures that the present work is completely processed before the session is deactivated. When all outstanding work has been processed, the SLU returns a Shutdown Complete (SHUTC) request to the PLU. Then the PLU sends the SLU an UNBIND request to deactivate the session, and the SLU returns an UNBIND response to the PLU. As with session termination by the SLU, the PLU and SLU both send the SSCP SESSEND messages to notify it that the session has ended.

Session Termination by Independent LUs

Independent LUs can terminate sessions without the assistance or knowledge of SSCPs. Between independent LUs, either the PLU or the SLU can terminate a session. Independent LUs do not send SHUTD and SHUTC requests to request session termination. Instead, LU *1* sends a Bracket Initiation Stopped (BIS) request to LU *2*. When all outstanding work has been processed, LU *2* returns a BIS of its own to LU *1*. Then LU *1* sends an UNBIND request to LU *2* to deactivate the session, and LU *2* returns an UNBIND response to LU *1*.

Chapter 9. Transaction Services

This chapter discusses architectures within the SNA transaction services layer that aid in managing end-user data in a network.

Transaction Services Architectures

The growth of distributed systems has created a demand for standardized transaction services for distributed system support. In response to this demand, IBM has supplied four transaction services architectures. They include:

- Distributed Data Management
- SNA/Distribution Services
- SNA/File Services
- Document Interchange Architecture.

These architectures define services for the interchange of user data among distributed systems.

Distributed Data Management

Distributed Data Management (DDM) is an architecture for a data management interface that enables concurrent file access among distributed SNA systems. It contains a data connectivity language and a set of standardized file models that enable data to be interchanged among different kinds of systems. Using DDM, an application transaction program (TP) can retrieve data from and update data on a file on a remote system.

DDM Concepts

The system that initiates a command to access data on a remote system is called the ***source system***. The ***source server*** is an architected service transaction program on the source system that translates the command into a standard transfer syntax. The source server routes the standard command to a communication support service on the source system, which transmits the command to the target system.

The system that contains the requested data is called the ***target system***. The ***target server*** is an architected service transaction program on the target system that translates the received command into a data management command that the target system understands. Once the target system has processed the command, it returns its response to the target server. The target server then routes the response to a communication support service on the target system, which transmits the response back to the source system.

DDM File Models

DDM defines models for four different file types: sequential, direct, keyed, and alternate index. Users access files through system software called ***access methods***. The rules for accessing each file type are defined by the access method.

The records in a ***sequential file*** are arranged in the chronological order in which they were first written. A user can access a particular record either sequentially or directly. A user accesses a particular record sequentially by reading all preceding records in the file, one at a time. A user accesses a particular record directly by providing its displacement from the beginning of the file to the access method. This displacement is called the ***relative byte address*** (RBA).

In a ***direct file***, there is an application-defined relationship between the data in a record and the record's position relative to other records in the file. A user can access a particular record directly by providing the record's relative position to the access method.

In a ***keyed file***, the position of record is determined by a part of the record called its ***primary key***. A user can access a particular record directly by specifying its primary key to the access method. The access method uses the key to determine the record's location by retrieving the location from an index file. An ***index file*** is a file used by the access method to store the primary keys of a keyed file. Each record of an index file contains a primary key together with the location of its corresponding record in the keyed file.

Alternate index files are index files that contain alternate keys for accessing keyed or sequential files. An ***alternate key*** is a part of a record other than that represented by the primary key. Each record of an alternate index file contains an alternate key together with its corresponding primary key or RBA. A record in a keyed file or sequential file can have alternate keys in multiple alternate index files. A user can access a particular record in a keyed or sequential file by specifying any of its alternate keys.

DDM Protocol Boundary

The DDM connectivity language is a protocol boundary defining a vocabulary of data management commands and a set of rules for using the commands. As with the LU 6.2 protocol boundary, the transaction program interface of a TP using DDM must have a semantic correlation to the DDM protocol boundary. DDM maps TP-specific commands into a standard internal syntax called the ***transfer syntax***. DDM translates TP-specific commands received from application transaction programs into the transfer syntax to be carried on LU 6.2 sessions to a remote system. It also translates commands received on LU 6.2 sessions in the transfer syntax into TP-specific commands for the receiving application transaction programs.

Examples of DDM commands issued on a source system include the following:

- OPEN establishes a logical connection with a file on the target system.
- SETKEY sets the file position based on a key supplied with the command.
- GETREC returns the record located at the current file position.
- SETUPDKY conveys the source program's intent to modify the record specified by the key supplied with the command.
- MODREC modifies the record located at the current file position.
- CLOSE terminates the logical connection with the file on the target system.

DDM Request Unit Formats

DDM uses the general data stream (GDS) for formatting data to be transmitted. The GDS data stream is supported by LU 6.2 and provides an identifier that DDM uses to specify the contents of the GDS variable data area. Possible data area contents include commands, responses, and objects. ***Objects*** include scalar objects and collection objects. A ***scalar object*** is a string of bytes formatted as required by a class description for the object. A ***collection object*** is an object that contains other objects in its data structure.

DDM Operation

Figure 120 depicts how DDM would service an application transaction program (TP) command requesting data from a remote system.

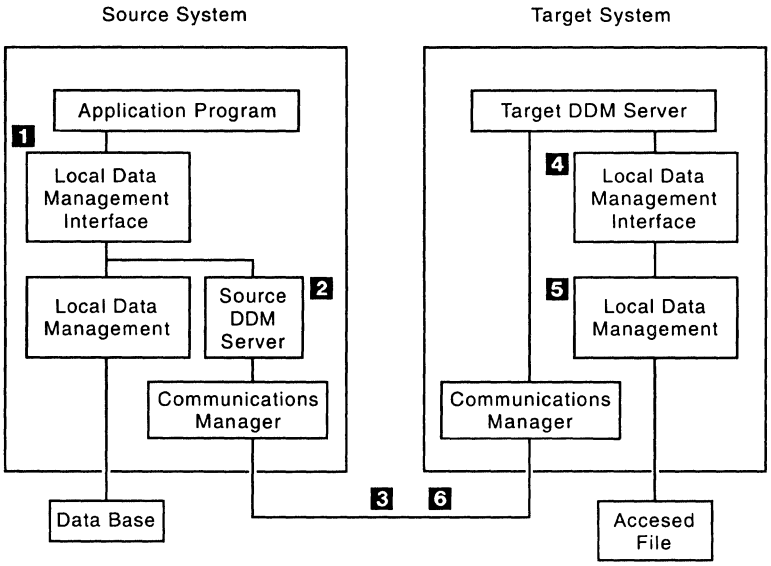


Figure 120. Overview of DDM Processing

The following comments correspond to the numbers in Figure 120.

1. The source TP routes the command to the local data management interface. The local data management interface determines that the requested data is not in the local system and directs the command to a source server.
2. The source server translates the local command into one or more standard DDM commands. It then builds a GDS data stream to carry the DDM commands.
3. The source system transmits the data stream to the target system.
4. The target system directs the DDM command to the target server which prepares the command for the data management interface of the target system.
5. The target data management interface retrieves the requested data and sends it back to the target server.
6. The target server packages the response data as DDM objects in a GDS data stream and transmits it back to the source system.

For additional information on DDM, refer to *Distributed Data Management Architecture: General Information Manual.*

SNA/Distribution Services

SNA/Distribution Services (SNA/DS) provides, at a user's request, the asynchronous distribution of the user's data to other users. Asynchronous data distribution is required by many distributed applications and systems services, including office systems, network management, and file transfer. The initial implementations of SNA/DS are for office systems applications and the change management category of SNA/management services (SNA/MS). (For information on SNA/MS, see Chapter 10, "Managing an SNA Network.")

Asynchronous communication is not to be confused with asynchronous data link protocol. ***Asynchronous communication*** means that the sender and receiver need not be in communication simultaneously. It is analogous to the delivery of a letter by a postal service—the sender can transmit data without the participation or knowledge of the receiver. The communication service at the destination queues the data until the receiver is located and can receive the data.

As in synchronous communication, asynchronous communication can require responses to requests. But whereas with synchronous communication, requests and responses are synchronized, with asynchronous communication, responses to requests can be returned in any order. The correlation of requests and responses is the responsibility of the application, not of SNA/DS.

SNA/DS Concepts

A ***distribution service unit*** (DSU) is a collection of service transaction programs within the type 6.2 logical unit. DSUs provide the distribution service to application transaction programs. SNA/DS defines an application transaction program that uses the services of SNA/DS to be an ***agent***. Agents employ SNA/DS services on behalf of users. A ***user*** can be a person, department, application program, or database that invokes SNA/DS through an agent.

The relationship of agents to users is a many-to-many relationship. Multiple users can use a single agent, or copies of the agent, dispersed at multiple locations. Conversely, a single user can use multiple agents to handle a series of SNA/DS requests.

A ***distribution network*** is a collection of interconnected DSUs. The work performed by a DSU is called a ***distribution***. The work performed on a distribution includes accepting a request, generating and moving copies of the distribution across the network, delivering the distribution to the specified destinations, and optionally confirming delivery by returning reports back to the originator.

Users, agents, and DSUs can assume three SNA/DS roles: origin, intermediate, and destination. A distribution is initiated by an ***origin user*** or ***origin DSU*** through an ***origin agent*** at the origin DSU. The distribution request designates one or multiple ***destination users*** or ***destination DSUs***. The origin user or DSU need not know the locations of the destinations; SNA/DS determines the location of each destination. The distribution is created by the origin agent and includes the name of the ***destination agent***. Though a distribution may have multiple destination users or DSUs, only a single destination agent (or multiple copies of that agent) is named to handle the distribution. The origin DSU transmits the distribution to each destination DSU (including DSUs representing destination users). As it passes through the network, the distribution can pass through multiple ***intermediate DSUs***. When the distribution reaches a destination DSU, SNA/DS invokes the destination agent to handle the distribution. The destination agent notifies any destination users named as recipients of the distribution.

Figure 121 shows a distribution network containing an origin agent (agent *y*), an origin DSU, and two destination DSUs, each containing a copy of the destination agent (agent *x*).

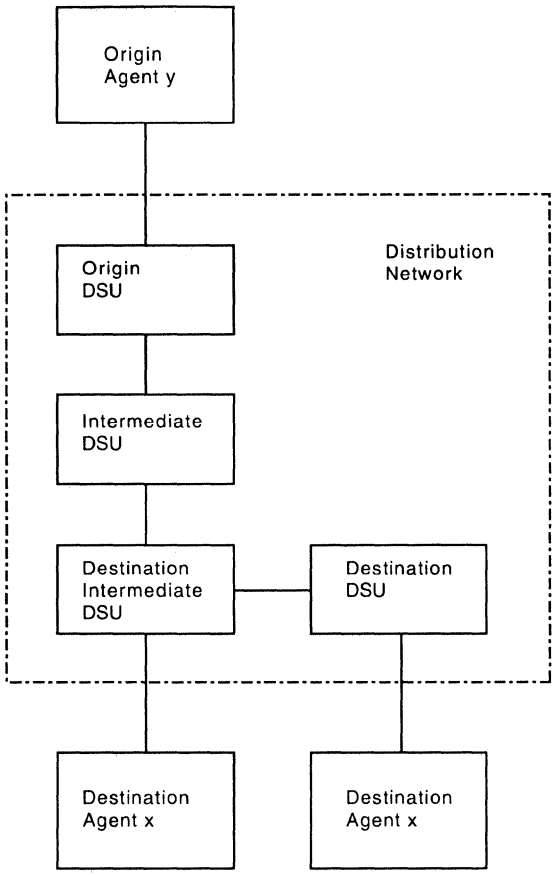


Figure 121. Network of Distribution Service Units

The data distributed by SNA/DS is called an ***object***. A SNA/DS distribution contains two kinds of objects: an agent object and a server object. The ***agent object*** is intended for small amounts of data that can be stored by SNA/DS and passed directly to the destination agent. The ***server object*** is intended for large amounts of data or data that requires specialized handling (such as encryption), and is passed to a destination server. A DSU uses ***servers*** to retrieve and store server objects.

Two classes of servers are defined in SNA/DS: specific servers and general servers. ***Specific servers*** provide mapping between object byte streams and agent-specific formats. They are typically able to interpret the meanings of byte streams. ***General servers*** write and read copies of server objects. Unlike specific servers, general servers are not concerned with the contents of server objects. A general server is used by an intermediate DSU for temporarily storing a server object as it is transferred from one adjacent DSU to another.

SNA/DS Protocol Boundaries

SNA/DS defines protocol boundaries for communicating with agents and servers. Like the LU 6.2 protocol boundary, they consist of sets of verbs whose value lie in their semantics, not their syntax. SNA/DS uses the ***agent protocol boundary*** for communicating with agents and the ***server protocol boundary*** for communicating with servers.

Agent Protocol Boundary

The verbs SNA/DS defines for communicating with agents include the following:

- SEND_DISTRIBUTION is issued by an origin agent to initiate a distribution.
- QUERY_DISTRIBUTION is issued by an origin agent to determine the current state of a distribution. The origin DSU may need to perform auxiliary server operations before sending the distribution.
- RECEIVE_DISTRIBUTION is issued by a destination agent to receive a distribution.
- RECEIVE_DISTRIBUTION_REPORT is issued by an agent to receive a distribution report.
- OBTAIN_LOCAL_SERVER_REPORT is issued by an agent to obtain a report generated by a local specific server.

Server Protocol Boundary

The verbs SNA/DS defines for communicating with servers include the following:

- INITIATE_READ is issued by SNA/DS to prepare a server to perform the read function.
- ASSIGN_READ_ACCESS is issued by SNA/DS to instruct a server to place a "lock" on an object before SNA/DS reads it. The lock ensures, in case the distribution is to be transmitted on multiple sessions, that the object remains unchanged while it is being read.
- READ is issued by SNA/DS one or more times to instruct a server to read an object.
- RELEASE_READ_ACCESS is issued by SNA/DS to remove the lock set by ASSIGN_READ_ACCESS.
- TERMINATE_READ is issued by SNA/DS to complete the read function.
- INITIATE_WRITE is issued by SNA/DS to prepare a server to perform the write function.
- WRITE is issued by SNA/DS one or more times to instruct a server to write or update an object.
- TERMINATE_WRITE is issued by SNA/DS to complete the write function.

SNA/DS Request Unit Formats

SNA/DS defines a request unit format called the ***Distribution Transport Message Unit*** (DTMU) for formatting the data to be transmitted in a distribution. The DTMU is shown in Figure 122.

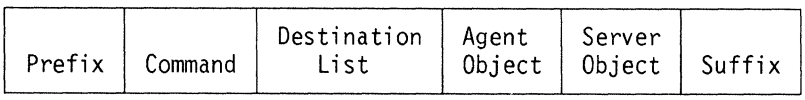


Figure 122. Distribution Transport Message Unit

The ***prefix*** structure identifies the start of the DTMU. The ***command*** structure contains the name of the destination agent along with other control information. The ***destination list*** structure contains the names and addresses of distribution recipients. The agent and server objects are optional. The ***agent object***, if present,

contains an object created by the origin agent to be delivered to the destination agent. The ***server object***, if present, contains one or more SNA/File Services (SNA/FS) information objects to be delivered to a SNA/FS server for storage. (See "SNA/File Services" on page 267 for information on SNA/FS.) Finally, the ***suffix*** structure defines the end of the DTMU.

As part of a distribution request, the originator may ask that SNA/DS provide feedback on the status of the distribution. For example, the originator may wish to be informed if SNA/DS is unable to deliver the distribution. SNA/DS uses a ***Distribution Report Message Unit*** (DRMU) in which to return such feedback information. The DRMU is shown in Figure 123.

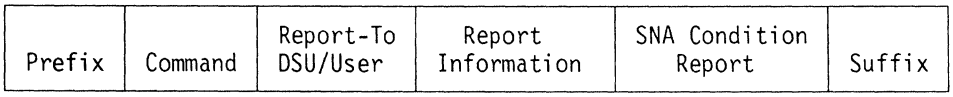


Figure 123. Distribution Report Message Unit

Like a DTMU, a DRMU is introduced by a prefix and concluded by a suffix. The ***command*** contains the control information for the DRMU. The ***report information*** and ***SNA condition report*** describe the particular condition being reported. The ***report-to DSU/user*** is the DSU or user to which the report is being sent.

Figure 124 illustrates the major components of a DSU.

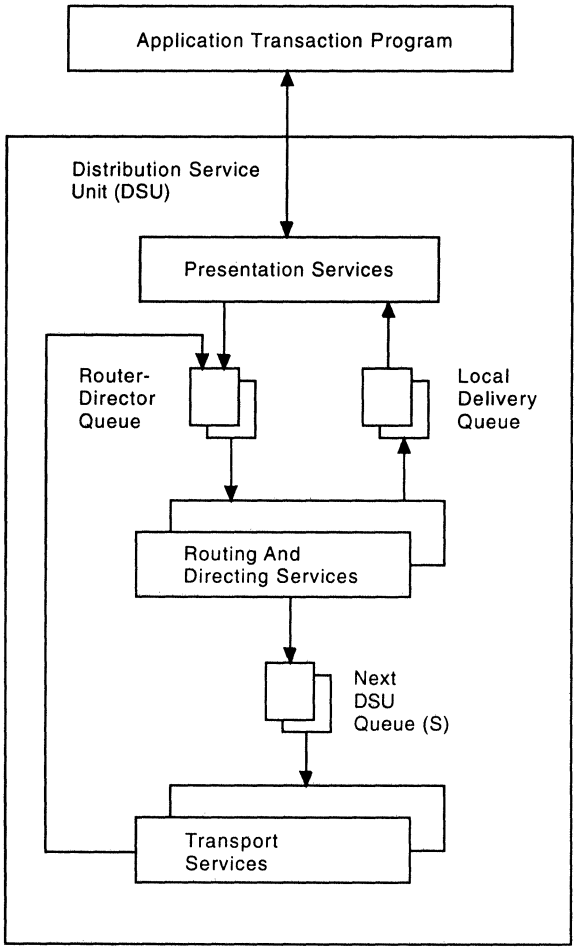


Figure 124. Components of a Distribution Service Unit

Presentation services (PS) processes a distribution verb issued by an application transaction program (TP). In processing SEND_DISTRIBUTION, PS maps the verb to an internal form and places the entry on a ***reader-director queue*** (R-D queue). In processing a RECEIVE_DISTRIBUTION, PS checks the ***local delivery queue*** for entries destined for the TP. If one is found, PS dequeues it, maps it into a format understood by the TP, and delivers it to the TP.

Routing and directing services processes entries on the R-D queue. For each entry, it determines the address of the recipient and routes the distribution to the appropriate distribution queue. If the recipient is a local recipient, the distribution is routed to the local delivery queue; if the recipient is a remote recipient, the distribution is routed to a ***next-DSU queue***. A ***local recipient*** is another TP served by the same DSU, and a ***remote recipient*** is a TP served by another DSU.

Transport services uses the facilities of LU 6.2 to transfer distributions between adjacent DSUs. When sending a distribution to another DSU, transport services processes an entry from a next-DSU queue. When receiving a distribution from another DSU, transport services builds an entry and places it on the R-D queue to be processed by routing and directing services.

For additional SNA/DS information, refer to *SNA Distribution Services Reference*.

SNA/File Services

SNA/File Services (SNA/FS) defines facilities for identifying an enterprise's files in a structured, location-independent manner and for distributing them to nodes in an SNA network. It defines a common service for identifying, fetching, moving, and storing files. SNA/FS is designed to provide specific file server functions for SNA/DS and SNA/DS agents such as SNA/MS. SNA/DS and SNA/DS agents invoke SNA/FS to perform the file functions.

SNA/FS Concepts

SNA/FS identifies a file with a global name. A ***global name*** is a file name that is unique across the distribution network. It consists of a string of up to ten concatenated ***tokens***, which are symbolic names assigned by an enterprise. Within a global name, the position of a token relative to the other tokens in the name would be typically based on its hierarchical relationship to the other tokens. For example, a global name consisting of four tokens might be:
XCORP.EASTDIV.DEPT5.TIMECARDS.

SNA/FS accepts only commands that refer to files by their global names. The global names are stored in ***catalogs***. A catalog enables SNA/FS to make the association between a file's global name and the name by which it is known on a local system.

SNA/FS defines four roles for nodes in an SNA network: requester, source, target, and report-to. All of these roles can be involved in a single request. At a ***requester node***, an agent can request SNA/DS to fetch an object from a ***source node***, transfer it to a ***target node*** where it is to be stored, and report the distribution to a ***report-to node***. Agents and servers cooperate in performing the functions prescribed by these roles. For a given distribution, an agent or SNA/FS server learns of its particular role either through protocol boundary verbs and parameters or through objects encoded in the server object.

SNA/FS Protocol Boundary

The SNA/FS protocol boundary consists primarily of the server protocol boundary verbs defined by SNA/DS. In addition, however, two verbs are provided for agents to modify the SNA/FS catalog. Agents issue CATALOG to add entries to the SNA/FS catalog and UNCATALOG to remove entries from the SNA/FS catalog.

An agent can invoke SNA/FS directly through the server protocol boundary. For example, the agent can store a data object by issuing INITIATE_WRITE, a series of WRITEs, and a TERMINATE_WRITE. In addition, an agent can invoke SNA/FS indirectly through the agent protocol boundary. This is done by using the SPECIFIC_SERVER_INFO parameters on the SEND_DISTRIBUTION verb to describe the actions to be performed by the SNA/FS server. For example, after storing an object, an agent can then distribute the object by issuing the SEND_DISTRIBUTION verb with server instructions encoded in the SERVER_SPECIFIC_INFO. SNA/DS then uses the SPECIFIC_SERVER_INFO to interact with SNA/FS and retrieve the information for distribution.

SNA/FS Request Unit Format

SNA/FS does not define request unit formats of its own, but makes use of the Distribution Transport Message Unit (DTMU) defined by SNA/DS. In a distribution involving SNA/FS, the distribution object structure of the DTMU contains a destination list, zero or one agent objects, and one server object. The ***server object*** structure contains one or more SNA/FS information objects created by a SNA/FS server for distribution to another SNA/FS server.

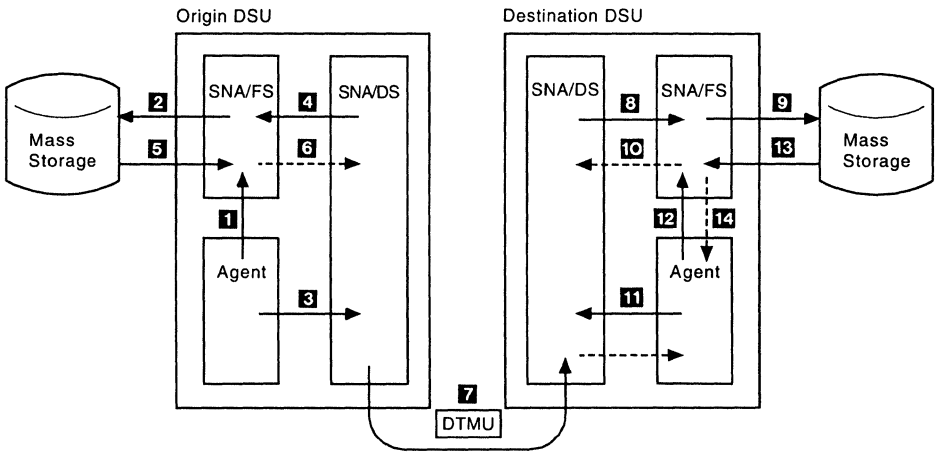
An agent encodes information for a destination agent in an agent object and passes the agent object to SNA/DS in a parameter of the SEND_DISTRIBUTION verb. SNA/DS then moves the agent object into the DTMU and delivers it to the destination DSU. At the destination DSU, SNA/DS passes the agent object to the destination agent in a parameter of the RECEIVE_DISTRIBUTION verb.

SNA/DS receives and delivers file data in server objects. SNA/DS receives a server object from SNA/FS in a parameter of the READ verb. In addition to the file data contents, the server object contains instructions to the destination SNA/FS server, information identifying the data, and information on the data's storage requirements. SNA/DS then moves the server object into the DTMU and delivers it to the destination DSU. At the destination DSU, SNA/DS passes the server object to the destination server in a parameter of the WRITE verb.

While SNA/DS processes a distribution, there is no direct interaction between server and agent. When and how an agent stores or retrieves data through a server, whether at the origin or destination DSU, is outside the awareness of SNA/DS.

SNA/FS Operations

The interaction of SNA/DS with agents and SNA/FS servers while delivering agent and server objects to a destination DSU is illustrated in Figure 125.



Legend:

DSU = Distribution Service Unit
DTMU = Distribution Transport Message Unit
SNA/DS = SNA/Distribution Services
SNA/FS = SNA/File Services

Figure 125. Interaction of SNA/DS with Agent and SNA/FS Server

The following comments correspond to the numbers in Figure 125.

1. The origin agent issues a series of server protocol boundary write-type verbs to pass a server object to SNA/FS.
2. SNA/FS writes the server object to mass storage.
3. The origin agent initiates a distribution by issuing a SEND_DISTRIBUTION verb to SNA/DS. The origin agent passes an agent object in a parameter of the SEND_DISTRIBUTION verb to be sent to the destination agent.
4. SNA/DS issues a series of server protocol boundary read-type verbs to retrieve the server object from SNA/FS.
5. SNA/FS retrieves the server object from mass storage.
6. SNA/FS passes the server object to SNA/DS on the READ verb completions.
7. SNA/DS transmits the agent and server objects in a DTMU over an LU 6.2 session to the destination DSU.
8. SNA/DS at the destination DSU invokes a SNA/FS server and stores the server object by issuing a series of server protocol write-type verbs.
9. The SNA/FS server writes the server object to mass storage.
10. When the server object has been stored, SNA/FS passes information to SNA/DS on the WRITE verb completions identifying and reporting on the file data. SNA/DS keeps this information in a control block to be passed at a later time to the destination agent. SNA/DS then causes the local operating system to start the destination agent.
11. Once started, the agent issues a RECEIVE_DISTRIBUTION verb. SNA/DS passes the control block to the agent in the verb completion. The control block contains not only the information received from the SNA/FS server, but also some SNA/DS control information and the agent object sent from the originating agent.
12. The destination agent issues a series of server protocol boundary read-type verbs to retrieve the server object from SNA/FS.
13. SNA/FS reads the server object from mass storage.
14. SNA/FS returns the server object to the destination agent on the READ verb completions.

For additional SNA/FS information, refer to *SNA File Services Reference*.

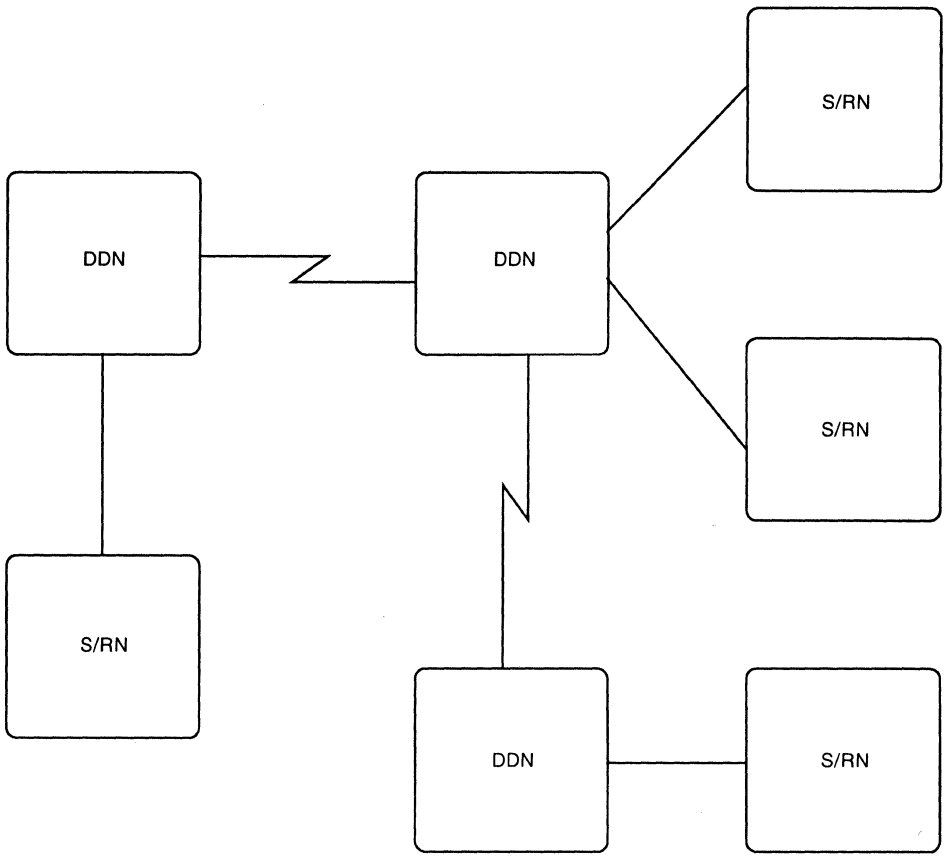
Document Interchange Architecture

Document Interchange Architecture (DIA) is a program-to-program communication architecture that enables documents to be interchanged among a broad spectrum of IBM office systems. It defines the protocol boundaries and data structures that enable application transaction programs to communicate instructions for filing, accessing, retrieving, and revising documents. DIA defines a ***document*** as an identifiable block of information of any size. Included are facsimile images, binary data files, and other information not usually thought of as documents.

DIA Concepts

A ***document distribution node*** (DDN) is a node providing DIA services that receive, store, route, and deliver information for source and recipient nodes. A ***source node*** (SN) is a node at which DIA services send information to recipient nodes on behalf of end users. A ***recipient node*** (RN) is a node at which DIA services receive information from source nodes on behalf of end users. Source and recipient nodes can be thought of as work stations locally attached to DDNs.

Naturally, a source node can also be a recipient node, depending on whether it is sending or receiving information. When the distinction is not relevant to the context in which it is being discussed, a source or recipient node is referred to as a ***source/recipient node*** (S/RN). A ***document distribution system*** is a network of DDNs together with their attached source/recipient nodes. Figure 126 illustrates a document distribution system.



Legend:

DDN = Document Distribution Node
S/RN = Source / Recipient Node

Figure 126. A Network of Document Distribution Nodes

DIA consists of an information interchange base that provides document distribution services, document library services and application processing services for general purpose office system functions. ***Document distribution services*** (DDS) requests, stores, and delivers documents in a document distribution system. DDS only transfers documents between S/RNs on the same system. DDS invokes SNA/Distribution Services to asynchronously interchange data between different systems in a network.

Document library services (DLS) stores and retrieves information objects. DLS functions include filing, searching, retrieving, and deleting of information. DLS, and the libraries it maintains, resides on a document distribution node.

Application processing services (APS) executes and controls other office system applications and DIA services. APS functions include formatting documents from revisable form to final form, modifying descriptive information, and enabling one application transaction program to execute another.

DIA Protocol Boundary

The DIA protocol boundary is a set of commands provided for intercommunication between source nodes, recipient nodes, and document distribution nodes. DIA defines three sets of commands:

- Document distribution services commands
- Document library services commands
- Application processing services commands.

Document distribution services commands (DDS commands) support both asynchronous document distribution between DDNs, and direct document exchange between SNs and RNs. DDS commands are used for initiating distributions, delivering documents, and returning status regarding distributions. Examples of DDS commands include the following:

- REQUEST_DISTRIBUTION is issued by an SN to request its DDN to transport a document to specified recipients.
- DISTRIBUTE is issued by a DDN to transport a document to another DDN.
- STATUS_LIST is issued by a DDN to notify one of its RNs that one or more documents is available for the RN.
- LIST is issued by an RN to request its DDN to return a list of the documents held by the DDN for the RN.
- OBTAIN is issued by an RN to request its DDN to return one or more documents held by the DDN for the RN.
- DELIVER is issued by a DDN to return information to an RN in response to LIST or OBTAIN commands.

Document library services commands (DLS commands) maintain user documents in a document library. Examples of DLS commands include the following:

- FILE is issued by an SN to request its DDN to store a document in the library for an authorized document owner. A document can have more than one owner.
- RETRIEVE is issued by an RN to request its DDN to return a copy of a document for an authorized document requester.
- SEARCH is issued by an RN to request its DDN to locate and return a list of documents in the library that have the characteristics specified in the SEARCH command.
- DELETE is issued by an S/RN to its DDN to request that access privileges to a document by the requester be removed. If all owners of a document have requested DELETE, the DDN removes the document from the library.

Application processing services commands (APS commands) request the execution of application transaction programs. Examples of APS commands include the following:

- EXECUTE is issued by an S/RN to request its DDN to invoke a specified program for execution.
- FORMAT is issued by an S/RN to request its DDN to submit a specified document to an specified program for formatting.

DIA Request Unit Format

DIA defines a request unit format called a ***document interchange unit*** for formatting data to be exchanged between DIA processes. As shown in Figure 127, a document interchange unit (DIU) consists of a prefix, a command sequence, zero or more data units, zero or more document units, and a suffix.

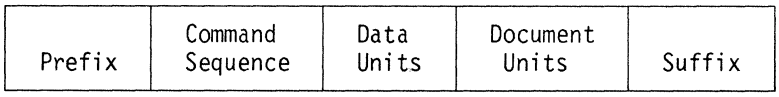


Figure 127. A DIA Document Interchange Unit

The ***prefix*** and ***suffix*** structures are defined for the DIU in the same way as they are defined for the SNA/DS interchange unit. The ***command sequence*** structure can contain up to 255 commands. The commands are executed by the receiver of the DIU in the order in which they occur in the DIU. A ***data unit*** contains information that is referenced by operands of one or more commands in the command sequence structure. Information common to multiple commands can be placed in the the data unit structure to avoid repetition. The ***document units*** structure contains the document itself together with a ***document unit ID*** identifying the DCA-defined architecture of the document, and a ***document profile*** describing the document structure and contents.

DIA Operation

This section describes a scenario in which an end user at a source node (SN A on DDN A) distributes a document to an end user at a recipient node (RN B on DDN B).

When the end user at SN A initiates a distribution, SN A submits a REQUEST_DISTRIBUTION command to DDN A specifying a recipient at RN B. DDN A acknowledges the command to SN A. DDN A then invokes SNA/DS to send a DIU to DDN B containing the document and a DISTRIBUTE command specifying the recipient at RN B. DDN B responds to DDN A with an acknowledgment indicating that it has stored the document and will proceed with the request.

DDN B invokes an application transaction program at DDN B to service local end users and enqueues the document to be delivered when the recipient end user signs on. After the recipient end user signs on to RN B, RN B submits, at the end user's request, a LIST command to DDN B to determine what documents are enqueued for the recipient end user. DDN B returns a DIU to RN B containing a DELIVER command and a document unit containing data about all documents enqueued for the end user. After the end user selects a document for delivery, RN B sends a DIU to DDN B containing an OBTAIN command specifying the document to be delivered. DDN B responds with a DIU containing a DELIVER command and the requested document.

The recipient end user may then wish to save the document in the library under a name of the end user's own choosing. If so, RN B sends a DIU containing a FILE

command and a unique global name for the document. Document library services of DDN B files the document and acknowledges the FILE command.

For additional DIA information, refer to *Document Interchange Architecture: Technical Reference*.

Chapter 10. Managing an SNA Network

This chapter discusses SNA/Management Services, the functions of SNA components in management services, and the management services request units.

Management Services Categories

SNA/Management Services (SNA/MS or MS) helps plan, organize, and control an SNA network. Management services consists of five categories, each of which has several elements. The five categories are:

- Problem management
- Performance and accounting management
- Configuration management
- Change management
- Operations management.

Problem Management

Problem management detects and corrects a problem. A problem refers to some condition that results in an end user losing access to a system resource. Problem management has five elements:

- Problem determination—detecting a problem or an impending problem
- Problem diagnosis—finding the cause of the problem
- Problem bypass and recovery—using an alternative resource until the problem is corrected
- Problem resolution—correcting the problem
- Problem tracking and control—reviewing the handling of the problem.

Performance and Accounting Management

Performance and accounting management monitors the operation of the network, starts problem management procedures if the performance is not what it should be, and fine-tunes the operation of the network to improve its performance. Performance and accounting management has seven elements:

- Response-time monitoring—monitoring the response times of end users and starting problem management procedures if the times are too great
- Availability monitoring—monitoring the availability of a component
- Utilization monitoring—monitoring the usage of resources and starting problem management procedures if the usage becomes too high
- Component delay monitoring—monitoring delays at critical components
- Performance tuning—improving performance based on information from performance tracking and control
- Performance tracking and control—collecting and reporting performance information
- Accounting—recording and tracking usage charges for system resources.

Configuration Management

Configuration management maintains information about the resources in the network. Other categories of SNA/MS can then use the information that configuration management controls. For example, problem management can use the information to determine the network name of a failed resource, the network address of the resource, and the organization responsible for servicing the resource. Configuration management can be broken down into:

- Knowledge of the physical identification of resources
- Knowledge of the logical relationship between resources.

Change Management

Change management plans, controls, and applies changes to the network. Change management is closely related to configuration management in that configuration management deals with the existing state of the network and change management deals with how that state changes. Change management is concerned with three types of changes:

- Hardware changes
- Microcode changes
- Software changes.

Change management uses SNA/Distribution Services and SNA/File Services to distribute large files, to distribute requests to manipulate the files, and to distribute reports to track distribution and installation of the files.

Operations Management

Operations management provides the means to query and control distributed network resources from a central site. The two sub-categories that accomplish this are the ***common operations services*** function set and the ***operations management*** function set. Depending on solution requirements, either (or both) of these function sets provides for skill centralization, network automation, and, thus, a reduced manual workload for network operators.

Common Operations Services

Common operations services is a set of services used by all of the major categories of MS to communicate with specialized management application programs not currently defined by MS. The common operations services architecture provides mechanisms for transporting information between network management application programs, or between a network operator and one such application program. The meaning of the information being transported is entirely defined by the application programs using the supporting services.

Common operations services provides functions to:

- Package free-form (application-defined) data within request units defined by common operations services. The data is enveloped between common operations services control data fields that identify how the data is encoded, but not its meaning.
- Route free-form commands from a network operator to an application program named by the operator, and route free-form messages from an application program to an operator named by the application program.

Through these services, common operations services provides a network management environment that accommodates multivendor and future data communication products.

Operations Management

Operations management provides the capability to control distributed network resources. Activating and deactivating resources, as well as setting network resource clocks are all functions included in this category. In addition, a cancelation function enables previously sent commands (including those executing at the target) to be terminated.

Operations management extends the functions of common operations services and change management, improving a network manager's ability to control distributed system resources. Additionally, if an interface between the operations management and problem management focal-point operators is implemented, then operations management commands can be used to assist in problem bypass and recovery.

Managers of large enterprise networks require the ability to operate systems in the network from a central site. The following are the benefits of this approach:

- Labor costs are reduced, since the requirement to attend systems locally can be eliminated (or at least the required skill level of those people attending the local systems can be reduced).
- Operations management commands and procedures are standardized across the network.

Management Services Roles

Two MS roles are defined for SNA nodes: entry points and focal points.

Entry Points

An ***entry point*** is a node that provides distributed network management support. Entry points are the sources of MS data on the status of network resources. Nodes of all types can act as entry points. An entry point:

- Exchanges MS data with focal points. MS data includes commands received from focal points, and MS information sent to focal points. The MS commands can be requests for information or requests for an action on the part of the entry point or a resource within the entry point. The MS information can be *solicited* MS information (requested by the focal point), or *unsolicited* MS information on events occurring within the entry point. An example of unsolicited information would be an *alert* sent by an entry point as notification of a link failure.
- Exchanges MS data with the resources in its node. Commands received from a focal point can be routed to specific node components. Solicited information returned from a node component is forwarded to the focal point that requested the information. Unsolicited information received from a node component is sent to the focal point responsible for handling the MS category to which the event belongs. In the case of a link failure, the focal point handling the problem management category would be notified.

Focal Points

A ***focal point*** is an entry point that provides centralized network management for itself and other entry points for one or more network management categories. Focal points are the destinations of MS data on the status of network resources. The focal point role can be performed at a T5 node in a subarea network, or an APPN network node in an APPN network. A focal point:

- Exchanges MS data with entry points. A focal point sends MS commands to entry points to request information on, or to manage entry point resources. It also receives and logs solicited and unsolicited MS information received from an entry point.
- Exchanges MS data with a network operator. A ***network operator*** is a person or application program that manages network resources. A focal point receives commands from the network operator and reports solicited and unsolicited MS information to the operator. The data can be reported immediately, or stored and reported at a later time.
- Exchanges MS data with another focal point. A focal point for problem management or user-defined categories has the ability to forward data that it receives to another focal point.

MS data can be routed to a single focal point from multiple domains. The set of domains for which a focal point is responsible is called the focal point's ***sphere of control***. In a subarea network, one T5 node can act as a focal point not only for the nodes in its own domain, but for those the domains of other T5 nodes as well. Similarly, in an APPN network, a network node (NN) can act as a focal point for the domains of multiple network nodes.

A single network can also have multiple focal points. Each focal point can provide services for a different category of network management. For example, one focal point might be established for problem management and another for change management. The technique of distributing MS support across multiple focal points is known as ***distributed management services***.

The focal point function also includes:

- Nested focal point support—MS capabilities supports nested focal points, allowing a focal point to forward data that it receives to another focal point.
- End node and network node support—In the APPN environment, a network node provides services for an end node, including MS capabilities. This function ensures that a network node informs all end nodes that it serves (except migration nodes) about the name and status of the current focal point. APPN end nodes consider this focal point to be a domain focal point.
- Migration node support— OS/400 Version 2 Release 1.1 (or earlier) systems can function as migration nodes, as either network nodes or end nodes in an APPN network. Only Alert data can be sent from migration nodes to a focal point.
- Multiple backup focal points—Up to eight backup focal points can be specified.

Sphere of Control Manager (SOC-MGR)

The ***sphere of control manager (SOC-MGR)*** is an enhancement that provides additional functions for MS-CAPS and the network operator.

- Management Services Capabilities (MS-CAPS) functions:
  - Maintains a list of entry points that are within a focal point's sphere of control
  - Maintains the state of each entry point within the focal point's sphere of control
  - Attempts to re-establish a relationship with an entry point when the relationship between the entry point and the focal point is lost
  - For NetView, reads in information from the sphere of control configuration file during NetView initialization and uses the information to set up the sphere of control
  - For NetView, restores the sphere of control environment during NetView recovery.
- Network operator functions:
  - Deletes entry points from the sphere of control
  - Displays the names and states of entry points
  - Adds entry points to the sphere of control.

APPN Session Problem Determination

Session problem determination allows the user to solve logical, connectivity-related problems that occur in the network. These problems can include the inability to get the session started, a hung session, an abnormally failed session, and a poor session response time. With the new APPN session PD support, the NetView session monitor provides configuration and session status information for sessions for both pure APPN networks and mixed networks.

APPN session problem determination support adds the following functions:

- Support for SSCP takeover and giveback—the user can see which resources have been taken over and which resources have been given back
- Support for APPN border node—the user can see the border node in the network configuration
- New APPN session and route data in the external log.

MS Implementation Choices

Because the MS architecture can be implemented in different data communication products, the architecture defines a set of rules to which each product must conform for selecting the MS functions that the product will support. If each product selected the various MS functions arbitrarily, it is unlikely that any two products would select exactly the same set of functions or that the network components would combine to produce a complete set of functions. As a result, it could be difficult to connect the network components, and it would be impossible to guarantee that adequate functions were provided. The rules for selecting which MS functions to implement fall into the following categories:

- Base subsets and optional subsets of the function sets
- Role requirements
- Electives.

Base Subsets and Optional Subsets of the MS Function Sets

An MS ***function set*** is a collection of services that together perform an overall MS function for a physical unit or a control point. Each MS function set has a mandatory or ***base subset*** that all implementations of that function set must support. The rest of the function set is composed of ***optional subsets***. Implementations of that function set can choose to support some or all of the optional subsets, depending on their role requirements. Figure 128 illustrates the base and optional subsets for a typical product implementation.

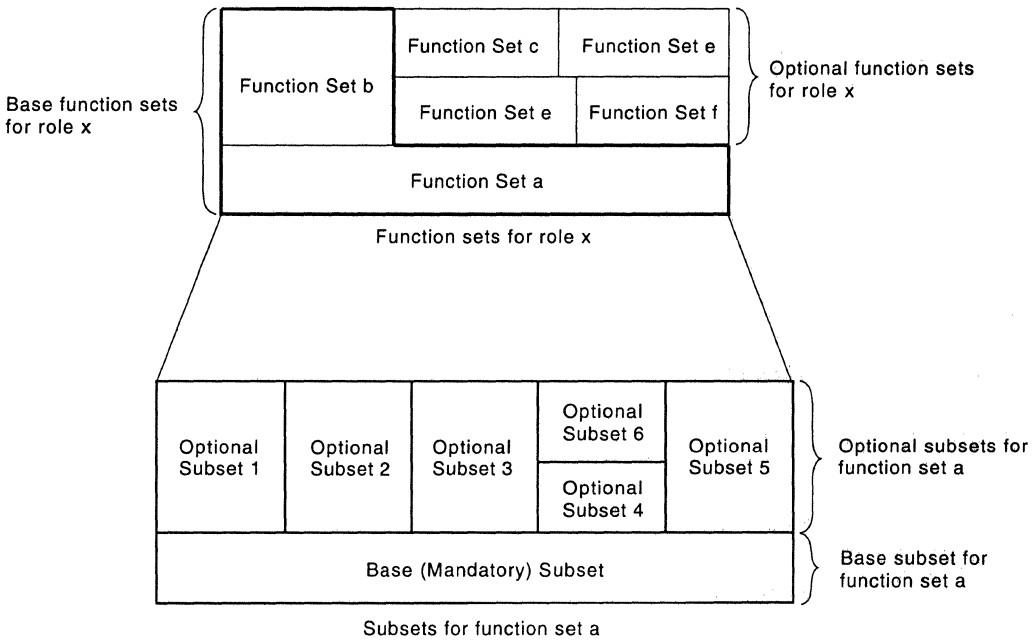


Figure 128. Role Function Sets and Their Base and Optional Subsets

Role Requirements

A data communication product's role in the network is determined by the base function sets and optional function sets chosen for it. A product's role must be defined before the base and optional function sets can be chosen. The base subset of each of the base function sets must be implemented to define the role. For example, in Figure 128, the base function sets for role *x* are function set *a* and function set *b*. The base subset of each of the two base function sets must be implemented to define role *x*. Optional function sets can be implemented depending on product considerations. Defining base function sets for each role ensures that the network is provided with the functions it needs, that network components can connect to each other, and that products do not have unneeded functions.

Electives

Certain functions can be implemented in more than one way. If the effect of the choice can be observed at the MS protocol boundary, then that choice is called an ***elective***. Electives are not optional functions, but are choices about how or when a function is provided. If another component can observe the effect of an elective choice, then that component must be able to support all of the possible effects of the elective choices. Product implementations make elective choices for performance or development-cost reasons.

Components of Management Services

Management services involves three node components: local management services, physical unit management services, and control point management services. These components interact with the network operator to perform their MS functions.

Local Management Services

Local management services (LMS) is the portion of each node component, or node layer, that is dedicated to MS-related functions. LMS exists within most layers of the node. LMS:

- Executes MS commands received from a centralized MS component within the node regarding a specific node component. In a T2.1 node, LMS receives commands from the control point management services (CPMS) component. In all other node types, LMS receives commands from the physical unit management services (PUMS) component.
- Sends solicited and unsolicited MS information to the CPMS or PUMS in its own node, reporting on a particular node component.

Physical Unit Management Services

Physical unit management services (PUMS) is a component of the physical unit (PU) in a T5, T4, or T2.0 node. (PUMS exists only in subarea networks.) PUMS provides entry point services for the node in which it resides. The PUMS component:

- Exchanges MS data with LMS within the node, sending MS commands to LMS and receiving solicited and unsolicited MS data from LMS.
- Exchanges MS data with the CPMS component in the system services control point (SSCP) controlling its domain, receiving MS commands from CPMS and sending solicited and unsolicited MS data to it.

Control Point Management Services

Control point management services (CPMS) is a component of the control point in a T5 or T2.1 node. The functions of CPMS depend on type and role of the node in which it exists. In a T2.1 node in a subarea network (acting as a peripheral node), CPMS has the same functions as PUMS. In a T5 node in a subarea network, CPMS:

- Exchanges MS data with the PUMS and CPMS components in the peripheral nodes in its domain and the PUMS component its own node. CPMS sends MS commands to those components and receives solicited and unsolicited MS data from them.

- Exchanges MS data with the ***NetView*** product; CPMS receives MS commands from the NetView program and sends solicited and unsolicited MS data to it. In a given T5 node, the NetView program may or may not be serving as a focal point. When serving as a focal point, the NetView program interacts directly with a network operator. Otherwise, it plays a supporting role by routing MS data between the CPMS component in its own node and the NetView program that serves as a focal point in another T5 node.

In an APPN network, every APPN node contains CPMS. The CPMS component in each node monitors and controls LMS in the various layers of the node. Other functions of CPMS depend on whether it is in an APPN end node or network node. In an APPN end node, CPMS acts only as an entry point. In an APPN end node, CPMS:

- May exchange MS data with CPMS in one or more focal points, or, alternatively,
- May exchange MS data with just the CPMS in its network node server. In this case, the network node server routes the MS data between the end node and the appropriate focal points.

In a network node, CPMS can act as an entry point or a focal point. In an APPN network node, CPMS:

- When serving as an entry point, exchanges MS data with CPMS in one or more focal points. It may also exchange MS data with CPMS in a client APPN end node, if the end node has elected not to communicate directly with the focal points.
- When serving as a focal point, exchanges MS data with entry points and with a network operator.

Management Services Communication

Entry points and focal points together manage the flow of MS data in an SNA network. There are two kinds of MS data that they manage: bulk MS data and non-bulk MS data.

Bulk MS data refers to the large files and reports exchanged by the change management category of management services. One use of bulk data transmission is the transport of files associated with the change management category of SNA/MS. For example, a focal point might send microcode program changes to an entry point for installation.

Non-bulk MS data refers to all other MS message units exchanged during the process of network management. Focal points transmit (at operator request) commands and requests for MS information to the entry points in their spheres of control, and entry points transmit solicited and unsolicited MS information to focal points.

Subarea and APPN networks differ in the way they transmit MS data.

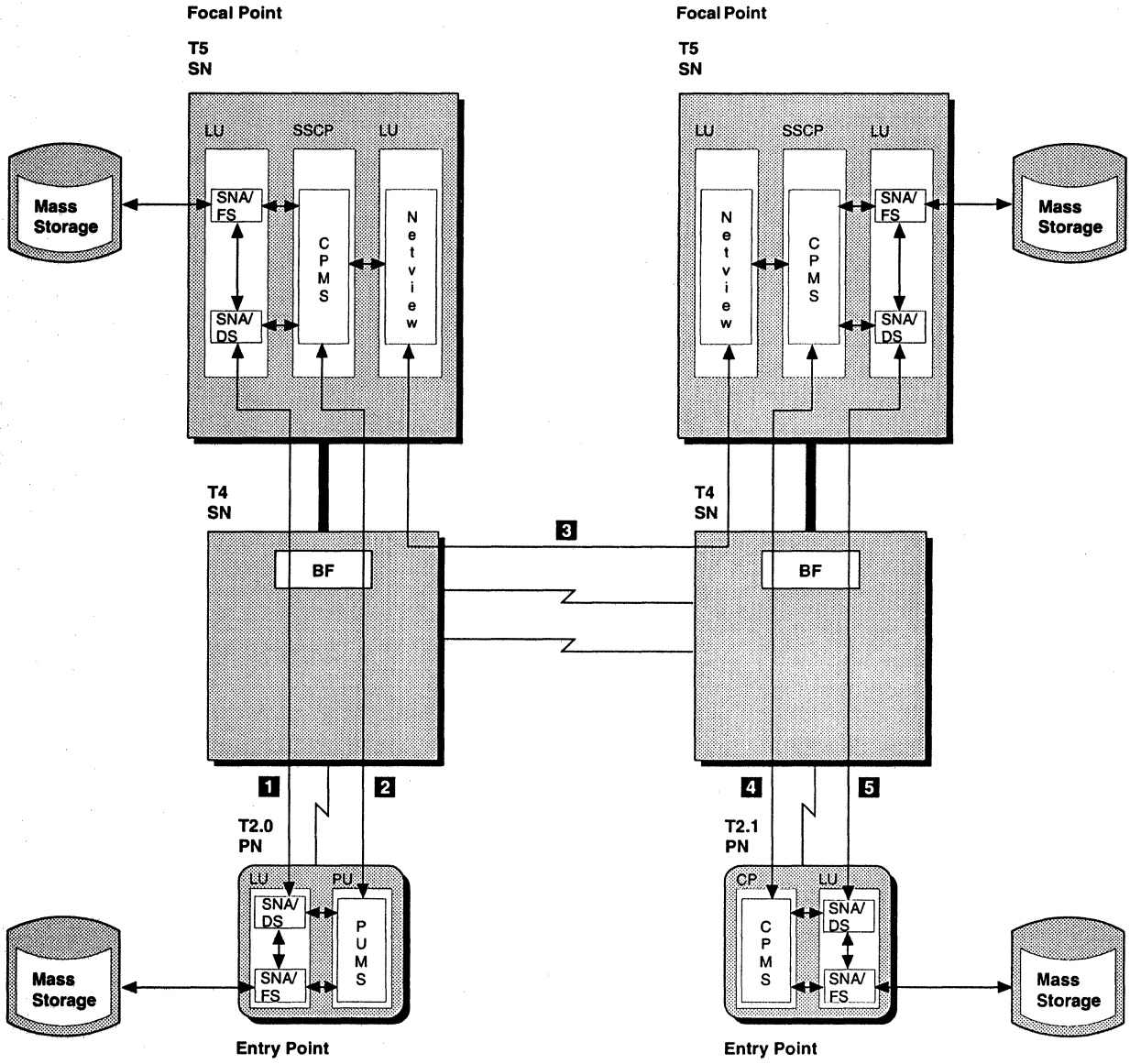
MS Communication in Subarea Networks

Among IBM products, the focus of management services in a subarea network is the NetView family of products, which is network management software that runs on a host processor. The NetView program is the key product for centralizing management services although it may provide management services from a central or remote site.

Bulk MS data in a subarea network is exchanged between CPMS in a T5 node and PUMS or CPMS in a peripheral node. CPMS and PUMS act as SNA/DS agents, using SNA/DS and SNA/FS to accomplish the data transfer and storage. SNA/DS transmits and receives bulk data on type 6.2 LU-LU sessions.

The sessions used to transmit non-bulk MS data depend on whether the transmission is within a domain (intradomain) or between domains (interdomain). Intradomain non-bulk MS data is exchanged between a T5 node and the T4 and peripheral nodes in the T5 node's domain, over the SSCP-PU sessions initiated during network activation. Interdomain non-bulk MS data is exchanged between T5 nodes over sessions established between instances of the NetView program residing in the nodes. (These sessions use protocols specific to the NetView product.)

Figure 129 on page 286 illustrates management services message unit flows in a subarea network.



Legend:

BF	= Boundary Function
CP	= Control Point
CPMS	= Control Point Management Services
LU	= Logical Unit
PU	= Physical Unit
PUMS	= Physical Unit Management Services
SN	= Subarea Node
SNA/DS	= SNA Distribution Services
SNA/FS	= SNA File Services
SSCP	= System Services Control Point

1 Bulk data transfer (LU-LU session)

2 Non-bulk NMVT data transfer (SSCP-PU session)

3 Non-bulk data transfer (NetView-NetView session)

4 Non-bulk NMVT data transfer (SSCP-PU session)

5 Bulk data transfer (LU-LU session)

Figure 129. Management Services Flows in a Subarea Network

In Figure 129, flows **1** and **4** represent the transmission of bulk MS data over SNA/DS LU-LU sessions. Flows **2** and **3** represent the intradomain transmission of non-bulk MS data over SSCP-PU sessions. Flow **5** represents the interdomain transmission of non-bulk MS data between instances of the NetView program.

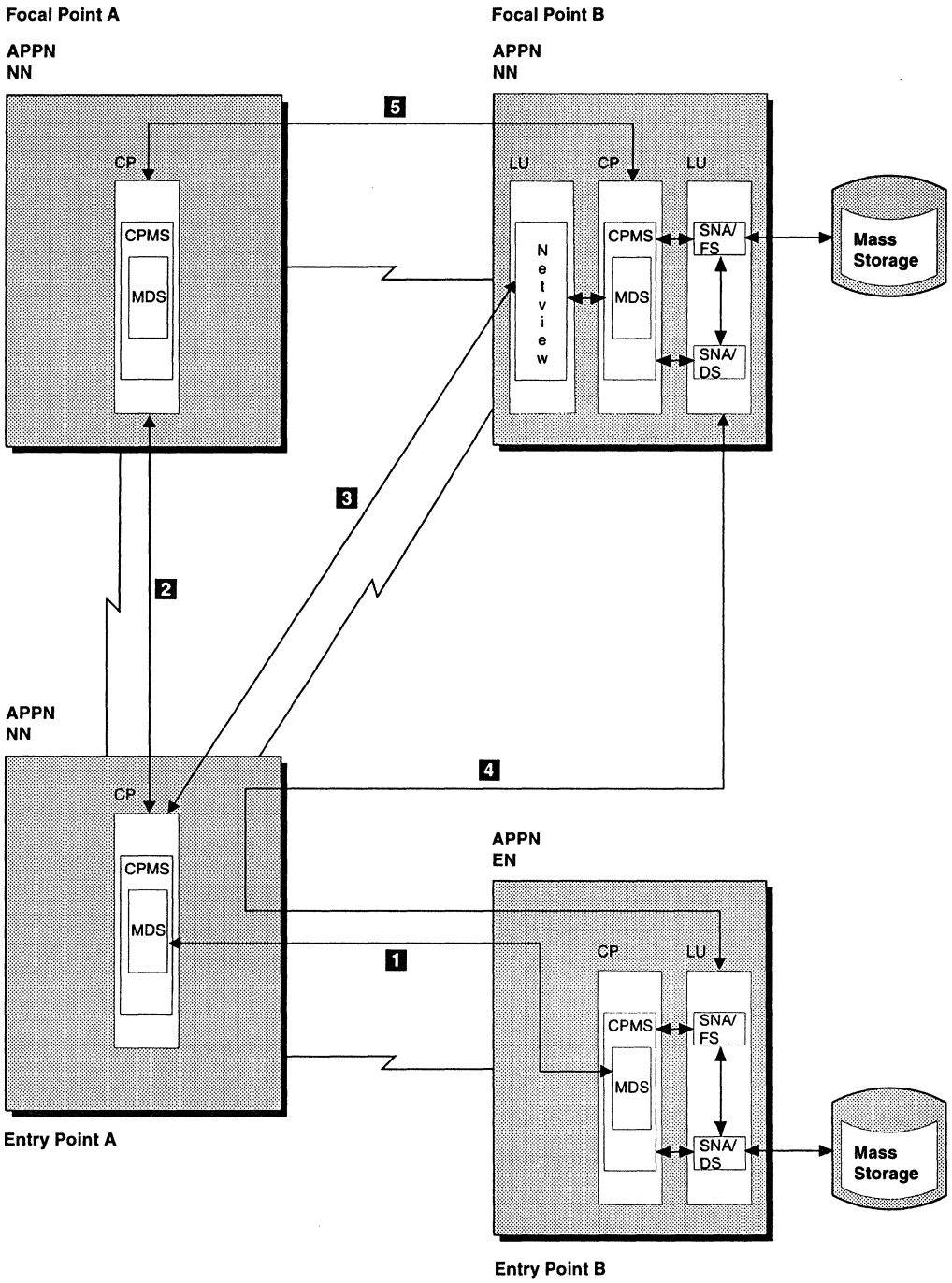
MS Communication in APPN Networks

As in a subarea network, bulk MS data in an APPN network is exchanged by SNA/DS over type 6.2 LU-LU sessions. CPMS uses SNA/DS agents in the nodes exchanging bulk data. There is very little difference other than the components (CPMS vs. PUMS) involved between the transmission of MS bulk data in APPN networks and subarea networks.

Non-bulk MS data in an APPN network is most frequently transmitted over LU-LU sessions. Between an APPN end node (EN) and its network node (NN) server, however, it can also be transmitted over the CP-CP sessions established between these nodes following link activation. The use of CP-CP sessions for non-bulk MS data transmission serves two purposes. First, an additional session for MS traffic need not be activated between an end node and its network node server. Second, the NN server can route MS message units between its attached APPN ENs and the focal point nodes This *session concentration* lessens the session overhead on both entry points and focal points. ENs need not initiate sessions with focal points for the exchange of non-bulk MS data. They can instead exchange the data with their network node servers. Likewise, a focal point need not have a session with every entry point in its sphere of control. Instead, it can exchange non-bulk MS data with network nodes, each of which can route the data between its client APPN end nodes and the focal point.

In an APPN network, non-bulk MS data is exchanged between management services application programs residing in different nodes. ***Management services application programs*** are architecturally defined application transaction programs that perform management services functions. The service provided by CPMS that provides for the routing of data between MS application programs over CP-CP and LU-LU sessions is called ***multiple-domain support*** (MDS). MDS consists of a router and multiple service transaction programs. The ***MDS router*** routes message units between MS application programs residing in the same node and uses the MDS service transaction programs to route message units between MS application programs residing in different nodes. The MDS router resides in the control point. An ***MDS service transaction program*** (STP) provides an interface to presentation services for transmitting MS message units over sessions between nodes. MDS STPs reside in both control points and logical units.

Figure 130 illustrates management services message unit flows in an APPN network.



Legend:

CP	= Control Point
CPMS	= Control Point Management Services
EN	= End Node
LU	= Logical Unit
MDS	= Multiple-Domain Support
NN	= Network Node
SNA/DS	= SNA Distribution Services
SNA/FS	= SNA File Services

1. Non-bulk MDS-MU data transfer (CP-CP session)
2. Non-bulk MDS-MU data transfer (LU-LU session)
3. Non-bulk MDS-MU data transfer (LU-LU session)
4. Bulk data transfer (LU-LU session)
5. Non-bulk MDS-MU data transfer (LU-LU session)

Figure 130. Management Services Flows in an APPN Network

In Figure 130, flow **1** represents the transmission of non-bulk MS data over the CP-CP sessions between APPN end node, entry point B, and its network node server, entry point A. Entry point A routes MS data between entry point B and the two focal points. This eliminates the need for LU-LU sessions between entry point B and the focal points. Flows **2** and **3** represent the transmission of non-bulk MS data over LU-LU sessions. Focal points A and B handle different MS categories. Flow **4** represents the transmission of bulk MS data over a SNA/DS LU-LU session. Flow **5** represents the transmission of non-bulk MS data over LU-LU sessions between focal point A and focal point B.

MS Communication in Interconnected Subarea and APPN Networks

MS data can be exchanged between subarea and APPN networks over a link connecting a T4 node to an APPN network node. This configuration enables the NetView program, running in a T5 node, to manage the composite network. Figure 131 on page 290 illustrates management services message unit flows in a network consisting of interconnected subarea and APPN networks.

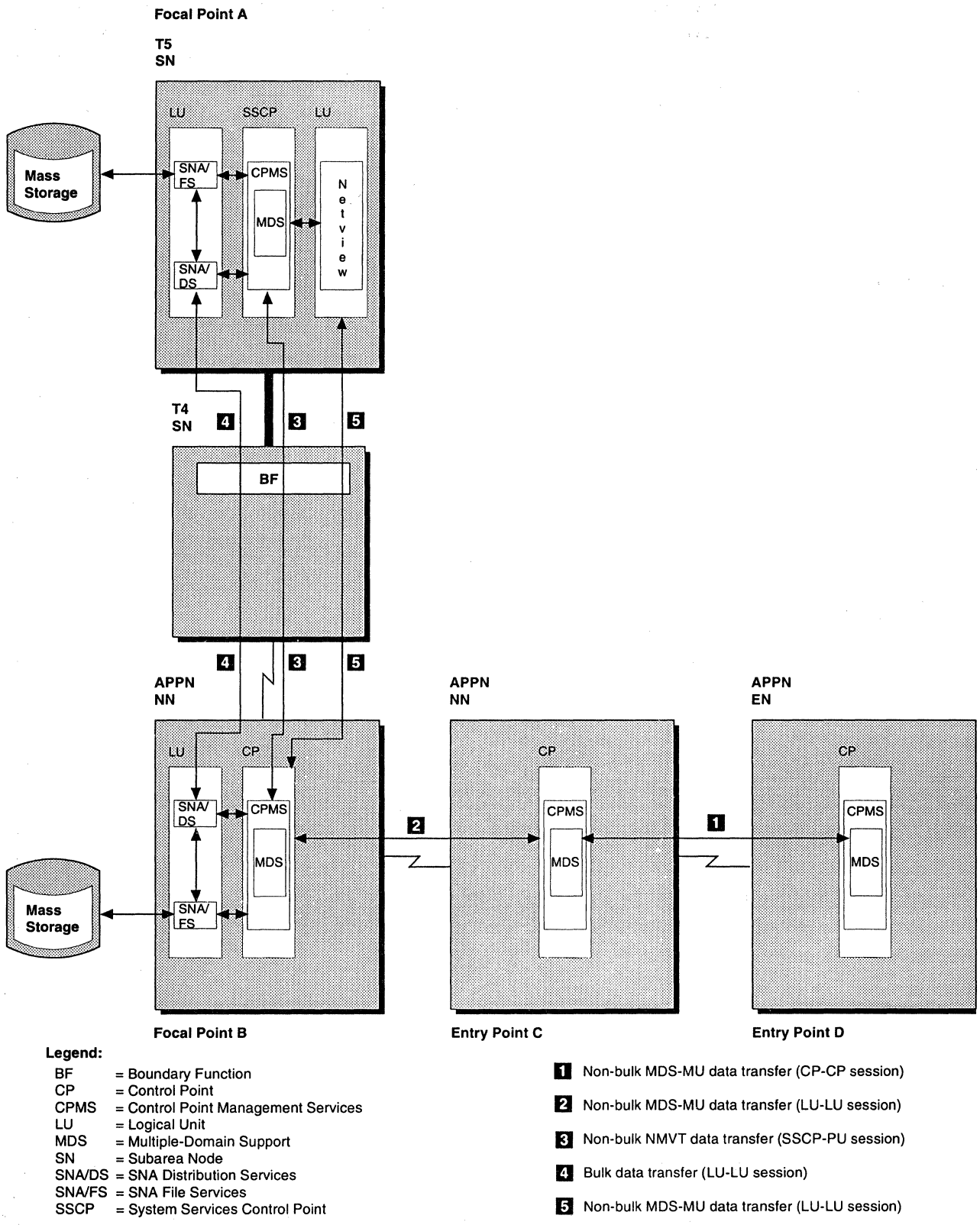


Figure 131. Management Services Flows in Interconnected Subarea and APPN Networks

In Figure 131, flow **1** represents the transmission of non-bulk MS data over the CP-CP sessions between APPN end node, entry point D, and its network node

server, entry point C. Flow **2** represents the transmission of non-bulk MS data over LU-LU sessions between entry point C and focal point B. Flow **3** represents the transmission of non-bulk MS data over SSCP-PU sessions between focal point B and focal point A. This flow enables network management to be centralized at focal point A. (It is also possible for focal point A to communicate directly with entry points C and D over LU-LU sessions.) Flow **4** represents the transmission of bulk MS data over a SNA/DS LU-LU session between focal point B and focal point A. Flow **5** represents the transmission of non-bulk MS data over LU-LU sessions between focal point B and focal point A.

Management Services Message Units

Two message unit formats are devoted exclusively to the transmission of MS data: the network management vector transport message unit, and the multiple-domain support message unit. The ***network management vector transport*** (NMVT) message unit is used for the transport of non-bulk MS data on SSCP-PU sessions in subarea networks. The ***multiple-domain support message unit*** (MDS-MU) is used for the transport of non-bulk MS data in APPN networks and may be used in a subarea environment. Bulk MS data is transported in both subarea and APPN networks in the distribution transport message unit (DTMU) used by SNA/DS. For a layout of the DTMU, see "SNA/DS Request Unit Formats" in Chapter 9, "Transaction Services."

Management Services Major Vector

Both the NMVT and the MDS-MU carry the management services major vector. The ***management services major vector*** is an MS encoding that contains the MS data. It is formatted according to a major vector, subvector, subfield scheme. Figure 133 illustrates the format of the management services major vector.

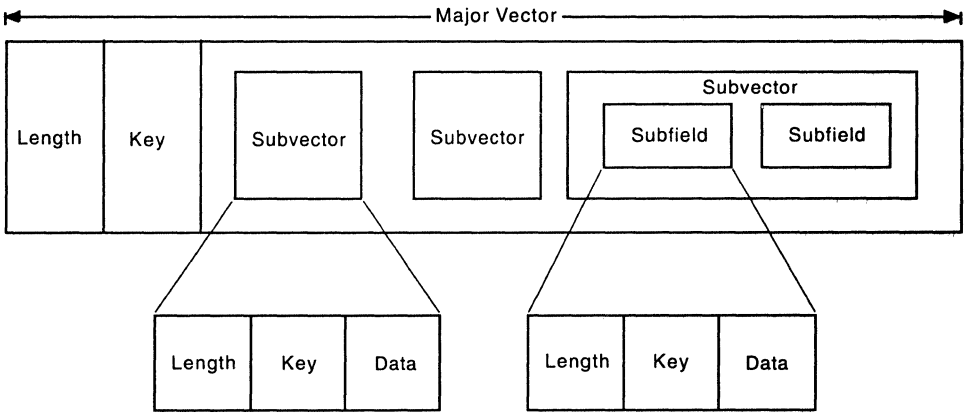


Figure 132. The Management Services Major Vector Format

The ***length*** portion of the major vector identifies how long the vector is, and the ***key*** portion identifies the general type of MS data that the major vector contains. One type of data might be, for example, problem determination data. The MS data is further divided into ***subvectors***, which contain specific pieces of MS data. Each subvector also contains length data and a key that identifies the kind of data to follow. Although the subvectors contain specific MS data, some subvectors are broken down again into even more specific data called ***subfields***. Each subfield also contains length, key and data elements that serve the same purpose as they do in the subvector.

Network Management Vector Transport Format

The NMVT message unit has two components: the header and the management services major vector. Figure 133 illustrates the format of the NMVT message unit.

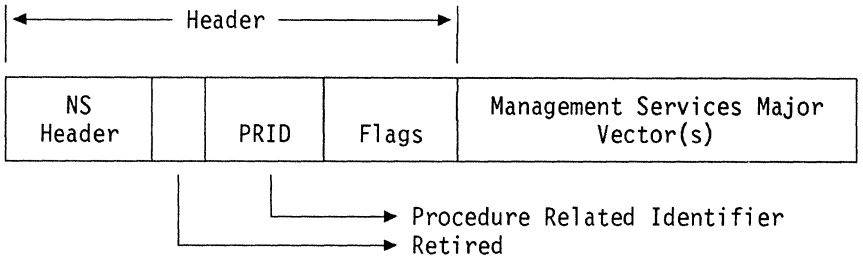
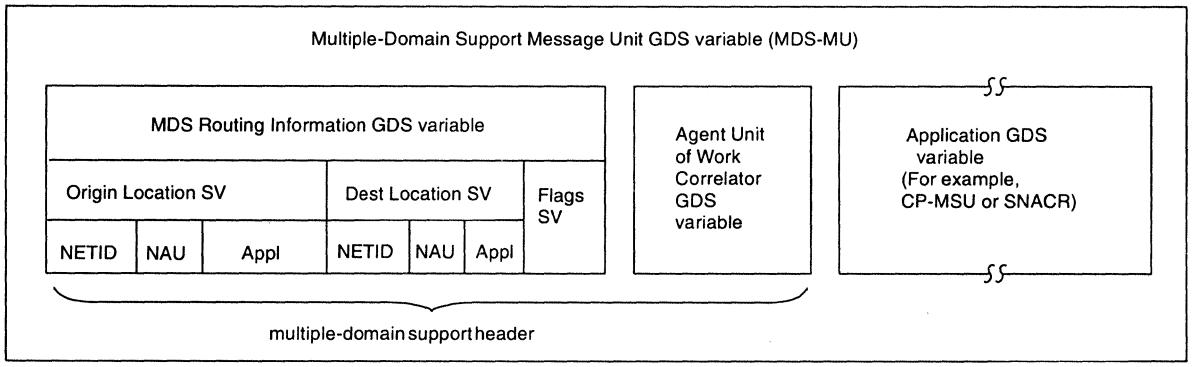


Figure 133. The NMVT Message Unit Format

The ***NMVT header*** consists of three parts: the NS header, the procedure related identifier, and the flags. The ***NS header*** identifies the message unit as an NMVT. The ***procedure related identifier*** (PRID) is used by the sender of an NMVT request to correlate the response with the request. The ***flags*** provide additional identification of the NMVT.

Multiple-Domain Support Message Unit Format

The MDS-MU has two components: the multiple-domain support header and the MS application program data. Figure 134 illustrates the format of the MDS-MU.



Legend:

Appl = Application Program Name
CP-MSU = Control Point Management Services Unit
GDS = General Data Stream
NAU = Network Accessible Unit Name
NETID = Network ID
SNACR = SNA Condition Report
SV = Subvector

Figure 134. The Multiple-Domain Support Message Unit Format

The ***multiple-domain support header*** consists of two general data stream (GDS) variables, the MDS routing information GDS variable, and the agent unit of work correlator GDS variable. The ***MDS routing information GDS variable*** contains data required for MDS routing, including the origin and destination location names. The ***agent unit of work correlator GDS variable*** provides a unique value for the MS application program that assigns it. This enables both MDS and the MS application programs to correctly correlate MDS-MUs.

The ***MS application program data*** is a GDS variable supplied by the MS application program. It may be a control point management services unit GDS variable, an SNA condition report GDS variable, or some other GDS variable.

The ***control point management services unit*** (CP-MSU) is a GDS variable that can be used for transmitting either non-bulk MS data in the MDS-MU or a SNA/DS agent object in a DTMU. Like the NMVT, the CP-MSU carries the management services major vector. Figure 135 illustrates the format of the CP-MSU.

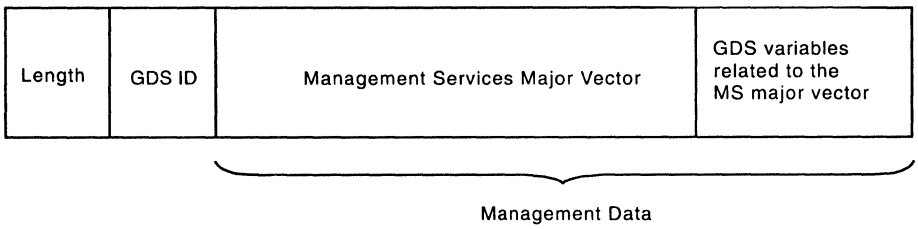


Figure 135. The CP-MSU GDS Variable Format

APPN Topology and Accounting Management (APPNTAM)

APPNTAM is a feature of NetView V2R4. With APPN topology and accounting management, the user is provided with two important facilities for managing APPN network resources from a NetView platform. APPN topology and accounting management implement topology monitoring with graphic display and accounting data collection for LU 6.2 resources. For more information on APPNTAM, refer to the following publications: *NetView APPNTAM Feature Agent Implementation Guide*, *NetView APPNTAM Feature Implementation Guide*, *NetView APPNTAM Feature Diagnosis*, and *NetView APPNTAM Feature Topology Data Model Reference*.

Manager-Agent Applications

The manager applications for APPN topology and accounting management are NetView applications. Agent applications that collect information for transmission to NetView reside on APPN end nodes and network nodes that use the OS/2 Communications Manager/2 platform.

APPNTAM CMIP Services

CMIP services is the component of APPNTAM that provides the communications support between the manager and agent applications. CMIP services communications is over LU 6.2 sessions and uses the OSI common management information Protocols (CMIP) and MS transport. The CMIP messages are encapsulated in MDS-MUs and routed between the manager and agent nodes using MDS routing.

APPN Topology Management

APPN topology manager provides a dynamic, centralized network management application for APPN networks. APPN topology management function provides a composite view of an APPN network, by graphically showing both the individual status of a resource and its relationship to other network resources, thus reducing problem determination time for error conditions.

The APPN topology manager uses the existing NetView components, including Resource Object Data Manager (RODM) and Graphic Monitor Facility Host Sub-

system (GMFHS), to manage and display APPN topology data at the NetView Graphic Monitor Facility (NGMF) OS/2 workstation. Data is stored in RODM dynamically and can be used for automation.

RODM— Is an object-oriented data cache for storing and manipulating data. Objects representing nodes, links, ports, and connections in an APPN network are defined to RODM according to the *APPNTAM Topology Data Model*. Details of this model are available so users can write automation programs and create their own objects. For more information on RODM, refer to *NetView Resouce Object Definition Manager Programming Guide*.

NGMF— Is used to display configuration and status in graphic views. NGMF uses the Command Tree/2 component of the NGMF workstation for command support. Operators use these views to monitor the status of the APPN network, navigate through the network, and take action on failed resources. APPN views are built dynamically, which ensures the most current status and configuration are displayed to the operator. This is especially important for APPN networks since they can change configuration frequently as nodes activate connections and enter the network and as nodes deactivate connections and leave the network. As changes occur in the network, the views are updated. Operators are informed of changes through status color changes, configuration changes, and messages (when operators refresh their views).

APPN topology manager collects two types of topology:

- Network topology, which displays information about network nodes and the transmission groups (TGs) that connect them.
- Local topology, which displays information about the following:
  - LEN and APPN nodes
  - The connections between nodes
  - The ports and links that make up the connections.

Figure 136 illustrates an overview of the topology manager functions.

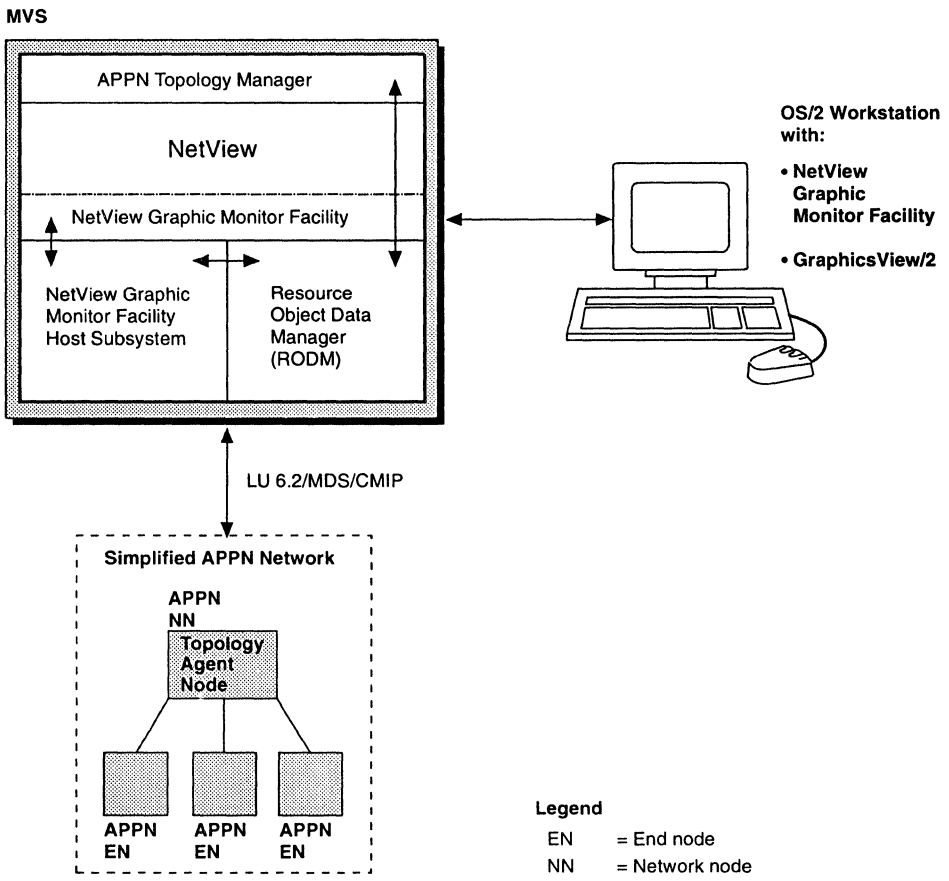


Figure 136. APPN Topology Manager Functional Overview

Topology Manager-Agent Relationship

The topology manager application works with one or more topology agents to gather topology information for the APPN network. Agent applications can be on APPN network nodes and APPN end nodes; APPN network nodes provide network and local topology, and APPN end nodes provide local topology.

When an operator issues a command to start monitoring topology (network or local), the topology manager sends a request to the agent. The agent application supplies APPN topology information about nodes and links in response to requests from the manager application, then continues to send topology updates (including status and network-configuration changes) to the manager as they occur. The data is stored and updated using NetView V2R4 host components, including RODM and GMFHS. These components provide the data for views displayed using NGMF. The pull-down menus, commands and view management functions available with NMGF are used with the APPN topology manager application, where applicable. The agent also handles activating and deactivating ports and links when it receives the commands from the manager. Figure 137 illustrates the topology agent notifying the topology manager of an update to the network topology. The topology agent sends the topology database, followed by a topology update, to the topology manager.

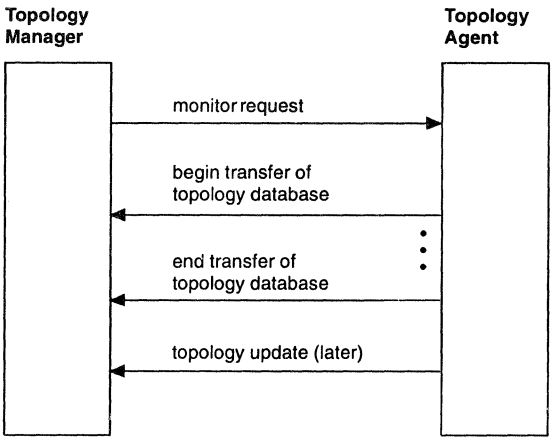


Figure 137. How Topology Agent Notifies Topology Manager of Topology Update

APPN Topology Manager Functions

In summary, the APPN topology manager provides the following functions:

- Monitors APPN network topology to view the connectivity between APPN network nodes
- Monitors APPN local topology to view the APPN agent node and its TGs, ports, logical links, and adjacent nodes
- Manages activation and deactivation of ports and links
- Navigates from high-level aggregate views to real resources
- Displays views of an APPN network, such as APPN subnetworks and their nodes, TGs, links, and ports, with successive detail from one view to the next
- Displays information about resources such as CP and link names
- Identifies which TGs have CP-CP sessions
- Identifies which network nodes, end nodes, and TGs have additional capabilities and displays them
- Uses existing NGMF functions to navigate and edit views
- Enables automated operations using RODM objects and messages
- Enables creating user-defined objects and views in RODM for customized operations.

APPN Accounting Management

APPN accounting manager retrieves accounting data from agents for APPC sessions and conversations. The accounting manager then formats the data and writes it to an external log, using the system management facilities (SMF) log or a user-defined logging facility. Possible uses for this data include usage reporting and billing. The APPN accounting function collects session and conversation data such as when the session or conversation began, when and why it ended, and the number of bytes sent and received.

The accounting manager application works with a corresponding agent application to retrieve accounting data. The agent application collects APPC accounting information in response to requests from the manager application. Scheduled times or intervals can be set for the manager to retrieve accounting data from a specified agent node. Also, thresholds can be established (percent of agent buffer full, or number of records in buffer) for triggering the manager to retrieve the data.

Accounting session data can be collected at an endpoint or an intermediate node in the session path. Conversation data can be collected only at an endpoint. The type of data to be collected by an agent is specified by the accounting manager application.

Accounting Manager-Agent Relationship

Accounting data is transferred between an accounting agent and an accounting manager based on a series of notifications and requests. In each case, the data transfer applies to a specified type of data; that is, conversation data, session data, or intermediate session data. For example, a user specifies that the manager should be notified when the conversation data buffer at the agent becomes 80 percent full. As shown in Figure 138, when the agent has collected enough data to fill its buffer to the specified threshold, the agent sends a notification to the manager. The manager then retrieves the accounting data from the agent. Figure 138 illustrates the agent sending a notification to the manager when the buffer threshold is reached.

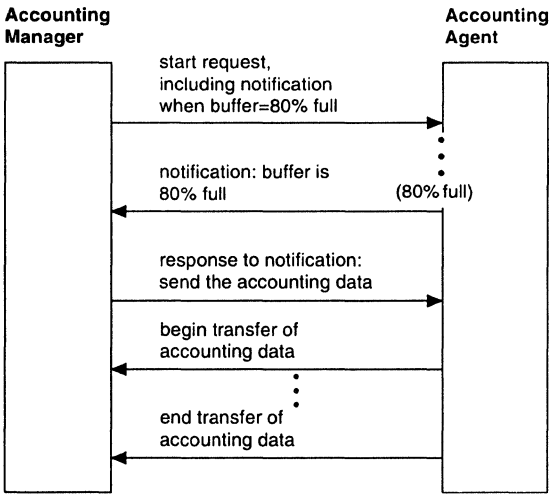


Figure 138. How Data Flows When Agent Buffer Reaches Threshold

The following list contains some of the session accounting information that is collected:

- Start and stop times at the session endpoint or intermediate node
- Session endpoint names (PLU and SLU)
- Mode name and class of service
- Route of session (RSCV)
- Fully qualified PCID (FQPCID)
- Number of PIU bytes, FMD and non-FMD PIUs
- Reason for ending the session.

The following list contains some of the conversation accounting information that is collected:

- Start and stop times at the endpoint of the conversation
- Type (basic or mapped)
- Conversation endpoint names (source and target LUs)
- Synchronization level
- TP name
- Fully qualified PCID (FQPCID) of underlying session
- Conversation security user ID

- Logical unit-of-work ID
- Conversation correlator
- Number of FM data RU bytes sent and received
- Reason for ending the conversation.

Figure 139 illustrates how data is collected at the accounting agent and sent to the accounting manager at NetView. Assume the NetView operator has requested that session and conversation data be collected at the agent. The agent then creates separate accounting management control objects (AMCOs) to manage the collection of session endpoint data, intermediate session data, and conversation data. The AMCOs contain all the instructions for managing the data collection, such as under what conditions to notify the manager that data is ready to be retrieved.

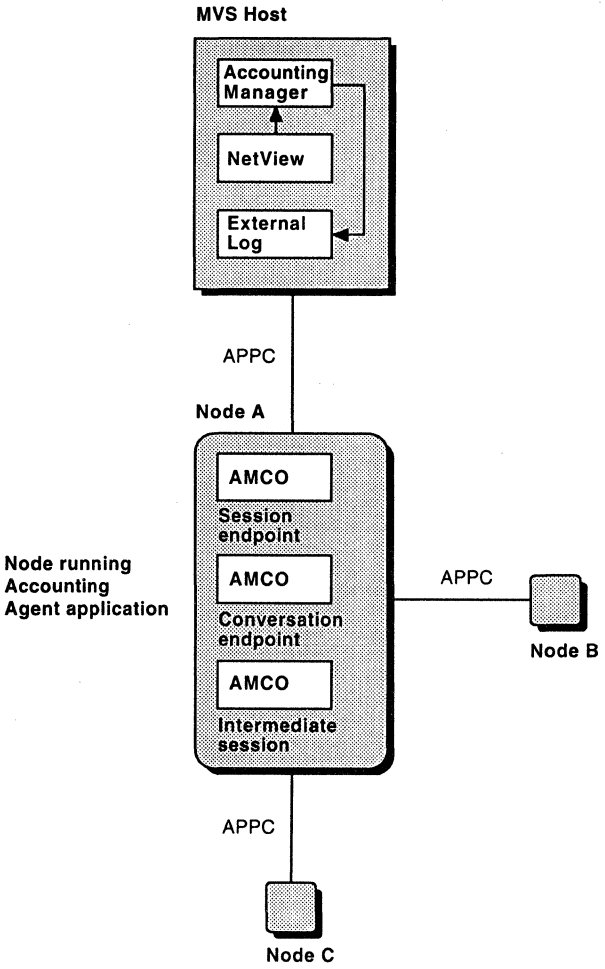


Figure 139. How the Accounting Function Works

APPN Accounting Manager Functions

In summary, the APPN accounting manager provides the following functions:

- Starts collecting accounting data at an agent node for a particular type of data
- Retrieves data from an agent
- Stops collecting a particular type of accounting data at a given agent node
- Modifies defaults, display information about data collection, and other maintenance tasks.

Benefits of APPNTAM

The APPNTAM functions provide the following benefits:

- Expanded scope of management for APPN resources

  The dynamic graphic display of APPN topology simplifies the task of gathering resource information and allows the network operator to concentrate on the actions required to effectively manage network resources. The graphic view of the APPN network provides the individual status of a resource and its relationship to other network resources which reduces problem determination time for error conditions.

  APPC accounting allows NetView to provide users with the capability of tracking usage to a specific APPN resource and predicting growth requirements in the network.

- Consistency with SystemView*.

  NetView expands its SystemView automation services to enable automation of APPN resources by storing APPN objects in RODM. These NetView applications are based on the OSI CMIP open standard.

- Increased user productivity

  Through the use of NGMF, NetView provides the network operator with a concise, easily understood view of network resources and the current status of each selected component. NGMF masks the underlying complexities necessary to convey status information, while supplying the operator with the control facilities required to activate and deactivate ports and links either manually or by using NetView's automation facilities.

- Ability to integrate APPN networks with existing SNA subarea networks

  The growth of dynamic, peer networks in a centrally managed environment requires a strategy tailored to the new technology and the investment applied to current network operations. The NetView managing system applications provides the APPN topology and APPC accounting functions on an existing product base. In addition, the accounting application optionally uses SMF for storing accounting records, thus preserving the investment in service level reporting.

Appendix A. Sequence Charts

Typical Request Unit Sequences for Activating and Deactivating Network Resources

Figure 141 through Figure 151 present typical request unit sequences for activating and deactivating various portions of a network. Figure 140 gives the meaning of each of the symbols and abbreviations that appear in these request unit sequences. Chapter 5, "Activating the Network" explains how control points activate and deactivate network resources.

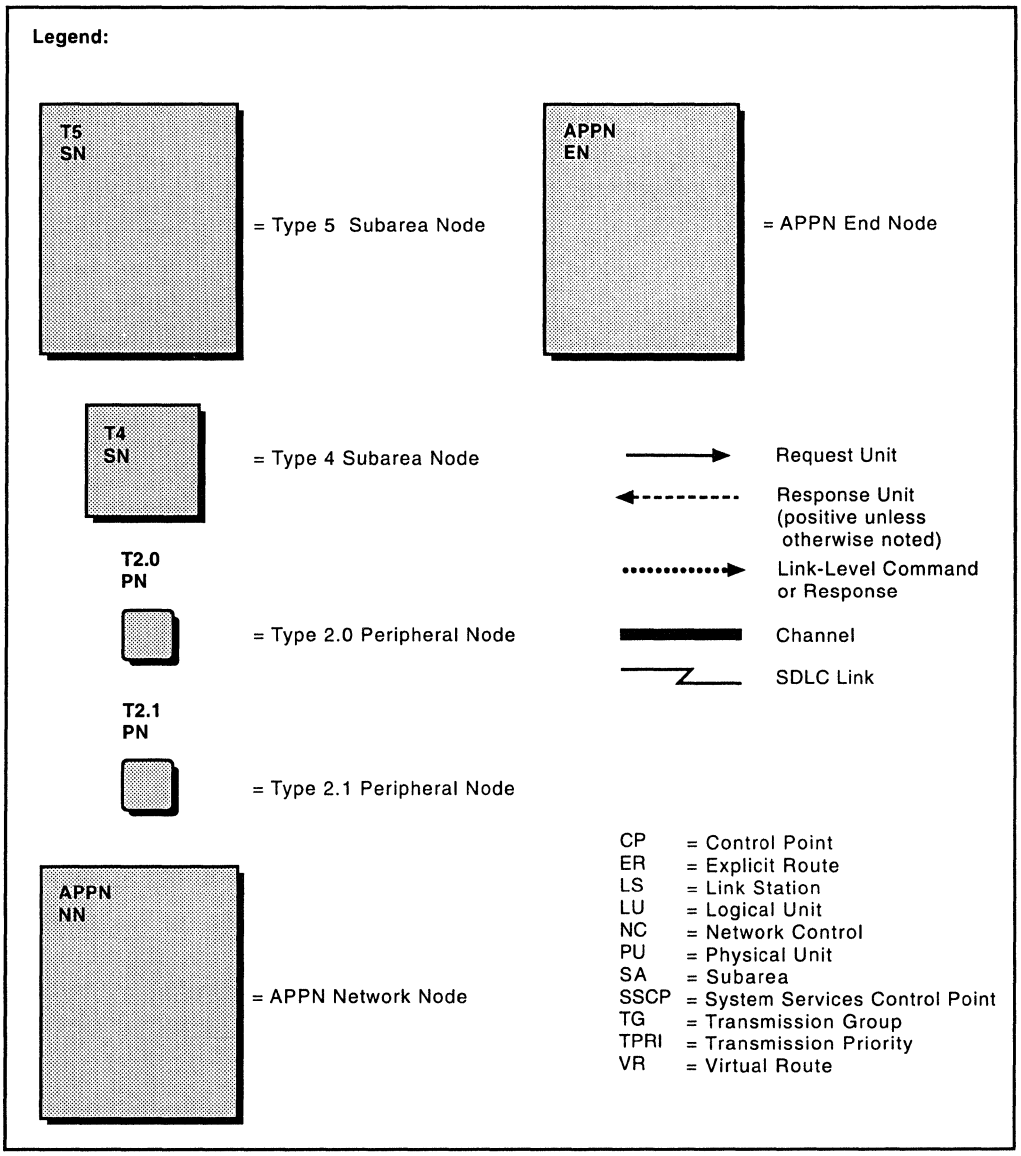
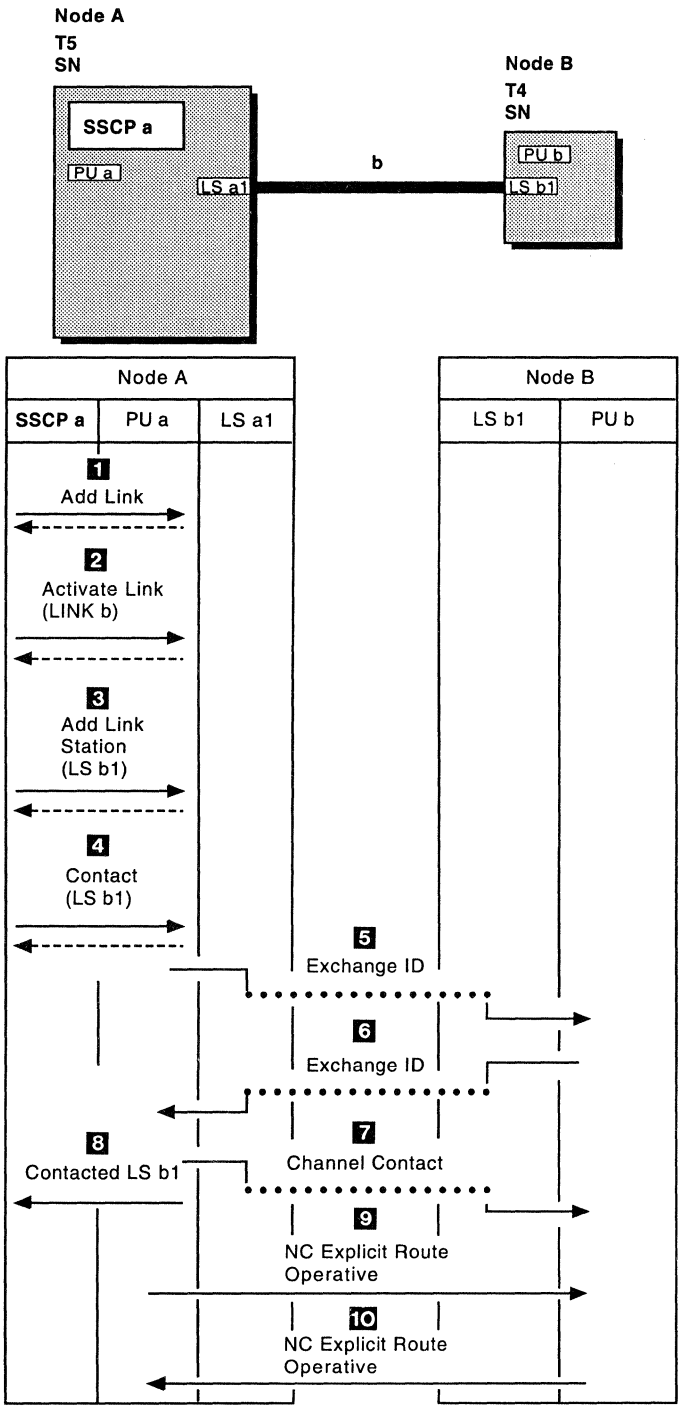
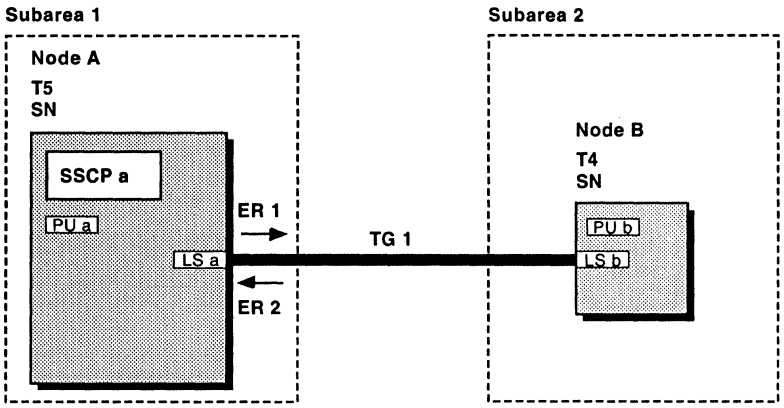


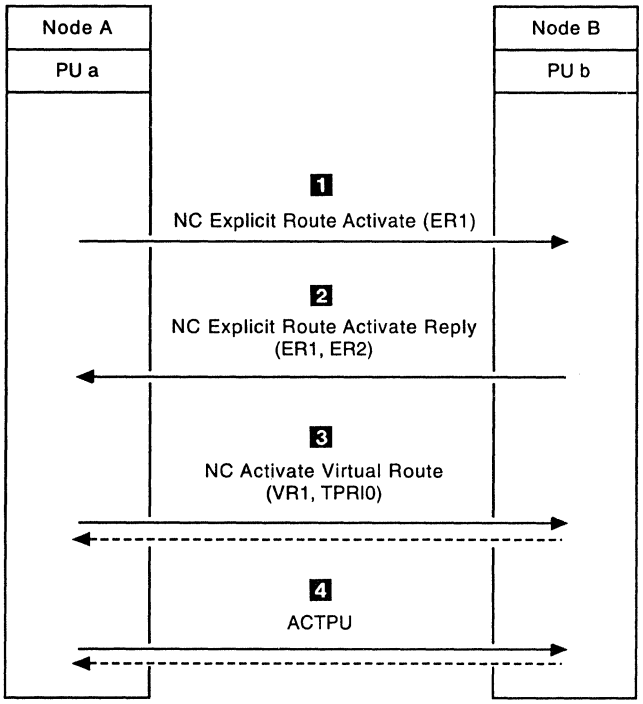
Figure 140. Symbols and Abbreviations for Figures 141 through 151



1. SSCP a requests PU a to furnish SSCP a with a network address for the designated link connection. PU a does so on the response.
2. SSCP a tells PU a to activate LINK b and to prepare to issue and receive data link control commands and responses for the link connection.
3. SSCP a requests PU a to furnish SSCP a with a network address for the designated link station. PU a does so on the response.
4. SSCP a tells PU a to contact the adjacent link station LS b1. The representation of the link station in Node A has network address of a1.
5. PU a sends PU b information about Node A, including the maximum number of bytes that Node A will accept across the channel at one time.
6. PU b informs PU a that the parameters sent by PU a are acceptable, and sends PU a information about Node B.
7. PU a completes the activation of LINK b by accepting the parameters sent by PU b.
8. PU a informs SSCP a that message units can now be sent to PU b through link station LS a1.
9. PU a tells PU b which subareas can be reached from Node A, and which explicit routes are used to reach these subareas.
10. PU b tells PU a which subareas can be reached from Node B, and which explicit routes are used to reach these subareas.

Figure 141. Activating a T5 Node, a Channel-Attached T4 Node, and the Channel Between Them





1. PU a initiates the activation of an explicit route between subarea 1 and subarea 2. This route has an explicit route number of 1. Activation of an operative but inactive explicit route is initiated when the route is needed to satisfy a session activation request.

2. PU b completes activation of the explicit route between subarea 1 and subarea 2, replying that the reverse explicit route number for this explicit route is 2. (ER1 and ER2 in this case refer to the same explicit route; an explicit route is known by two explicit route numbers - one for each direction.) PU b indicates to PU a that the explicit route has a length of one transmission group. This length is used in determining the pacing group size for virtual route pacing.

3. PU a activates a virtual route between subareas 1 and 2. This virtual route, which has a virtual route number of 1 and a transmission priority of 0, uses the explicit route identified by explicit route numbers 1 (in the Node A-to-Node B direction) and 2 (in the reverse direction). An inactive virtual route is activated when it is needed to satisfy a session activation request.

4. SSCP a activates an SSCP-PU session with PU b over the explicit and virtual routes just activated.

Figure 142. Activating Explicit and Virtual Routes between Adjacent Subarea Nodes

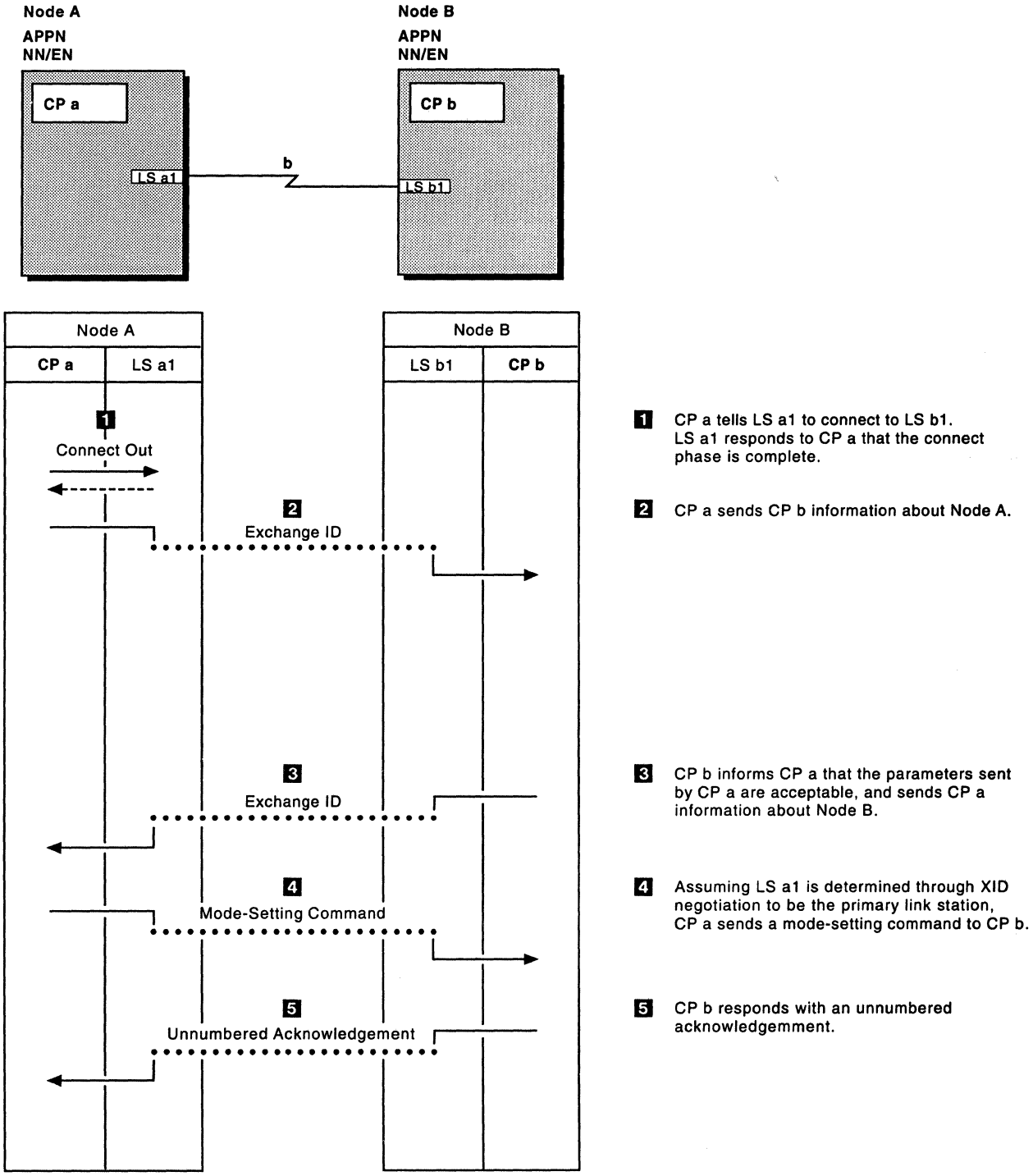
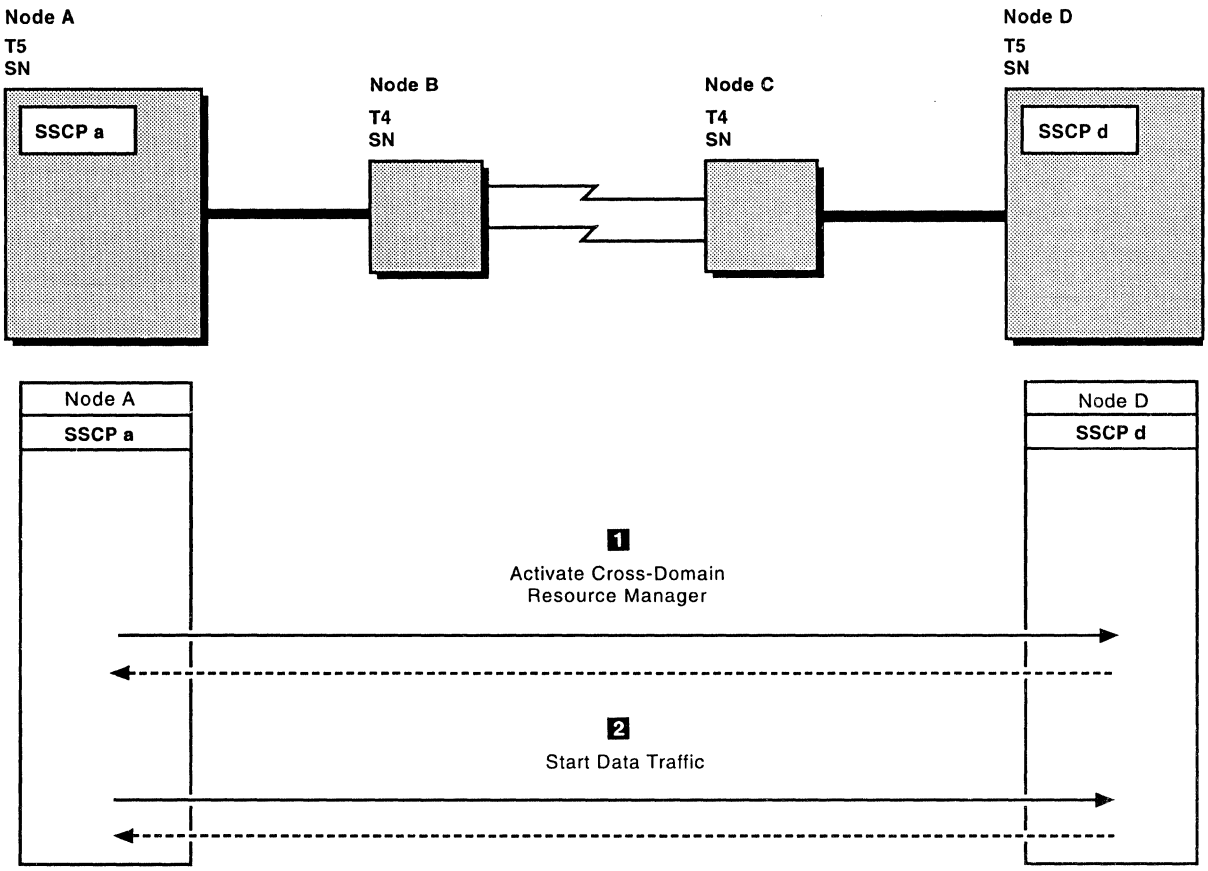


Figure 143. Activating a Nonswitched SDLC Link Between APPN Nodes



1 SSCP a activates a session with SSCP d.

2 SSCP a enables the flow of message units over its SSCP-SSCP session with SSCP d.

Figure 144. Activating an SSCP-SSCP Session

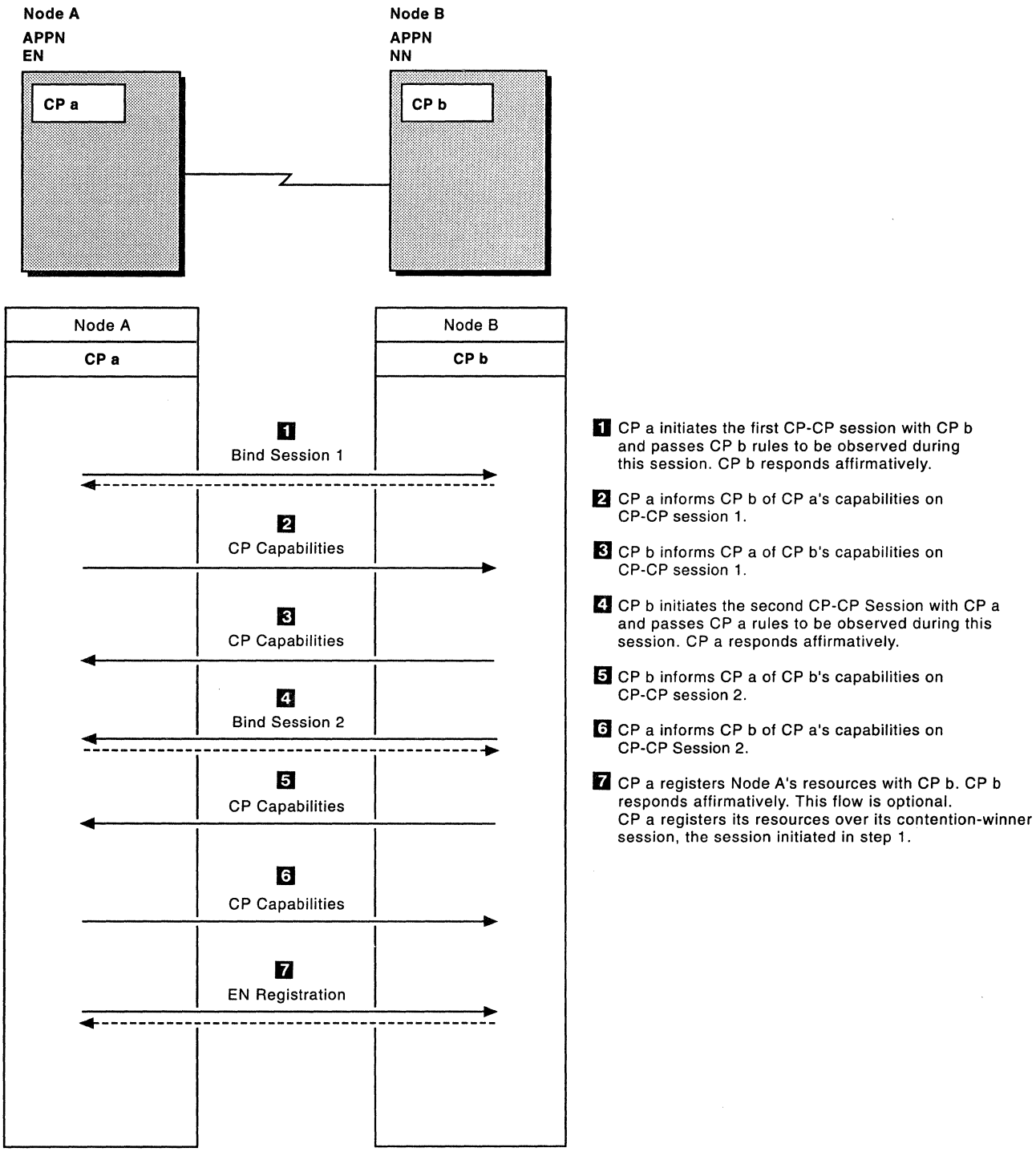


Figure 145. Activating an ENCP-NNCP Session and Registering the EN's Resources

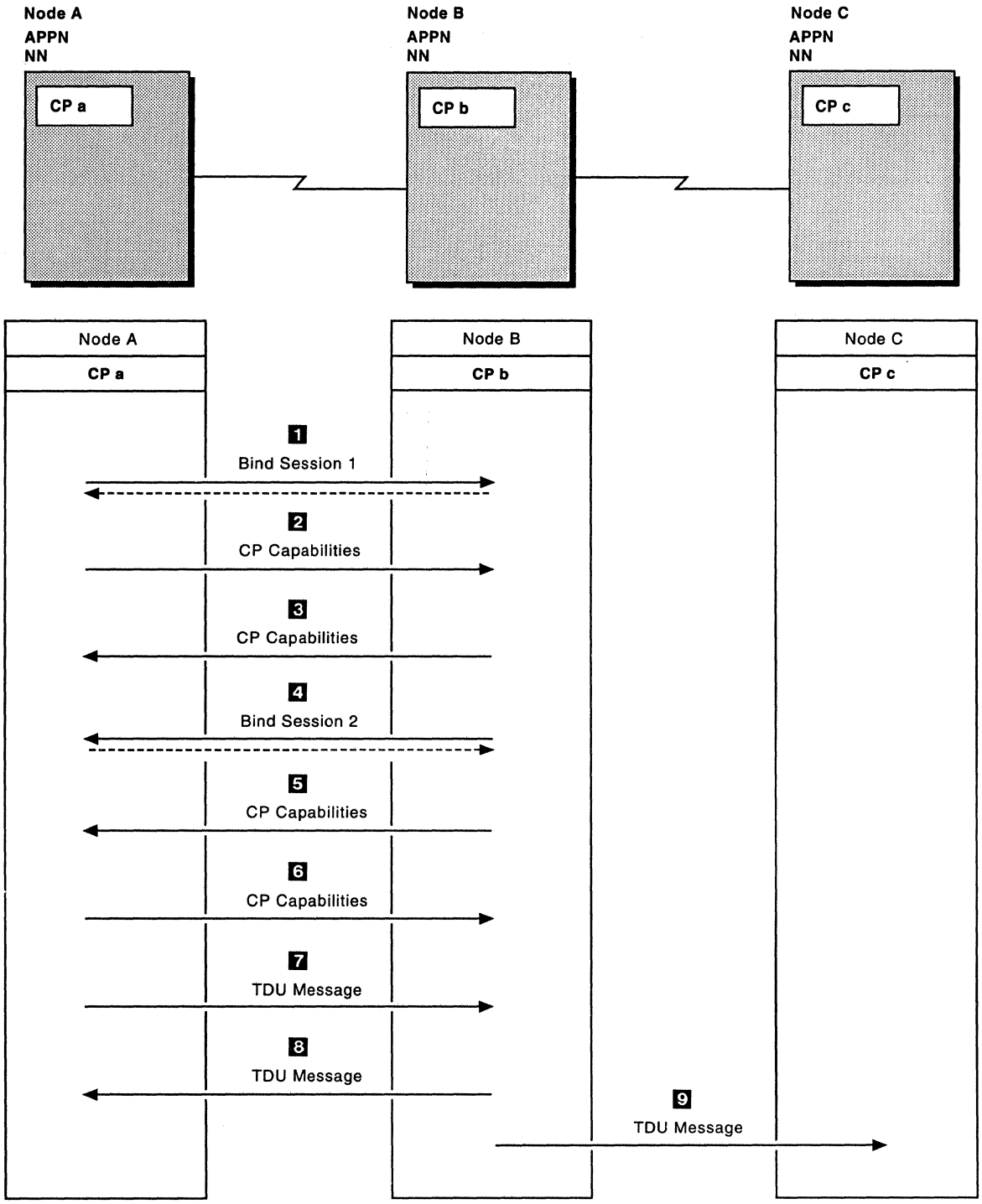


Figure 146 (Part 1 of 2). Activating an NNCP-NNCP Session and Exchanging Initial Topology

1. CP a activates the first CP-CP session with CP b and passes CP b rules to be observed during this session. CP b responds affirmatively.
2. CP a informs CP b of CP a's capabilities over CP-CP session 1.
3. CP b informs CP a of CP b's capabilities over CP-CP session 1.
4. CP b activates the second CP-CP session with CP a and passes CP a rules to be observed during this session. CP a responds affirmatively.
5. CP b informs CP a of CP b's capabilities over CP-CP session 2.
6. CP a informs CP b of CP a's capabilities over CP-CP session 2.
7. CP a begins the initial topology exchange by sending its view of the network topology to CP b. The information is sent in a TDU message over CP a's contention-winner session.
8. CP b completes the initial topology exchange by sending its view of the network topology to CP a. The information is sent in a TDU message over CP b's contention-winner session.
9. CP b sends a TDU broadcast message to CP c containing a resource update for each new resource of which CP b has been made aware by the initial topology exchange.

Figure 146 (Part 2 of 2). Activating an NNCP-NNCP Session and Exchanging Initial Topology

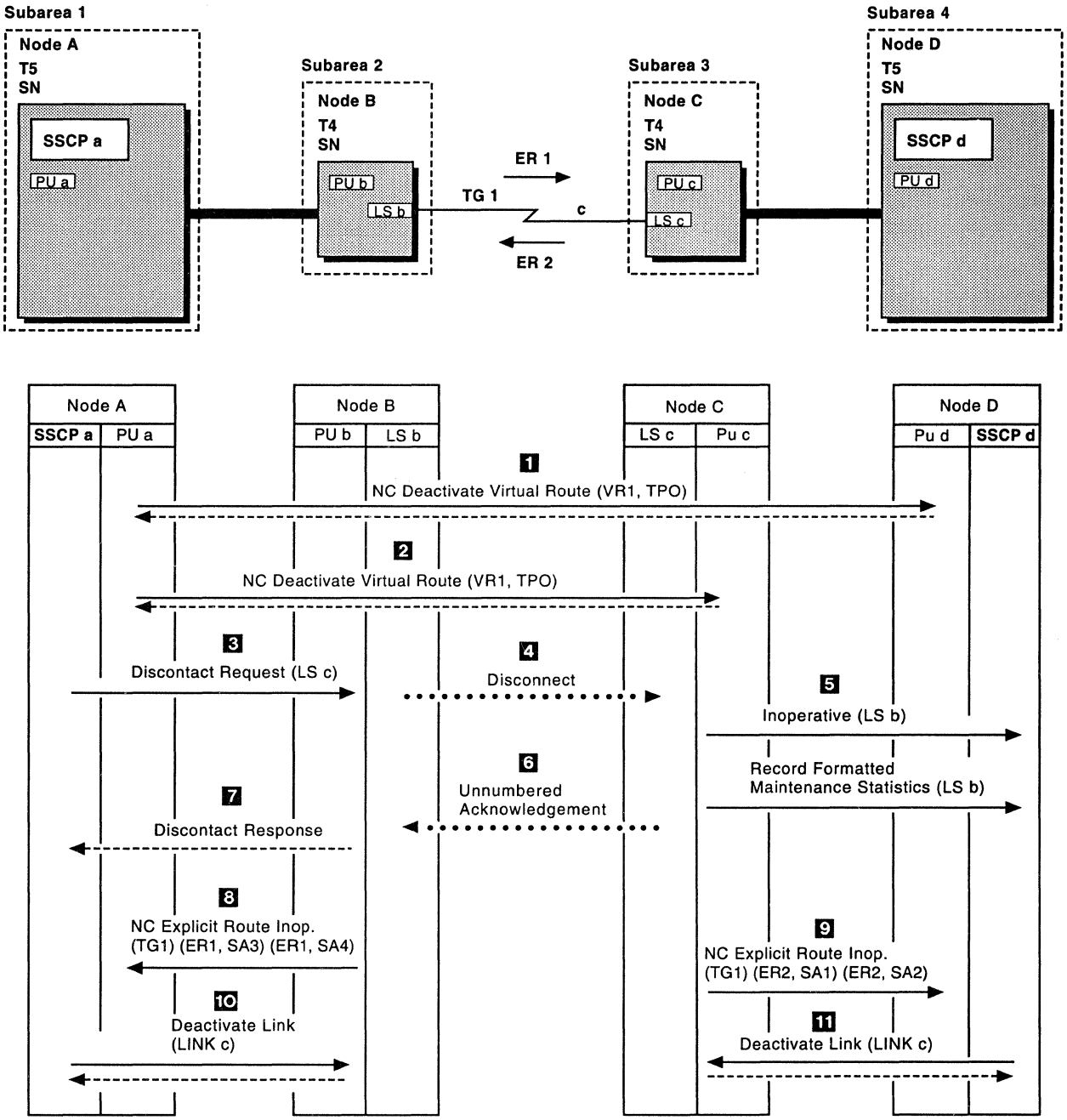


Figure 147 (Part 1 of 2). Deactivating Virtual Routes, Explicit Routes, and SDLC Links

1 PU a deactivates the virtual route having a virtual route number of 1 and a transmission priority of 0 between subarea 1 and subarea 4, because the last session assigned to this virtual route has been deactivated.

2 PU a deactivates the virtual route having a virtual route number of 1 and a transmission priority of 0 between subarea 1 and subarea 3, because the last session assigned to this virtual route has been deactivated.

3 SSCP a tells PU b to break contact with link station LS c. The representation of the LS in Node C has a network address of c.

4 PU b causes local link station LS b to initiate data link control procedures to break contact with link station LS c. LS b tells LS c to go into disconnect mode.

5 PU c informs SSCP d that link station LS b is inoperative, and sends SSCP d maintenance statistics relating to LS b. The representation of the LS in Node C has a network address of b.

6 LS c informs LS b that LS c has entered disconnect mode.

7 PU b sends a positive Discontact response to SSCP a.

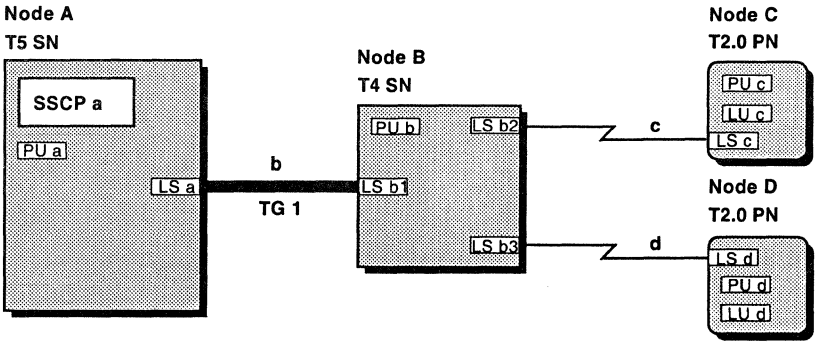
8 PU b informs PU a that TG 1 has had a routing interruption that rendered inoperative ER1 to subarea 3 and ER1 to subarea 4.

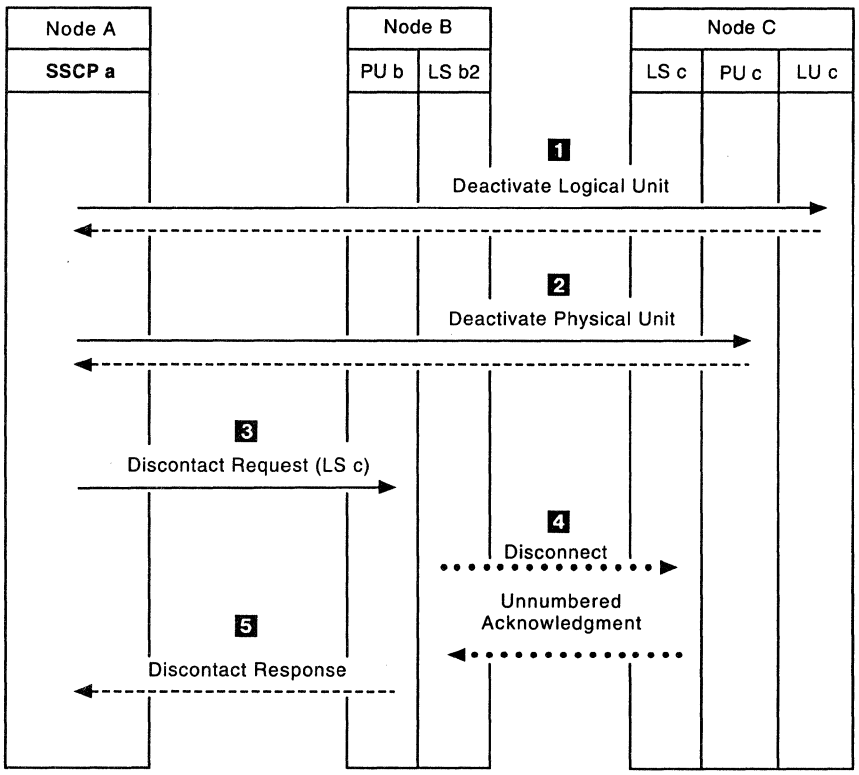
9 PU c informs PU d that TG 1 has had a routing interruption that rendered inoperative ER2 to subarea 1 and ER2 to subarea 2.

10 SSCP a tells PU b to deactivate Node B's end of link connection c.

11 SSCP d tells PU c to deactivate Node C's end of link connection c.

Figure 147 (Part 2 of 2). Deactivating Virtual Routes, Explicit Routes, and SDLC Links





1. SSCP a relinquishes control of LU c by deactivating the SSCP-LU session between itself and LU c.

2. SSCP a relinquishes control of Node C by deactivating the SSCP-PU session between itself and PU c.

3. SSCP a tells PU b to break contact with link station LS c.

4. PU b causes local link station LS b2 to initiate data link control procedures to break contact with link station LS c. LS b2 tells LS c to enter disconnect mode, and LS c responds affirmatively.

5. PU b sends SSCP a a positive response to the Discontact request.

Figure 148. Deactivating a Peripheral Node Attached by a Nonswitched SDLC Link

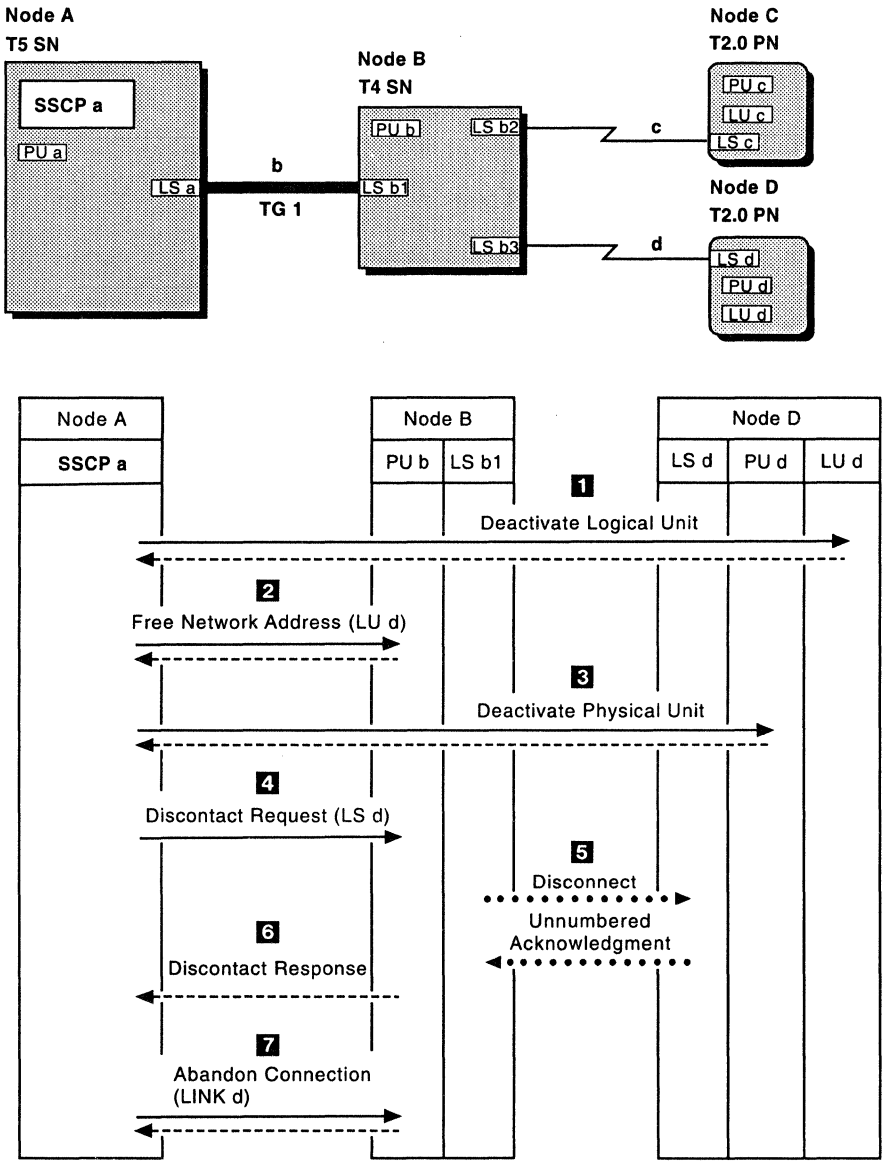


Figure 149 (Part 1 of 2). Deactivating a Peripheral Node Attached by a Switched SDLC Link

1. SSCP a relinquishes control of LU d by deactivating the SSCP-LU session between itself and LU d.

2. SSCP a tells PU b to disassociate LU d's network address from LU d. The freed network address is returned to a pool in Node B, from which it may be reassigned.

3. SSCP a relinquishes control of Node D by deactivating the SSCP-PU session between itself and PU d.

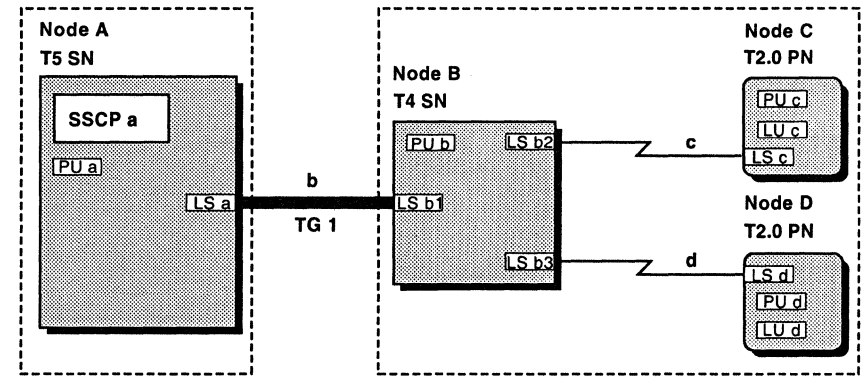
4. SSCP a tells PU b to break contact with link station LS d. The representation of LS d in Node B has a network address of d.

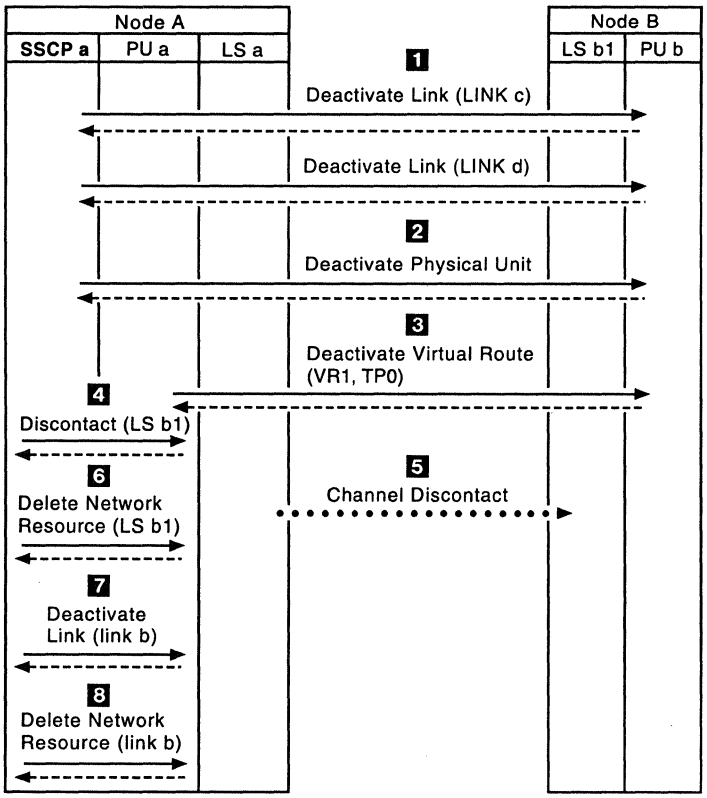
5. PU b causes local link station LS b to initiate data link control procedures to break contact with link station LS d. LS b tells LS d to enter disconnect mode, and LS d responds affirmatively.

6. PU b sends SSCP a a positive response to the Discontact request.

7. SSCP a tells PU b to break the switched connection between Node B and Node D (link d). PU b does so.

Figure 149 (Part 2 of 2). Deactivating a Peripheral Node Attached by a Switched SDLC Link





1. SSCP a causes PU b to deactivate all links to peripheral nodes in subarea 2.
2. SSCP a relinquishes control of Node B by deactivating the SSCP-PU session between itself and PU b.
3. PU a deactivates the virtual route having a virtual route number of 1 and a transmission priority of 0 between subarea 1 and subarea 2, because the last session assigned to this virtual route has been deactivated.
4. SSCP a tells PU a to break contact with link station LS b1.
5. PU a causes local link station LS a to initiate data link control procedures to break contact with link station LS b1. LS a disconnects channel b.
6. SSCP a tells PU a to break the association with the adjacent link station in Node B, thereby rendering its network address available for reasssignment.
7. SSCP a tells PU a to deactivate Link connection b.
8. SSCP a tells PU a to break the association between Link b and Link b's network address, thereby rendering the network address available for reassignment.

Figure 150. Deactivating a Channel-Attached Subarea Node and Associated Resources

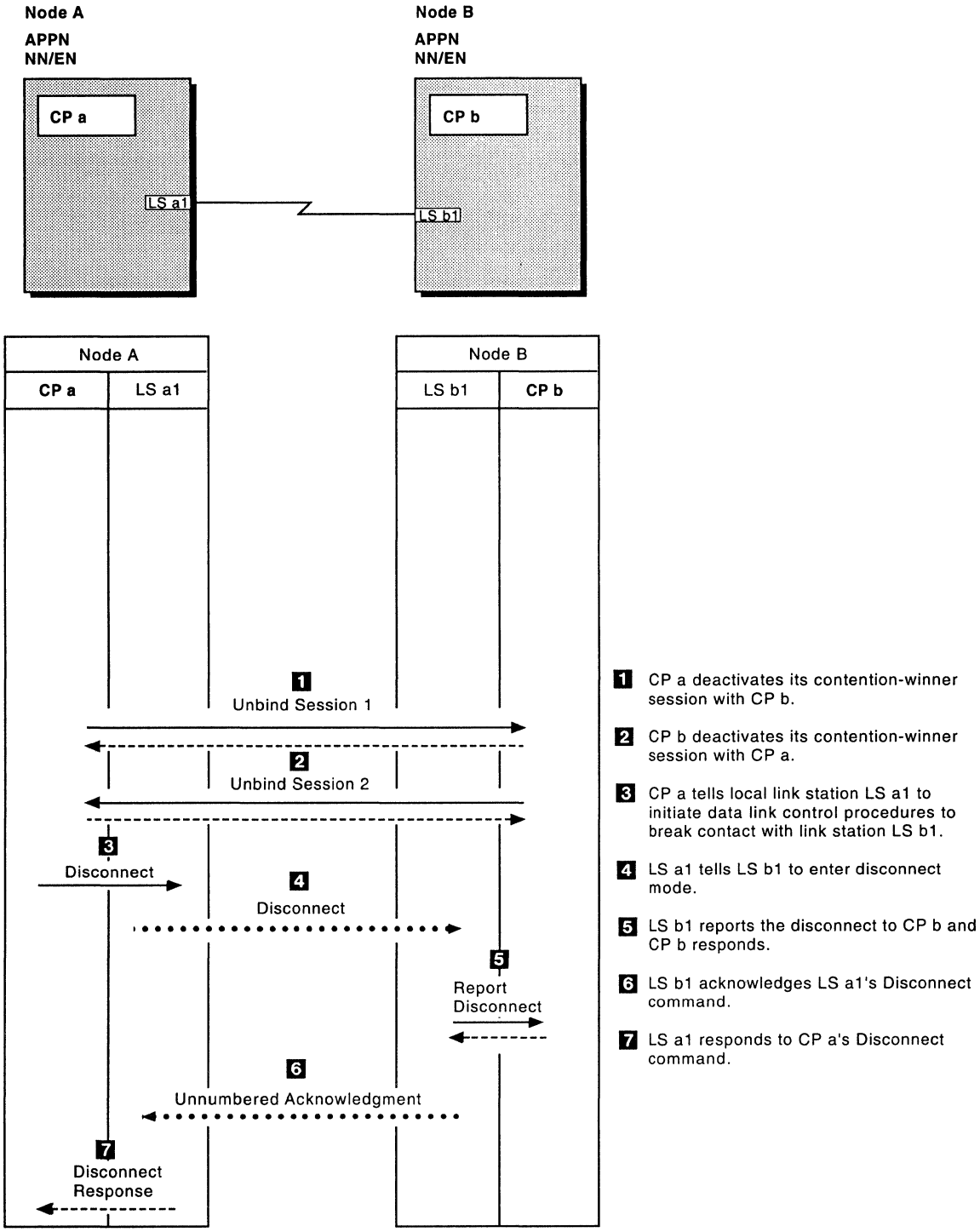


Figure 151. Deactivating a CP-CP Session and a Nonswitched SDLC Link Between APPN Nodes

Typical Request Unit Sequences for Routing

Figure 153 through Figure 155 present typical request unit sequences for routing data through the network. Figure 152 gives the meaning of each of the symbols and abbreviations that appear in these sequence charts. Chapter 6, "Establishing Routes Through the Network" explains how the path control network routes data between network accessible units.

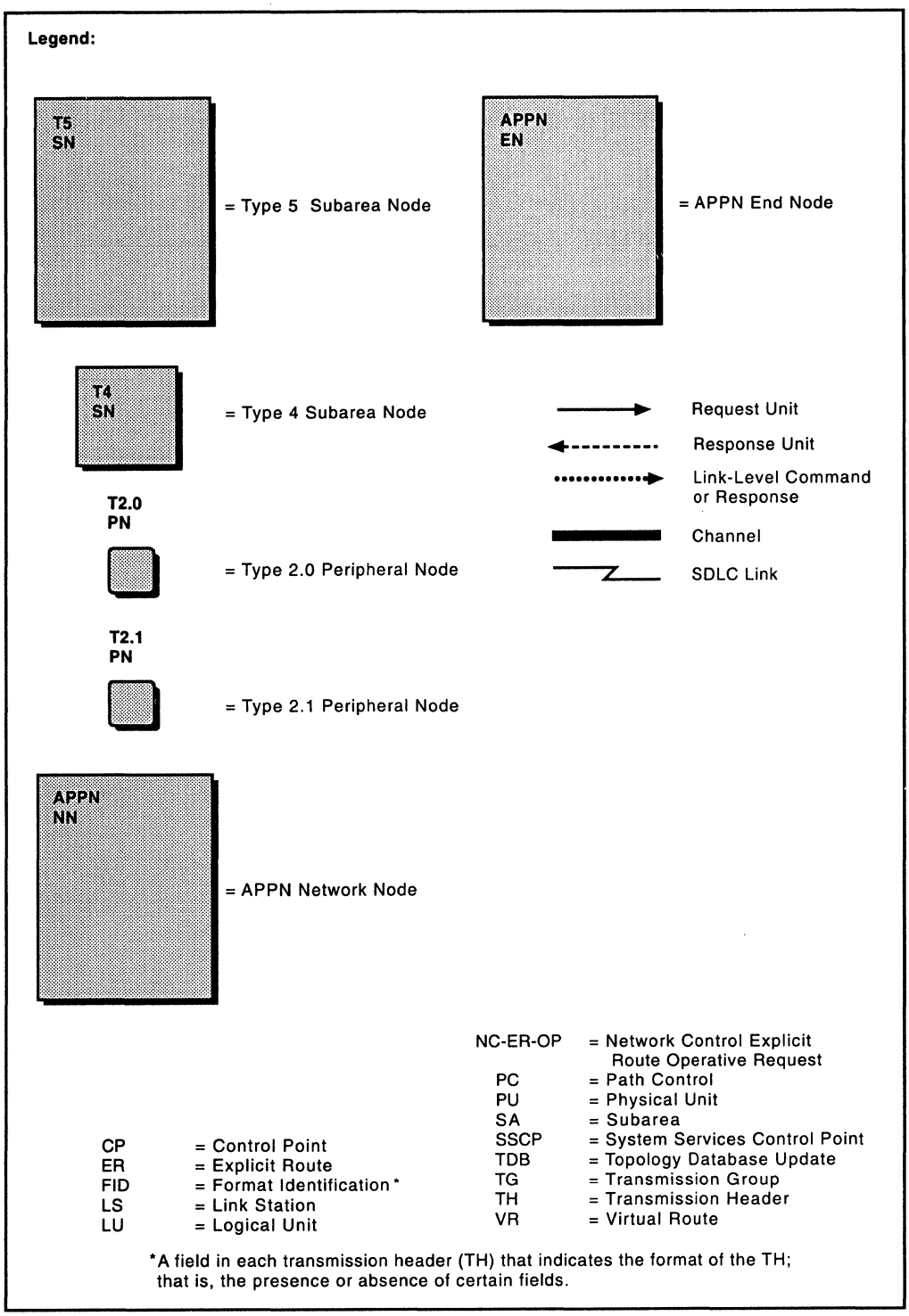


Figure 152. Symbols and Abbreviations for Figures 153 through 155

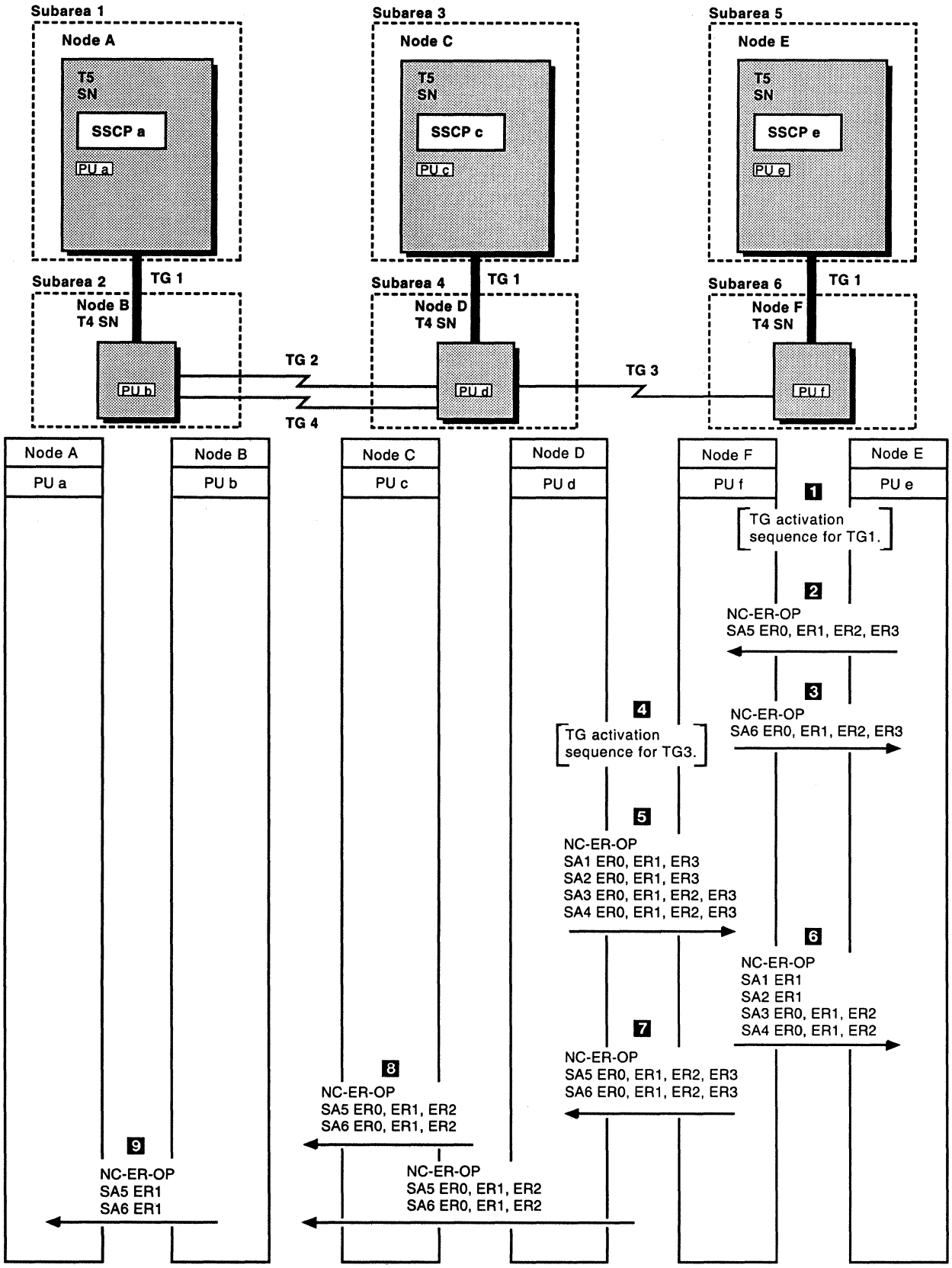


Figure 153 (Part 1 of 2). Propagation of Explicit Route Operative (NC-ER-OP) Requests

This sequence of request units assumes that all transmission groups are active except for TG 1 between Node E and Node F, and for TG 3, which are inactive.

1. Transmission group TG 1 is activated.

2. PU e tells PU f that PU e can handle data routed to subarea 5 over explicit routes 0, 1, 2, and 3, once these routes are activated. PU f uses this information in deciding which explicit routes to activate.

3. PU f tells PU e that PU f can handle data routed to subarea 6 over explicit routes 0, 1, 2, and 3.

4. Transmission group TG 3 is activated.

5. PU d tells PU f that PU d can handle data routed to the following subareas over the following explicit routes once these routes are activated:
   - Subareas 1 and 2 via explicit routes 0, 1, and 3
   - Subareas 3 and 4 via explicit routes 0, 1, 2, and 3

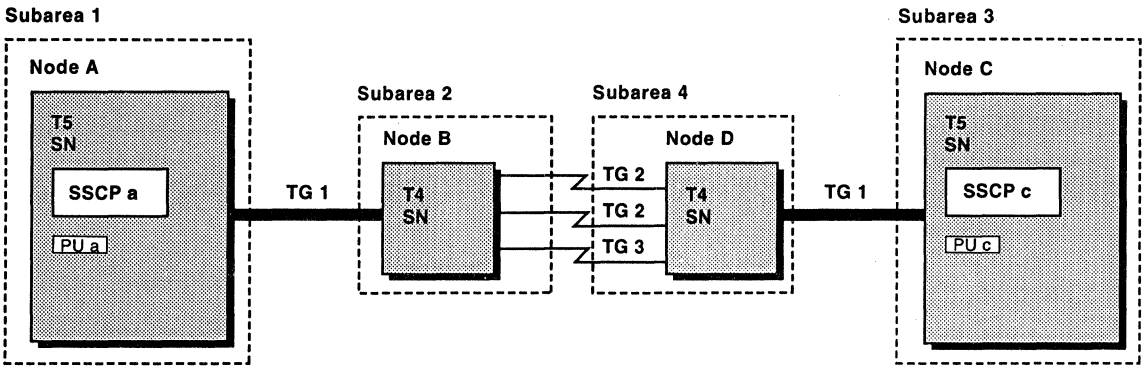
6. PU f tells PU e that PU f can handle data routed to the following subareas over the following explicit routes:
   - Subareas 1 and 2 via explicit route 1
   - Subareas 3 and 4 via explicit routes 0, 1, and 2

7. PU f tells PU d that PU f can handle data routed to subareas 5 and 6 over explicit routes 0, 1, 2, and 3.

8. After modifying the information from PU f to reflect the information in its own routing table, PU d propagates this information to the PUs in the other subarea nodes connected to it - that is, PU b in Node B and PU c in Node C.

9. PU b modifies the information it received from PU d and propagates it to PU a.

Figure 153 (Part 2 of 2). Propagation of Explicit Route Operative (NC-ER-OP) Requests



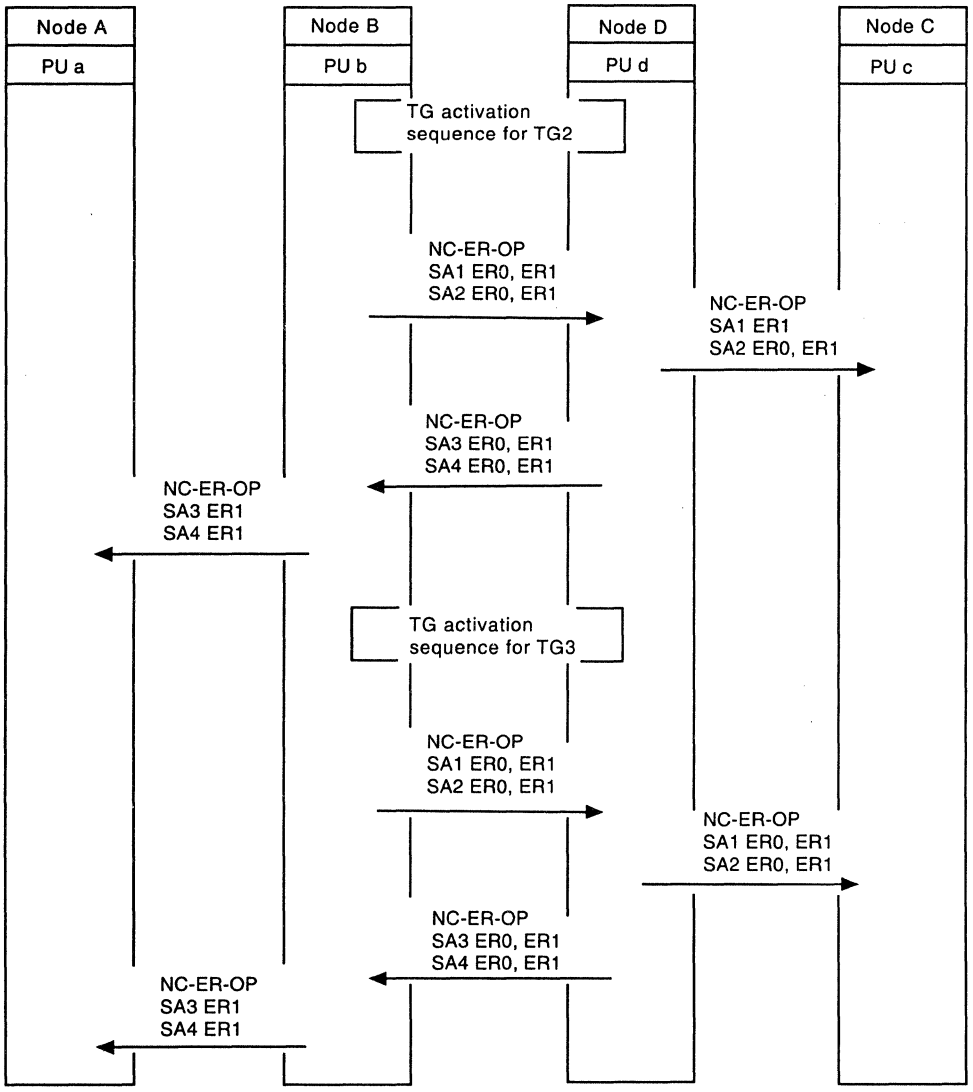
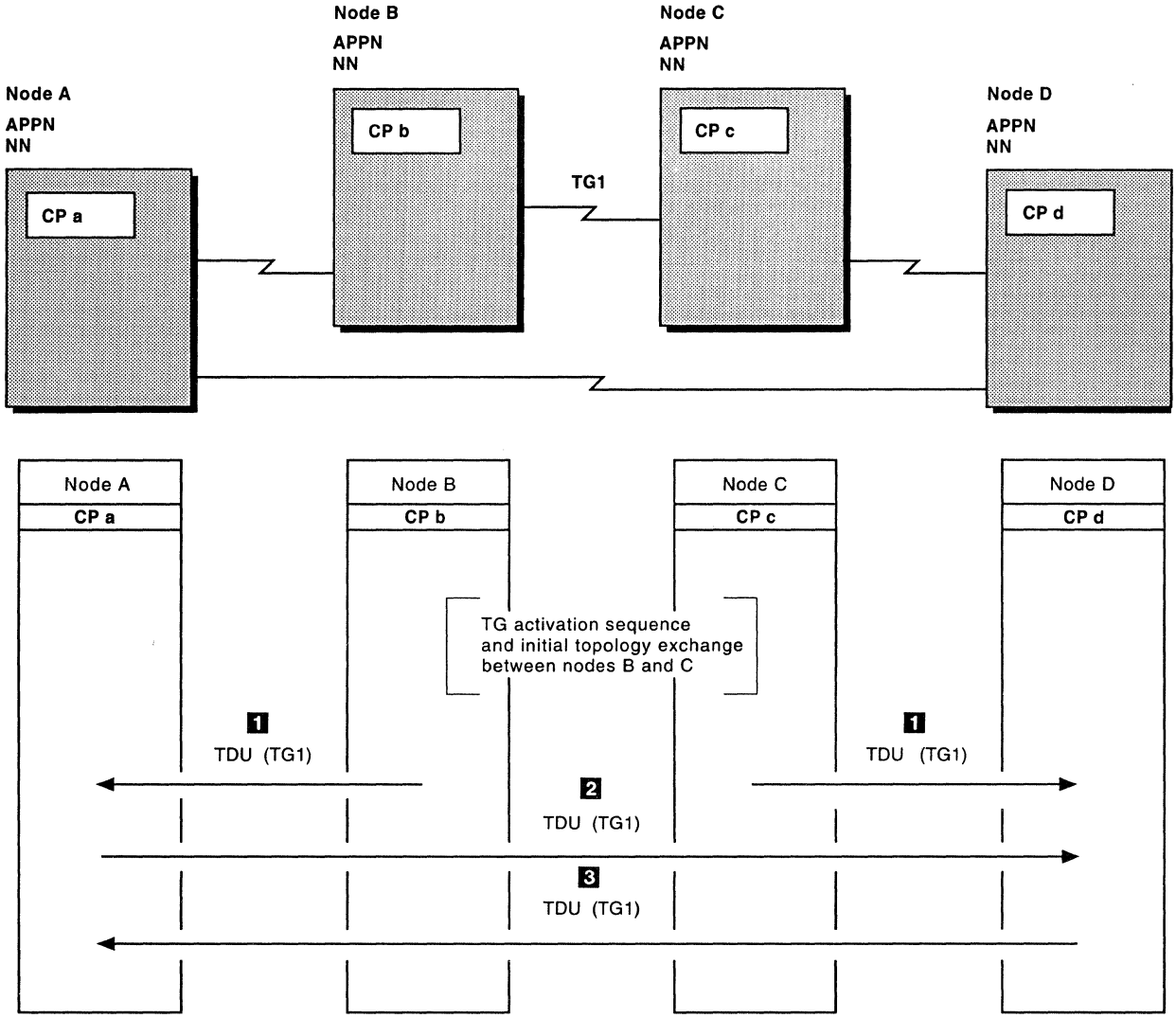


Figure 154 (Part 1 of 2). Propagation of Routing Information Following Activation of Multiple Transmission Groups between the Same Subareas

This sequence of request units assumes that transmission groups TG 2 and TG 3 are inactive and that both TG 1's are active.

1. TG 2 is activated.
2. PU b tells PU d that PU b can reach subareas 1 and 2 over explicit routes 0 and 1.
3. PU d tells PU c that PU d can reach subarea 1 over explicit route 1 and subarea 2 over explicit routes 0 and 1. PU d uses TG 3 to reach subarea 1 over explicit route 0; since TG 3 has not yet been activated, explicit route 0 is not yet available.
4. PU d sends the appropriate information on connectivity to PU b, which modifies the information to reflect the contents of its own routing tables and then propagates the information to PU a.
5. TG 3 is activated.
6. PU b sends PU d the same information that it sent in Step 2.
7. PU d in turn propagates this information to PU c. Because TG 3 is now active, PU d tells PU c that PU d can now reach subarea 1 over explicit route 0, as well as over explicit route 1.
8. PU d sends PU b the same information that it sent in Step 4. PU b propagates this information to PU a.

Figure 154 (Part 2 of 2). Propagation of Routing Information Following Activation of Multiple Transmission Groups between the Same Subareas



1 Following initial topology exchange, CPs b and c propagate changes to the network topology in TDU messages. CP b sends a TDU message regarding the active status of TG1 to CP a and CP a updates its topology database; CP c sends a similar TDU message to CP d and CP d updates its topology database.

2 CP a sends the TDU message it received from CP b on to CP d. CP d recognizes the TDU message as a duplicate of the one it received from CP c and discards it.

3 CP d sends the TDU message it received from CP c on to CP a. CP a recognizes the TDU message as a duplicate of the one it received from CP b and discards it.

Figure 155. Propagation of Topology Database Update Messages Following Activation of a Transmission Group between Two Network Nodes

Typical Request Unit Sequences for Initiating Sessions, Terminating Sessions, and Transferring Data

Figure 157 through Figure 175 present typical request unit sequences for initiating and terminating sessions between network accessible units and for transferring data over a session. Figure 156 gives the meaning of each of the symbols and abbreviations that appear in these sequence charts.

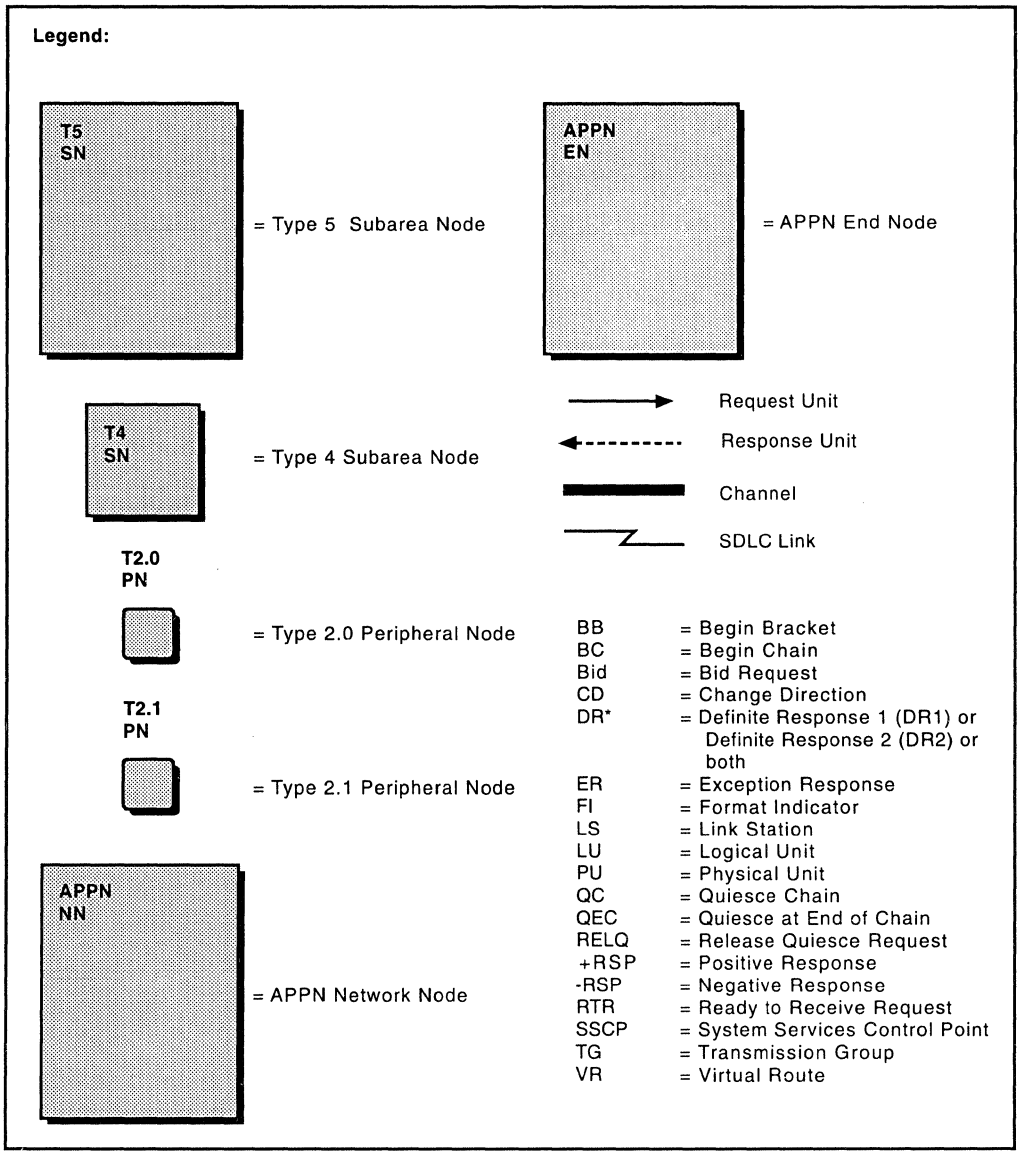


Figure 156. Symbols and Abbreviations for Figures 157 through 175

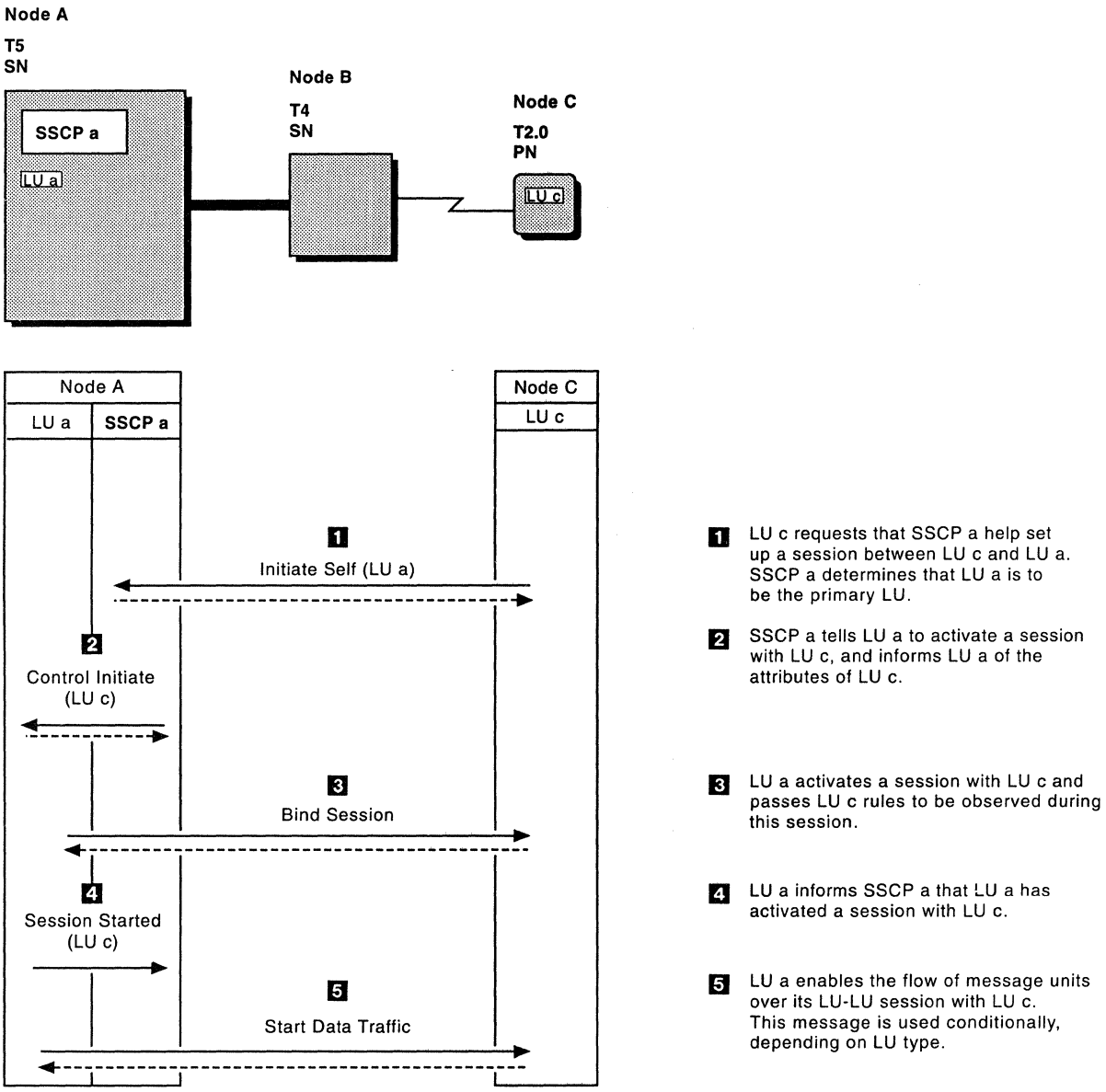


Figure 157. Initiating a Same-Domain LU-LU Session in a Subarea Network

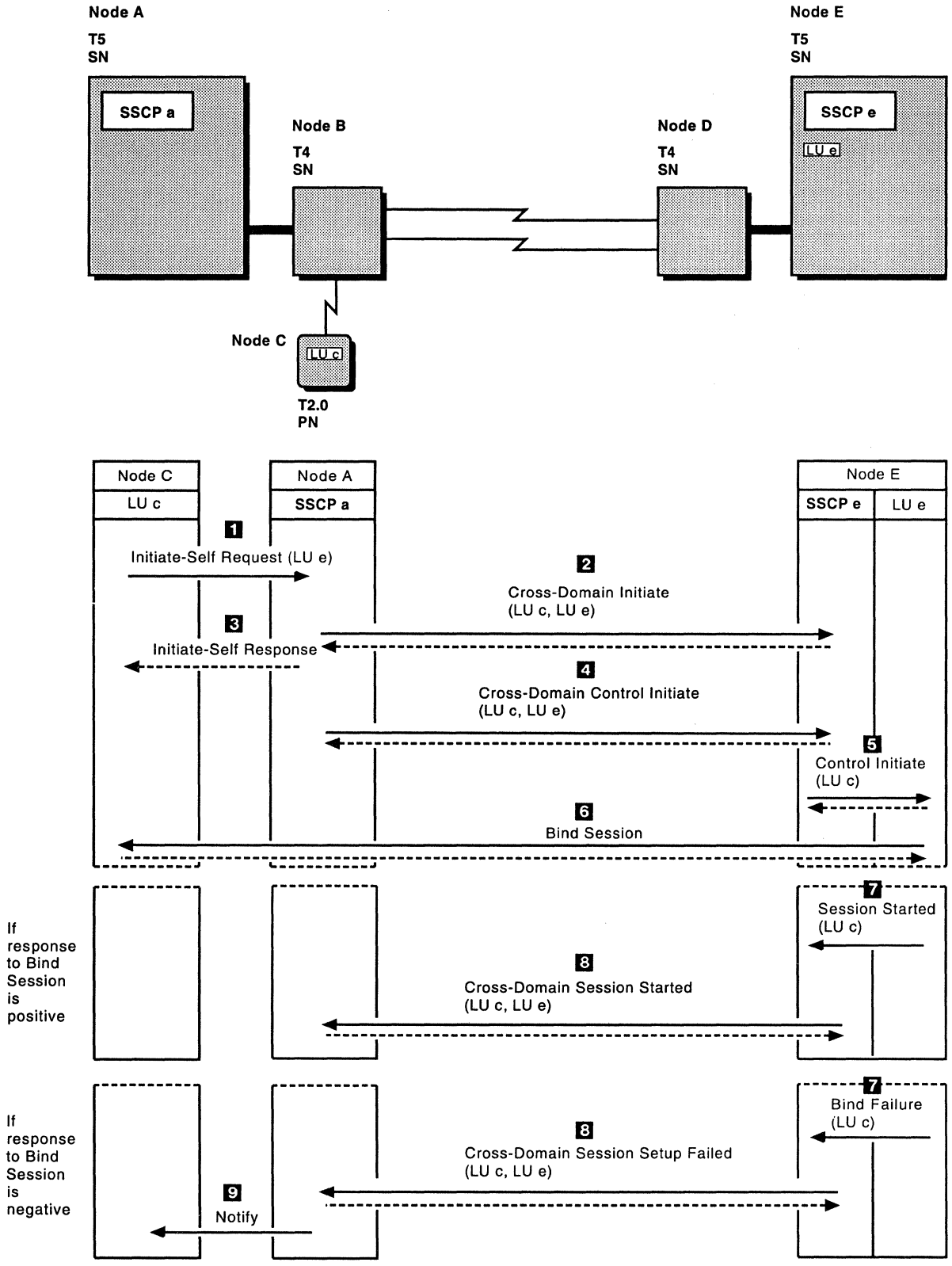


Figure 158 (Part 1 of 2). Initiating a Cross-Domain LU-LU Session in a Subarea Network

1 LU c requests that SSCP a help set up a session between LU c and LU e. SSCP a determines that LU e is in the domain of SSCP e and that LU e is to be the primary LU.

2 SSCP a tells SSCP e of LU c's request. After receiving a response from SSCP e, SSCP a returns to LU c a response to LU c's request.

3 SSCP a sends LU c a positive response to the Initiate-Self request.

4 SSCP a sends SSCP e the session rules that must be in effect for the session between LU c and LU e.

5 SSCP e tells LU e to activate a session with LU c using the session rules passed by SSCP a in step 3.

6 LU e activates a session with LU c using the session rules given by SSCP e in step 4.

If a session is activated successfully:

7 LU e informs SSCP e that LU e has activated a session with LU c.

8 SSCP e informs SSCP a of this fact.

If the session is not activated successfully:

7 LU e informs SSCP e that LU e has failed in its attempt to activate a session with LU c.

8 SSCP e informs SSCP a of this fact.

9 SSCP a tells LU c that LU e believes that LU e's attempt to deactivate the session has failed.

Figure 158 (Part 2 of 2). Initiating a Cross-Domain LU-LU Session in a Subarea Network

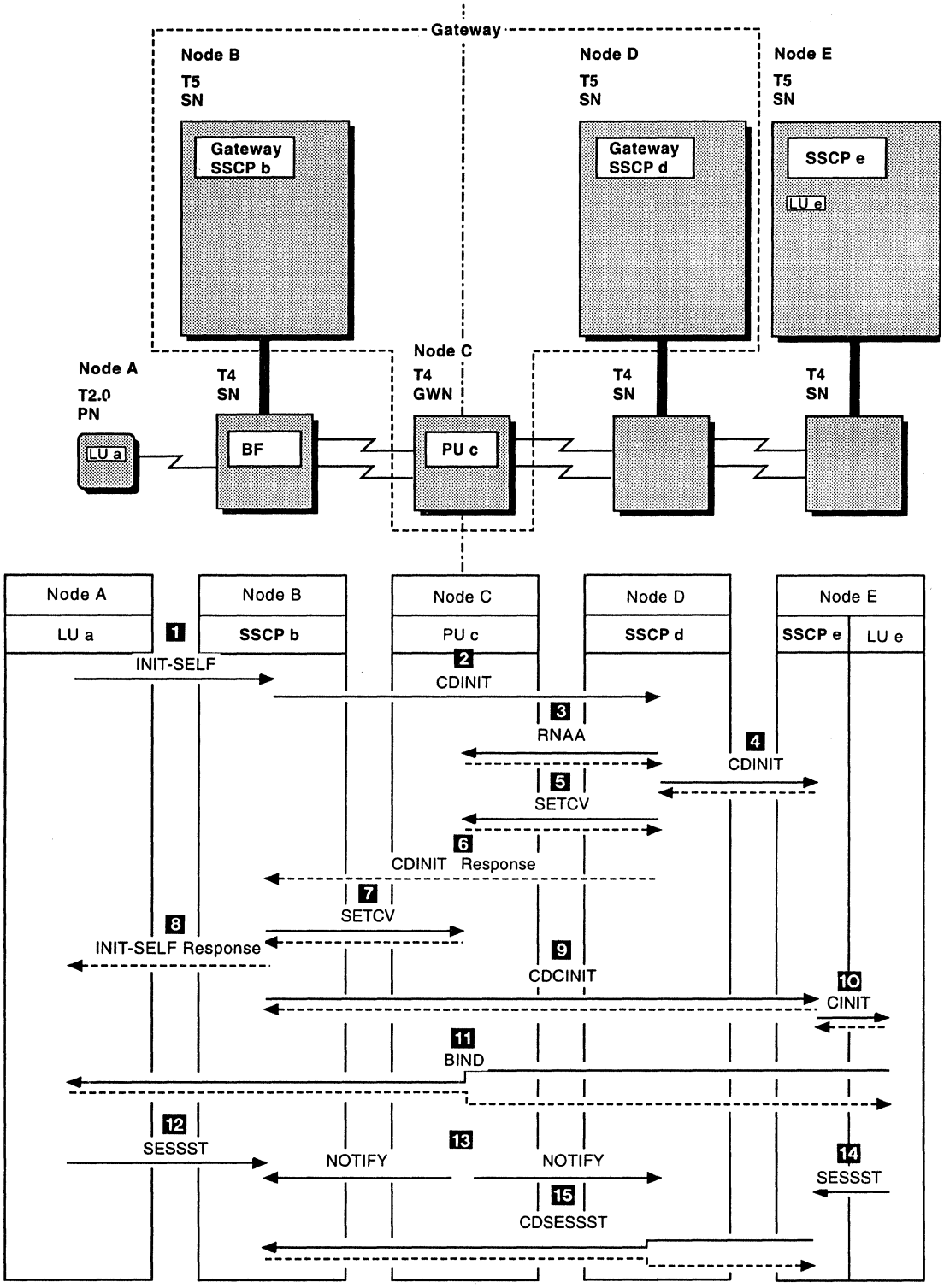


Figure 159 (Part 1 of 2). Initiating a Cross-Network LU-LU Session in a Subarea Network

1. LU a sends the SSCP in its domain, SSCP b, an Initiate-Self (INIT-SELF) request asking for a session with LU e. LU a is unaware that its SSCP also functions as a gateway SSCP.

2. SSCP a sends a Cross-Domain Initiate (CDINIT) request to the SSCP of the destination LU (DLU), SSCP e. The other gateway SSCP, SSCP d, must intercept the CDINIT, however, in order to obtain an alias address for LU a from the gateway node. Between gateway SSCPs, the CDINIT request carries control vectors containing information necessary for setting up the session. Such information includes the alias name of the DLU, the real address of the origin LU (OLU), the class-of-service (COS) and mode names, and the gateway node name.

3. Gateway SSCP d sends a Request Network Address Assignment (RNAA) request containing LU a's real network address to PU c requesting that PU c assign alias addresses for the OLU and DLU. (During the LU-LU session, each LU will use the alias addresses in message units destined for the other.) PU c returns the alias addresses to SSCP d in the RNAA response.

4. SSCP d reroutes SSCP b's CDINIT to SSCP e. The CDINIT now contains an alias address for LU a provided by PU c in step 3. The CDINIT response contains the real address of the DLU, LU e, in Network B.

5. SSCP d sends PU c a Set Control Vector (SETCV) request to inform PU c of the real address of LU e in Network B.

6. SSCP d reroutes SSCP e's CDINIT response back to SSCP b.

7. SSCP b resolves the COS name for LU a to a VR ID list and sends the list and the LU name transform to PU c in a SETCV request.

8. SSCP b sends LU a a positive response to its INIT-SELF request.

9. The SSCP of the secondary LU (SLU), SSCP b, sends a Cross Domain Control Initiate (CDCINIT) request containing parameters for the Bind image to the SSCP of the primary LU (PLU), SSCP e.

10. The SSCP of the PLU, SSCP e, sends a Control Initiate (CINIT) request that contains the Bind image to the PLU, LU e.

11. LU e sends LU a a Bind Session (BIND) request to activate the LU-LU session. LU a responds positively.

12. LU a sends SSCP b a Session Started (SESSST) request to notify it of the successful LU-LU session activation. (When the LU resides in a peripheral node, as in this case, the boundary function for the PN actually sends the SESSST.)

13. The gateway node notifies all gateway SSCPs in its gateway, SSCPs b and d, of the LU-LU session activation.

14. LU e sends SSCP e a Session Started (SESSST) request to notify it of the successful LU-LU session activation.

15. The SSCP of the PLU, SSCP e, sends a Cross-Domain Session Started (CDSESSST) request to the SSCP of the SLU, SSCP b, so that both SSCPs can update their session awareness records.

Figure 159 (Part 2 of 2). Initiating a Cross-Network LU-LU Session in a Subarea Network

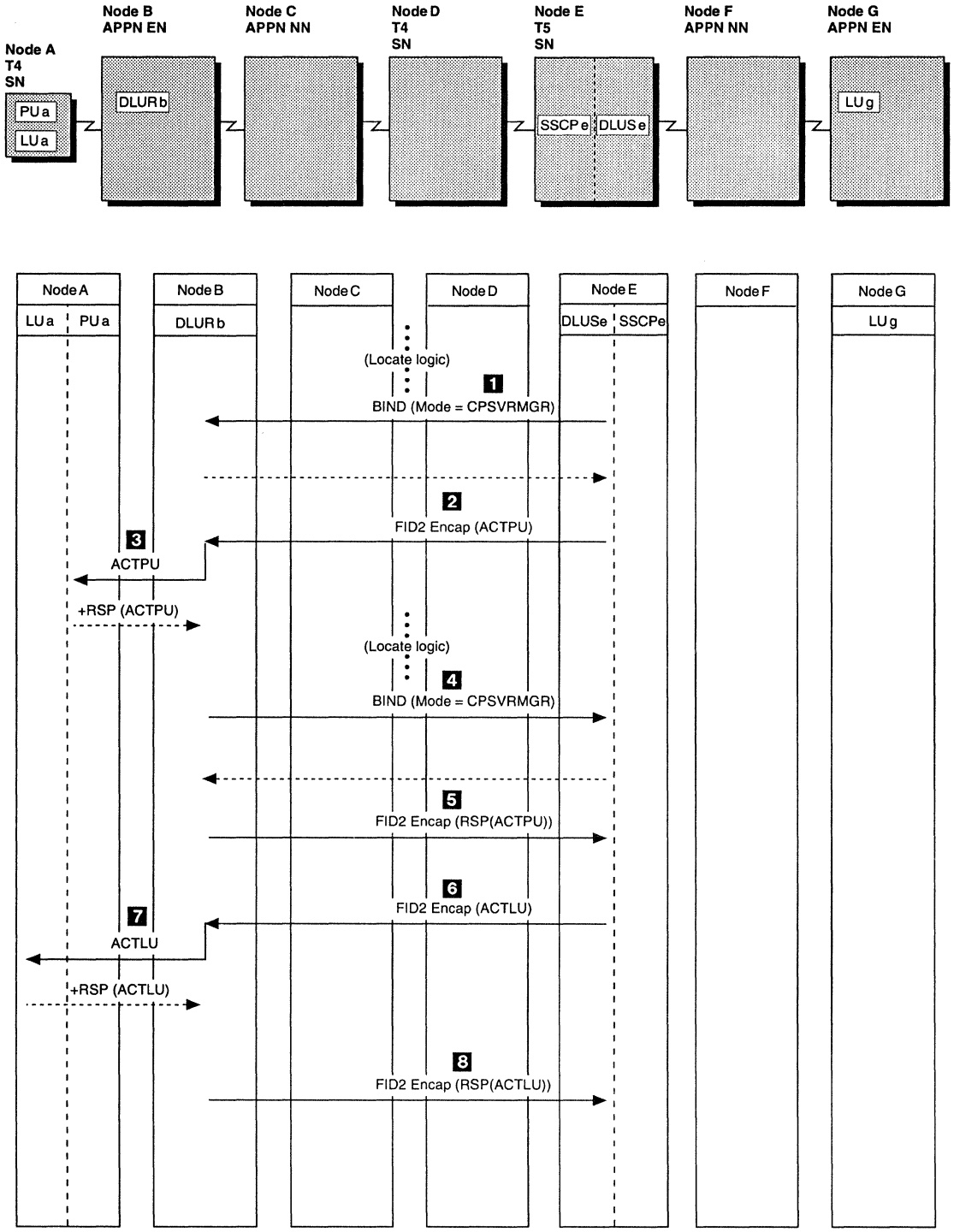


Figure 160 (Part 1 of 2). DLUS-Initiated CP-SVR Pipe Activation

1. DLUS e (node E) sends a BIND using the CPSVRMGR mode to DLUR b (node B) to activate the DLUS-DLUR CPSVRMGR session.
2. DLUS e uses the FID2 Encapsulation GDS variable to send an encapsulated FID2 ACTPU to DLUR b to activate the SSCP-PU session from the DLUS to the DLUR.
3. When DLUR b receives the encapsulated message, it send the ACTPU to Node A. Node A responds to the ACTPU by sending an ACTPU response to DLUR b.
4. DLUR b sends a BIND using the CPSVRMGR mode to DLUS e to activate the DLUS-DLUR CPSVRMGR session. This session and the session activated in step 1 comprise the ***CP-SVR pipe***.
5. DLUR b uses the FID2 Encapsulation GDS variable to send an encapsulated FID2 RSP(ACTPU) to DLUS e to activate the SSCP-PU session from the DLUR to the DLUS.
6. DLUS e sends an encapsulated FID2 ACTLU over the DLUS-DLUR CPSVRMGR session to DLUR b to activate the LU-LU session.
7. DLUR b strips off the FID2 Encapsulation GDS variable and sends the ACTLU to Node A. Node A sends a RSP(ACTLU) to DLUR b.
8. DLUR b sends an encapsulated FID2 ACTLU over the CP-SVR pipe to DLUS e.

Figure 160 (Part 2 of 2). DLUS-Initiated CP-SVR Pipe Activation

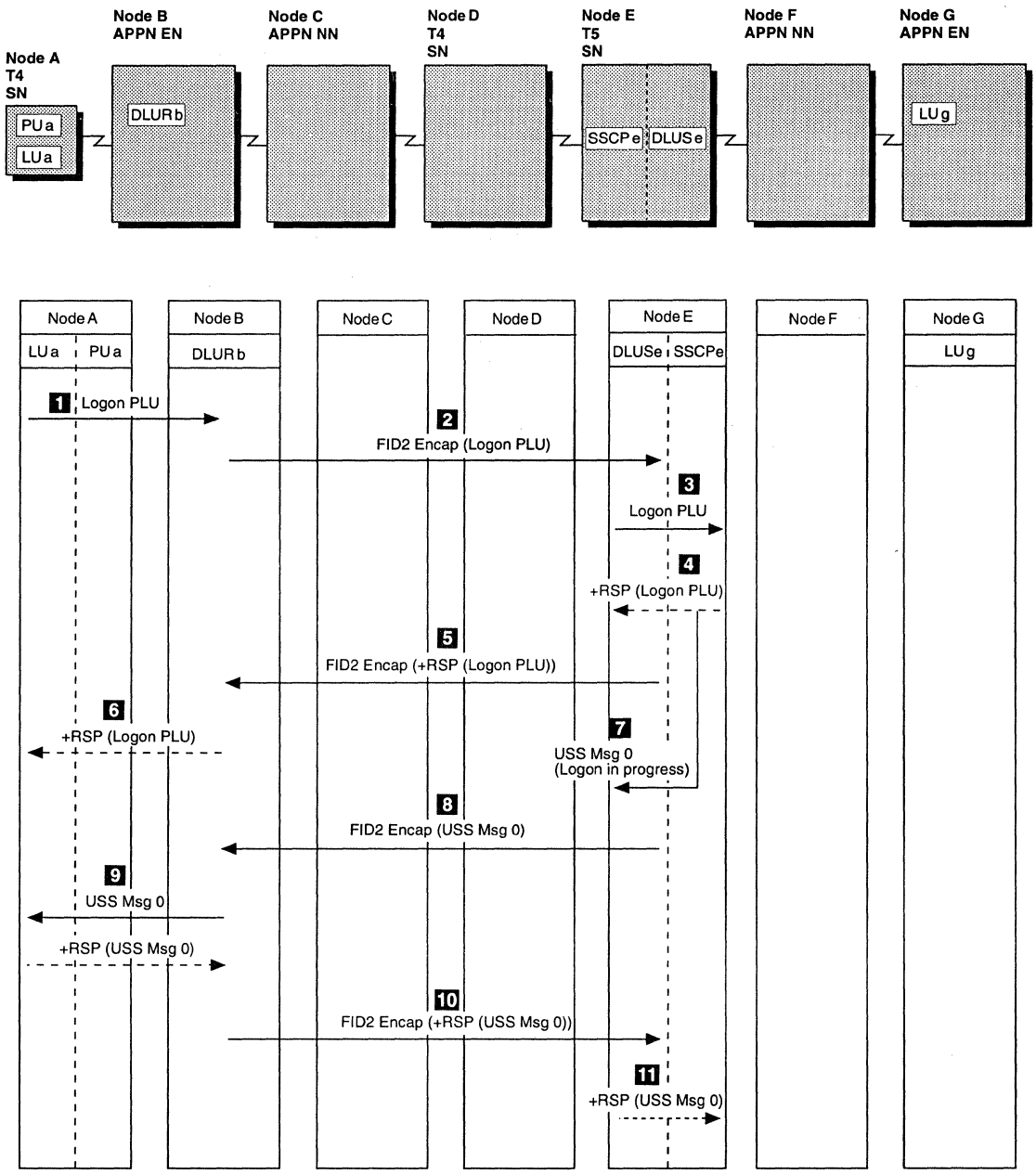


Figure 161 (Part 1 of 2). Unformatted Session Services (USS) SLU-Initiated LU-LU Session Using DLUR/S

1. The Logon PLU command is sent from LU a to DLUR b.
2. DLUR b uses the FID2 Encapsulation GDS variable to send an encapsulated FID2 Logon PLU command to DLUS e via the CP-SVR pipe.
3. DLUS e strips off the FID2 Encapsulation GDS variable and sends the Logon PLU command to SSCP e.
4. SSCP e receives the Logon command and issue a positive response to LU a via DLUS e.
5. Upon receipt of the RSP(Logon) (on its way to LU g), DLUS e uses the FID2 Encapsulation GDS variable to send an encapsulated FID2 Logon PLU command to DLUR b via the CP-SVR pipe.
6. DLUR b strips off the FID2 Encapsulation GDS variable and sends the +RSP(Logon PLU) to LU a.
7. SSCP e sends a USS (unformatted session services) Msg 0(logon in progress) to DLUS e.
8. DLUS e uses the FID2 Encapsulation GDS variable to send an encapsulated FID2 USS Msg 0 to LU a via DLUR b.
9. DLUR b strips off the FID2 Encapsulation GDS variable and sends the USS Msg 0 to LU a. LU a sends a +RSP(USS Msg 0) to DLUR b.
10. DLUR b uses the FID2 Encapsulation GDS variable to send an encapsulated FID2 +RSP(USS Msg 0) DLUS e via the CP-SVR pipe.
11. DLUS e strips off the FID2 Encapsulation GDS variable and sends the +RSP(USS Msg 0) to SSCP e.

Figure 161 (Part 2 of 2). Unformatted Session Services (USS) SLU-Initiated LU-LU Session Using DLUR/S

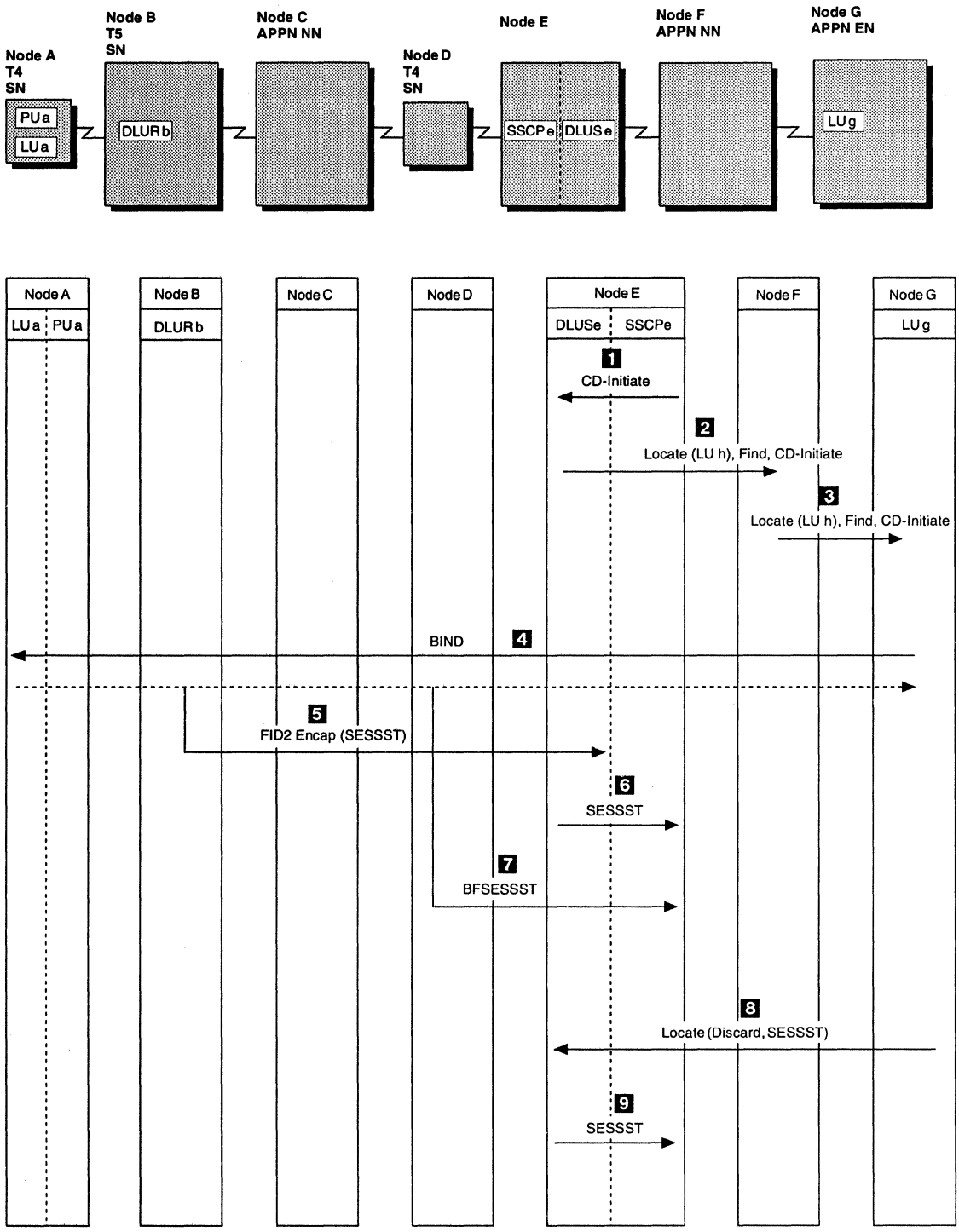
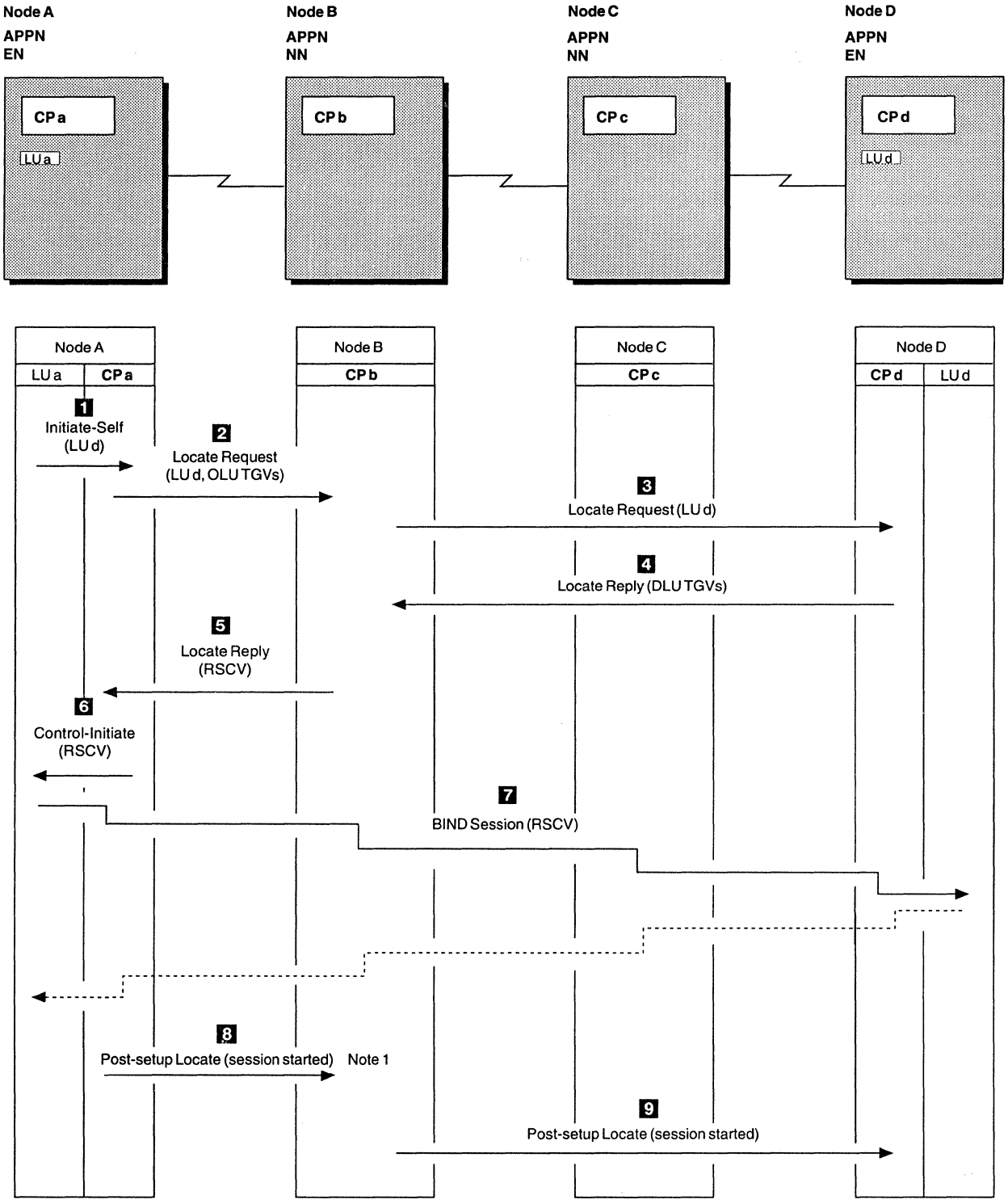


Figure 162 (Part 1 of 2). SLU-Initiated LU-LU Session Activation Using DLUR/S

1. The SSCP e portion of Node E sends a session initiation request to DLUS e portion of Node E. This indicates that the APPN portion of the network is to be searched for LU g.
2. DLUS e sends a Locate request to Node F to locate LU g.
3. Node F forwards the Locate request to Node G.
4. LU g sends a BIND to LU a in Node A.
5. Upon receipt of the RSP(BIND) (on its way to LU g), DLUR b uses the FID2 Encapsulation GDS variable to send an encapsulated FID2 SESSST to DLUS e in Node E to notify DLUS e that the LU-LU session has been established.
6. DLUS e strips off the FID2 Encapsulation GDS variable and sends the SESSST to SSCP e in Node E.
7. Upon receipt of the RSP(BIND), Node D creates a BFSESSST RU and forwards it to SSCP e in Node E.
8. Node G forwards a Locate (Discard,SESSST) to DLUS e in Node E.
9. DLUS e in Node E forwards a SESSST RU to SSCP e in Node E.

Figure 162 (Part 2 of 2). SLU-Initiated LU-LU Session Activation Using DLUR/S



Note: 1. A Post-setup Locate is the type of Locate that is sent after the Locate chain has been set up.

Figure 163 (Part 1 of 2). Initiating an LU-LU Session in an APPN Network

1. LU a sends CP a an Initiate-Self request to initiate a session with LU d.

2. CP a does not find LU d in its local directory and initiates a one-hop search by sending a Locate Search request to the CP in its network node server, CP b. The request includes the transmission group (TG) vectors of the origin LU (OLU), which will be used by CP b in calculating a route for the session.

3. CP b finds LU d in its local directory and initiates a directed search by sending a Locate Search request to CP d.

4. CP d sends a Locate Search reply back to a CP b verifying that LU d is in CP d's domain. The message includes the destination LU's (DLU's) TG vectors. The DLU's TG vectors, together with information from the topology database and the OLU's TG vectors enable CP b to calculate a route for the session.

5. CP b sends a Locate reply to CP a containing the route selection control vector (RSCV) for the route.

6. CP a sends a Control-Initiate request containing the RSCV to LU a.

7. LU a builds a Bind-Session request from the Control-Initiate request. containing the RSCV, and sends it to LU d; LU d responds positively with the Bind-Session response. In processing the Bind-Session and its response, LUs a and d each build and initialize half-sessions for the session and Nodes B and C each build session connectors. The address space space managers in all CPs route the BIND request according to the TG vectors specified in the RSCV.

8. CP b sends a post-setup Locate to CP a to notify CP a that the session between LU d and LU a is established.

9. CP b sends a post-setup Locate to CP d to notify CP d that the session between LU d and LU a is established.

Figure 163 (Part 2 of 2). Initiating an LU-LU Session in an APPN Network

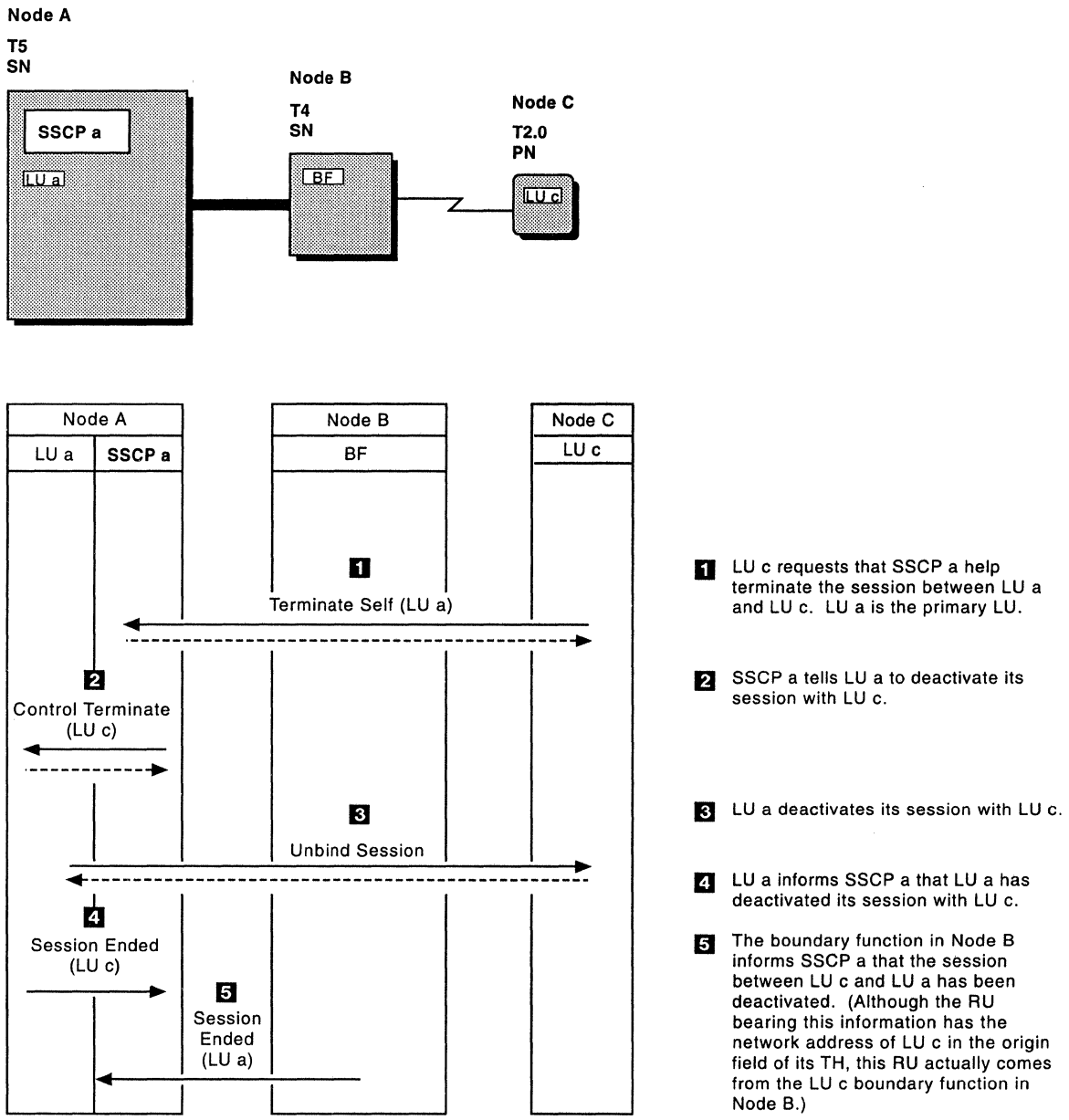


Figure 164. Terminating a Same-Domain LU-LU Session in a Subarea Network

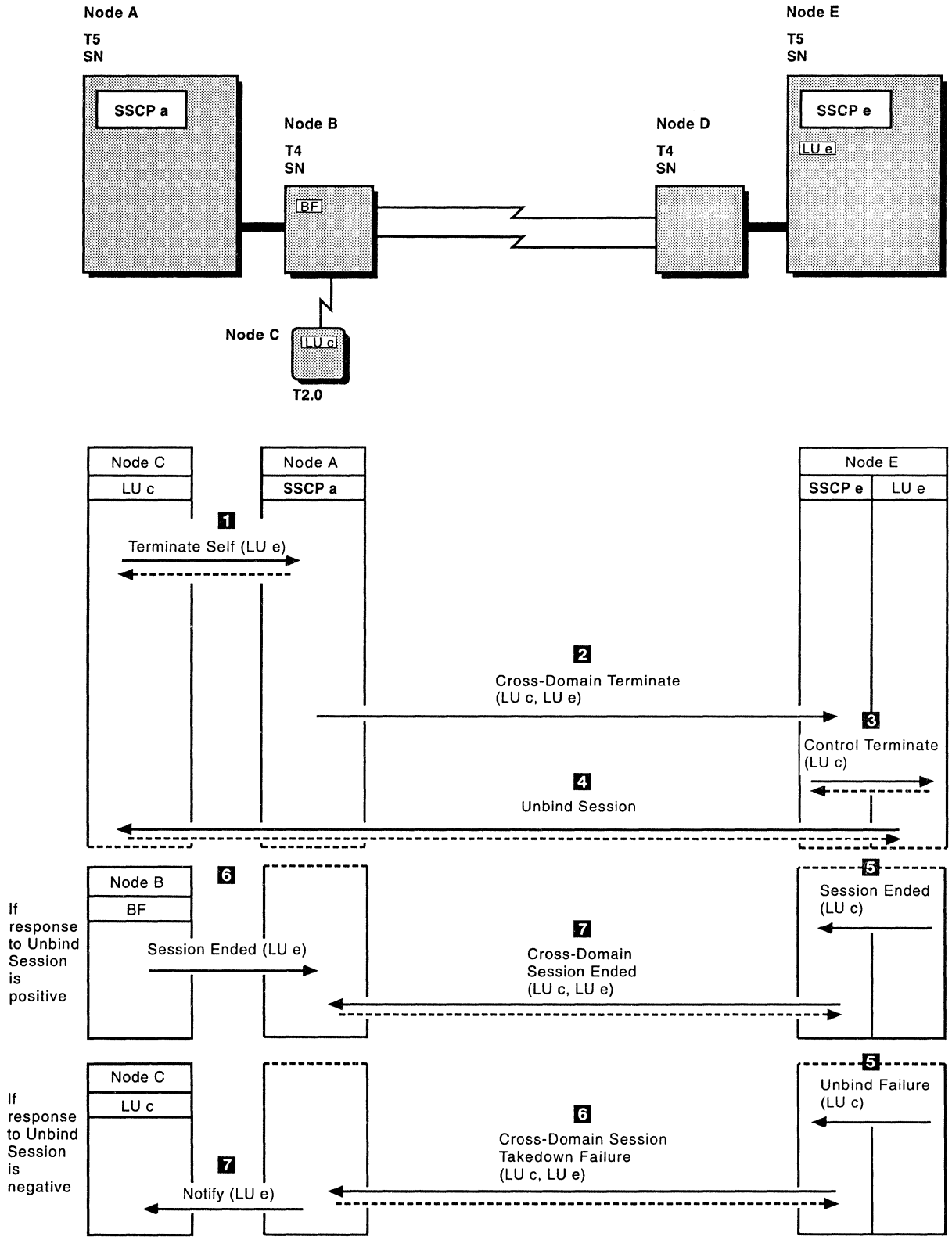


Figure 165 (Part 1 of 2). Terminating a Cross-Domain LU-LU Session in a Subarea Network

1. LU c requests that SSCP a help deactivate the session between LU c and LU e. LU e is the primary LU.
2. SSCP a tells SSCP e of LU c's request.
3. SSCP e tells LU e to deactivate its session with LU c.
4. LU e deactivates its session with LU c.

If the session is deactivated successfully:

5. LU e informs SSCP e that LU e has deactivated its session with LU c.
6. The boundary function in Node B informs SSCP a that the session between LU c and LU e has been deactivated. (Although the RU bearing this information has the network address of LU c in the origin field of its TH, this RU actually comes from the LU c boundary function in Node B.)
7. SSCP e informs SSCP a that the cross-domain session has been deactivated.

If the session is not deactivated successfully:

5. LU e informs SSCP e that LU e has failed in its attempt to deactivate a session with LU c.
6. SSCP e informs SSCP a that the attempt to deactivate the session between LU c and LU e has failed.
7. SSCP a informs LU c that the LU e believes that LU e's attempt to deactivate the session has failed.

Figure 165 (Part 2 of 2). Terminating a Cross-Domain LU-LU Session in a Subarea Network

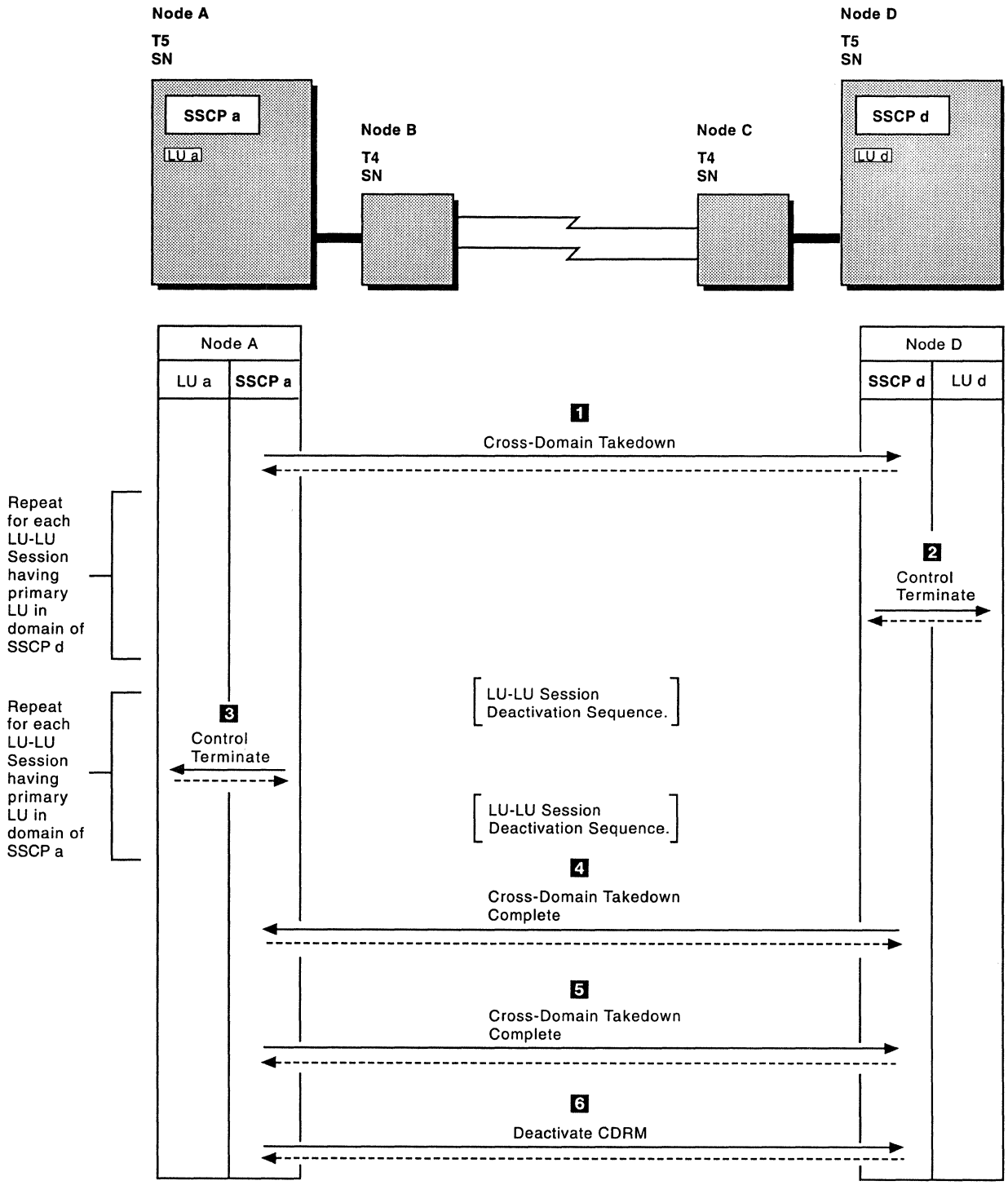


Figure 166 (Part 1 of 2). Cross-Domain Takedown Sequence in a Subarea Network

1. SSCP a initiates a procedure to cause the takedown of all cross-domain LU-LU sessions between LUs in SSCP a's domain and LU's in SSCP d's domain.

2. SSCP d initiates a session deactivation sequence for each LU-LU session involving a primary LU in SSCP d's domain and a secondary LU in SSCP a's domain.

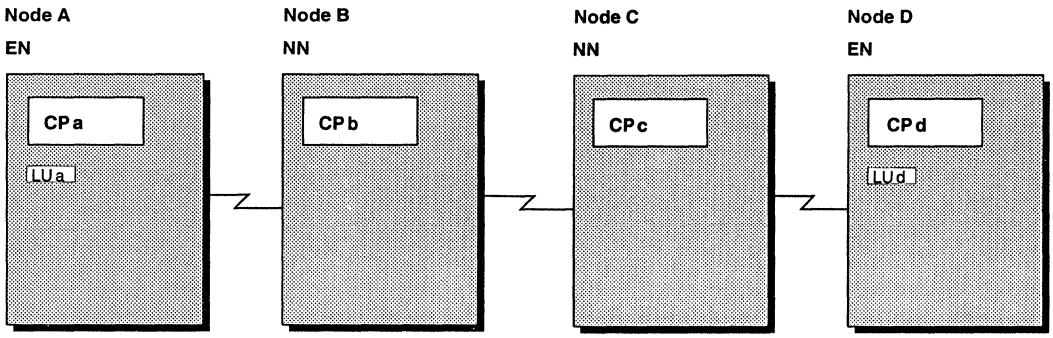
3. Concurrently, SSCP a initiates a session deactivation sequence for each LU-LU session involving a primary LU in SSCP a's domain and a secondary LU in SSCP d's domain.

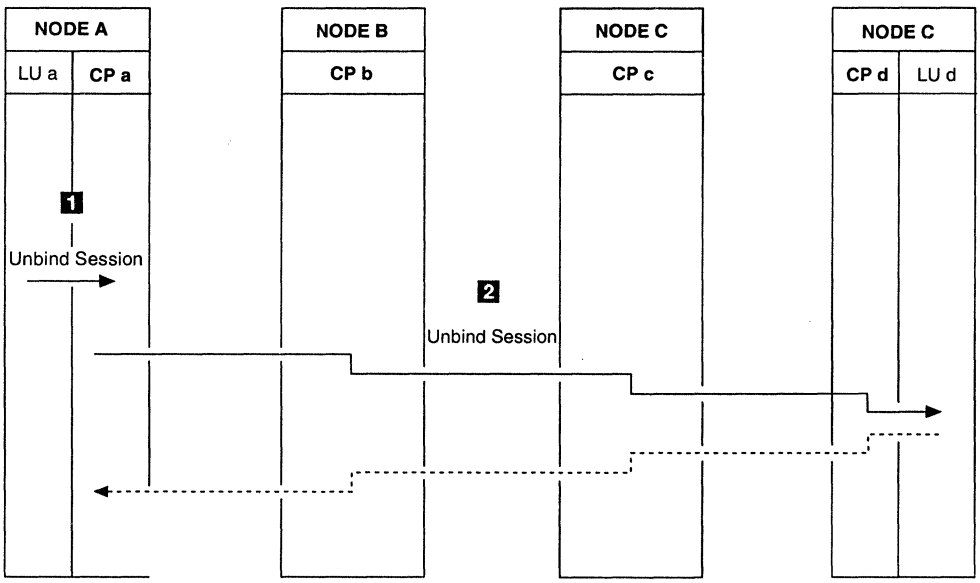
4. SSCP d informs SSCP a that all LU-LU sessions involving a primary LU in SSCP d's domain and a secondary LU in SSCP a's domain have been deactivated.

5. SSCP a informs SSCP d that all LU-LU sessions involving a primary LU in SSCP a's domain and a secondary LU in SSCP d's domain have been deactivated.

6. SSCP a deactivates its session with SSCP d.

Figure 166 (Part 2 of 2). Cross-Domain Takedown Sequence in a Subarea Network

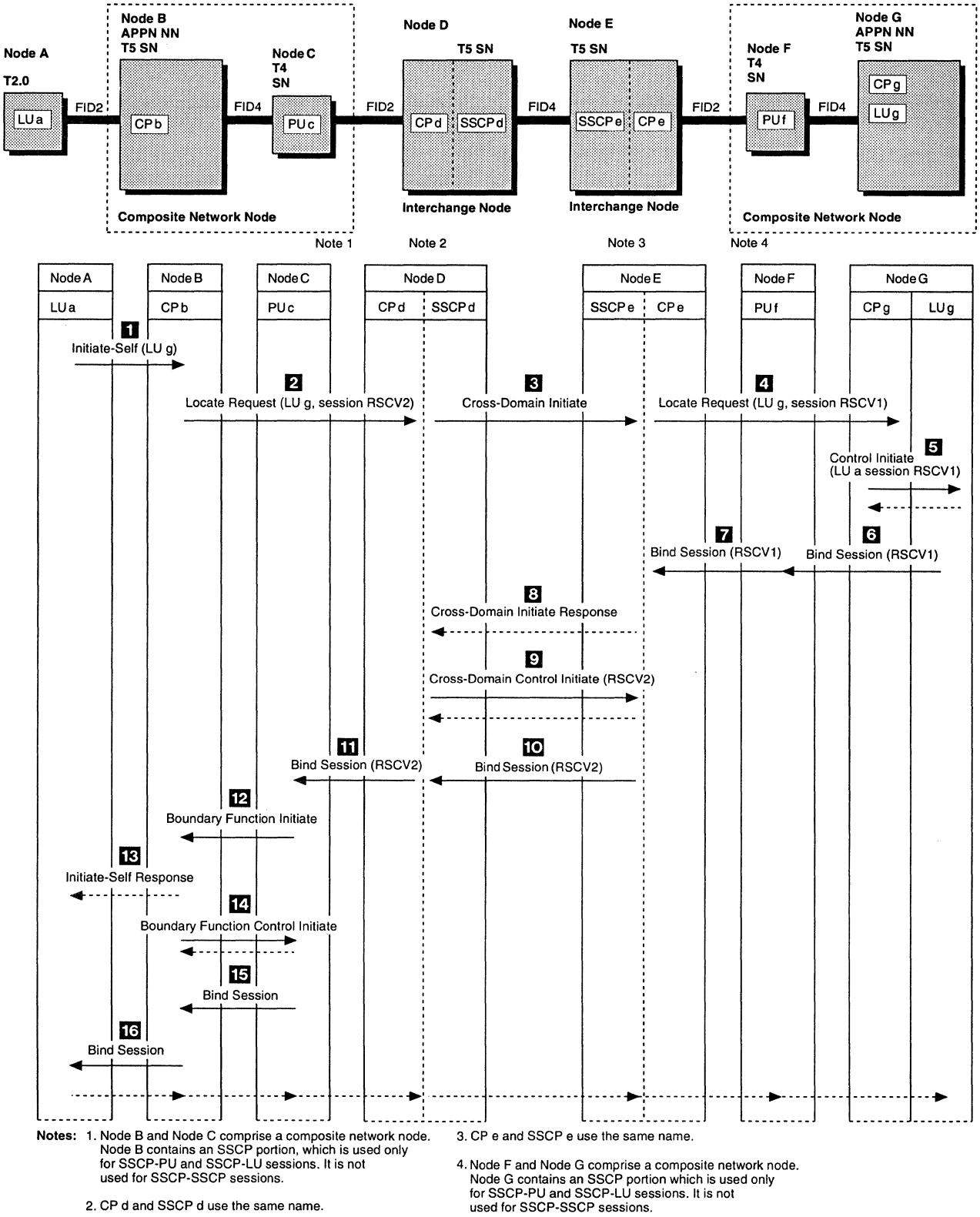




1 LU a sends an UNBIND request to CP a to deactivate its session with LU d, and dsmantles its half-session for the session.

2 CP a forwards the UNBIND to LU d and receives back the reply. In processing the UNBIND request and the UNBIND response, LU d dismantles its half-session for the session, and Nodes B and C dismantle their session connectors for the session. The address space managers in all CPs route the UNBIND request to each successive CP along the route.

Figure 167. Terminating an LU-LU Session in an APPN Network

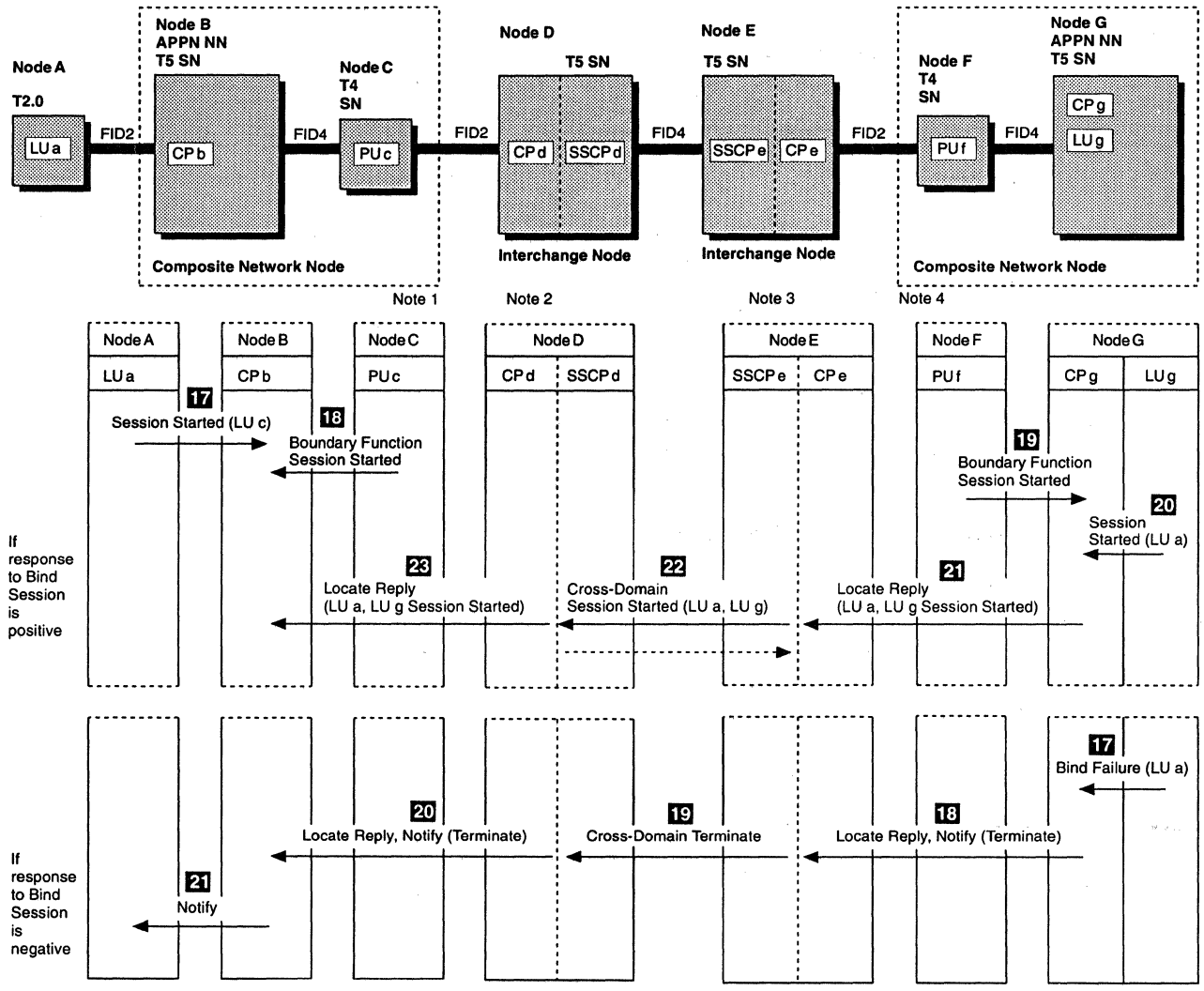


Notes:
1. Node B and Node C comprise a composite network node. Node B contains an SSCP portion, which is used only for SSCP-PU and SSCP-LU sessions. It is not used for SSCP-SSCP sessions.
2. CP d and SSCP d use the same name.
3. CP e and SSCP e use the same name.
4. Node F and Node G comprise a composite network node. Node G contains an SSCP portion which is used only for SSCP-PU and SSCP-LU sessions. It is not used for SSCP-SSCP sessions.

Figure 168 (Part 1 of 4). SLU-Initiated APPN-Subarea-APPN Session

1. LU a sends CP b an Initiate-Self to request that CP b help set up a session between LU a and LU g.
2. CP b sends (the CP d portion of) Node D a Locate request to search for LU g. The request includes the RSCV2. The RSCV2 is precalculated due to the cross-domain subarea portions in this configuration. The Locate request is sent toward LU g. The Locate request carries the mode name for the session between LU g and LU a.
3. Node D transforms the Locate request into a Cross-Domain Initiate. (The SSCP d portion of) Node D sends the Cross-Domain Initiate to (the SSCP e portion of) Node E. The mode name for the session is resolved to a class of service (COS) at the SSCP acting as SSCP of the SLU, that is, SSCP d.
4. Node E transforms the Cross-Domain Initiate back into a Locate request, containing RSCV1. The request includes the RSCV1. The RSCV1 is precalculated due to the cross-domain subarea portions in this configuration. (The CP e portion of Node E) sends the Locate request to CP g.
5. CP g sends a Control Initiate request containing the RSCV1 to LU g.
6. CP g initiates session activation by sending a Bind Session to PU f.
7. PU f transforms the FID4 Bind Session into a FID2 Bind Session. PU f forwards the Bind Session to CP e.
8. SSCP e sends a response to the Cross-Domain Initiate (which was sent in step 3) to SSCP d.
9. (The SSCP d portion of) Node D tells (the SSCP e portion of) Node E the session rules for the session between LU a and LU g. At this time, the COS is mapped to a virtual route list at the SSCP acting as SSCP for the PLU, that is SSCP e. Also, RSCV2 is sent to Node E.
10. Node E transforms the FID2 Bind Session into a FID4 Bind Session. Node E forwards the Bind Session to Node D.
11. Node D transforms the FID4 Bind Session into a FID2 Bind Session and forwards the Bind Session to PU c.
12. PU c sends Node B (which contains an SSCP as part of the composite network node) a Boundary Function Initiate request to continue session setup.
13. CP b responds to the Initiate-Self request issued by LU a in step 1.
14. Node B sends a Boundary Function Control Initiate request to PU c to tell PU c the session rules for the session between Node B and PU c.
15. Node C transforms the FID2 Bind Session into a FID4 Bind Session. PU c forwards the Bind Session to CP b.
16. Node B transforms the FID4 Bind Session into a FID2 Bind Session. CP b forwards the Bind Session to LU a.

Figure 168 (Part 2 of 4). SLU-Initiated APPN-Subarea-APPN Session



Notes:

1. Node B and Node C comprise a composite network node. Node B contains an SSCP portion which is used only for SSCP-PU and SSCP-LU sessions. It is not used for SSCP-SSCP sessions.
2. CP d and SSCP d use the same name.
3. CP e and SSCP e use the same name.
4. Node F and Node G comprise a composite network node. Node G contains an SSCP portion which is used only for SSCP-PU and SSCP-LU sessions. It is not used for SSCP-SSCP sessions.

Figure 168 (Part 3 of 4). SLU-Initiated APPN-Subarea-APPN Session

If the session is activated successfully:

17 LU a informs CP b that LU g has activated a session with LU a.

18 PU c informs CP b that the Bind Session response has reached PU c.

19 PU f informs CP g that the Bind Session response has reached PU f.

20 LU g informs CP g that LU g has activated a session with LU a.

21 CP g sends (the CP e portion of Node E) a Locate reply in reponse to the Locate request, which was sent in step 2. The Locate reply contains notification that a session was started.

22 Node E transforms the Locate reply into a Cross-Domain Session Started. (The SSCP e portion of) Node E sends the Cross-Domain Session Started to (the SSCP d portion of) Node D.

23 Node D transforms the Cross-Domain Session Started back into a Locate reply. The Locate reply contains notification that a session was started. (The CP d portion of) Node D sends the Locate reply to CP b to indicate that a session has been activated between LU g and LU a.

If the session is not activated successfully:

17 LU g informs CP g that LU g has failed in its attempt to activate a session with LU a.

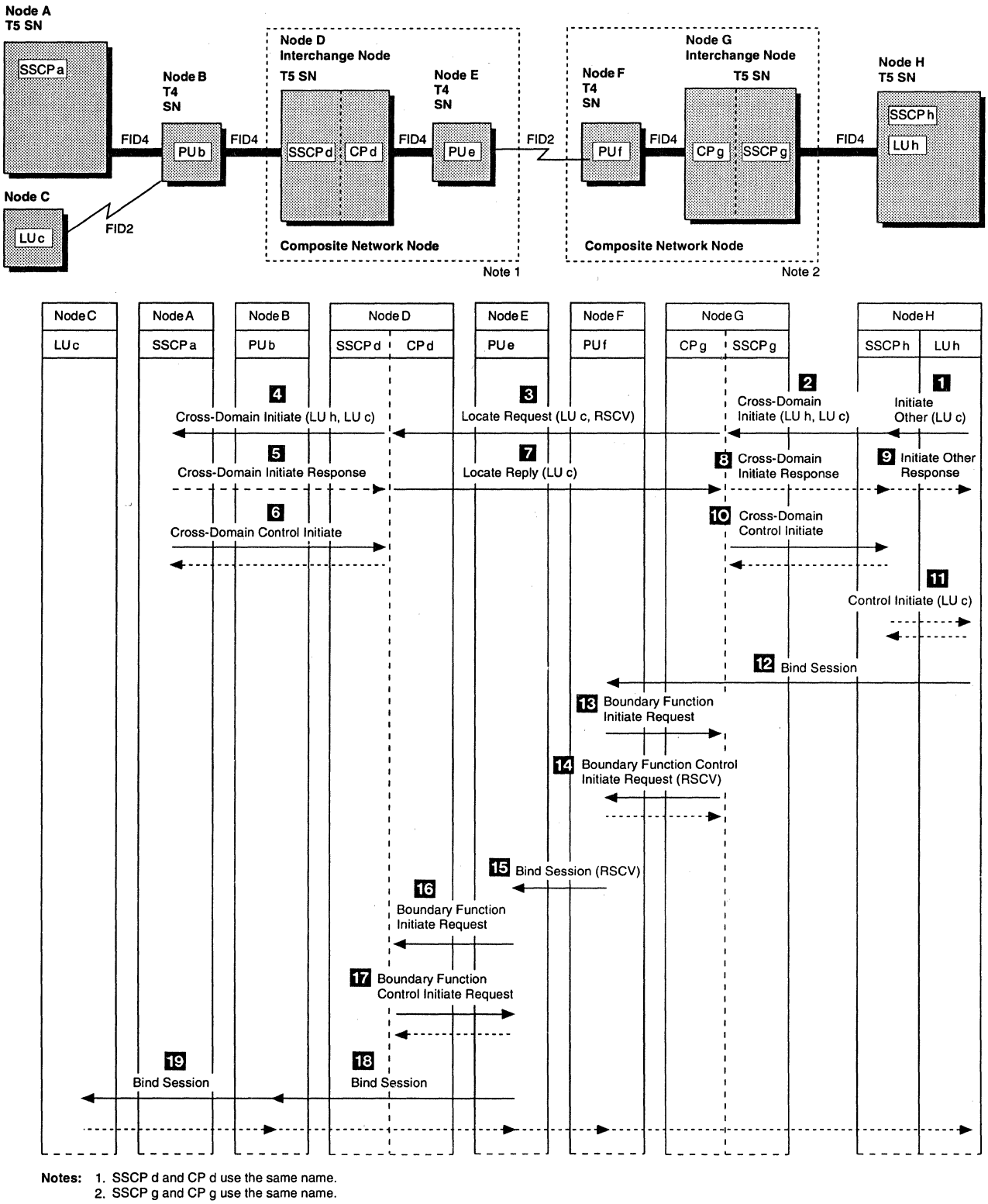
18 CP g sends (the CP e portion of) Node E a Locate reply with notification that the session was not established.

19 Node E transforms the Locate reply, Notify(Terminate) into a Cross-Domain Terminate. (The SSCP e portion of) Node E sends the Cross-Domain Terminate to (the SSCP d portion of) Node D.

20 Node D transforms the Cross-Domain Terminate back into a Locate reply, Notify (Terminate). (The CP d portion of) Node D sends the Locate reply, Notify(Terminate) to CP b.

21 CP b tells LU a that LU g's attempt to activate the session between LU g and LU a has failed.

Figure 168 (Part 4 of 4). SLU-Initiated APPN-Subarea-APPN Session

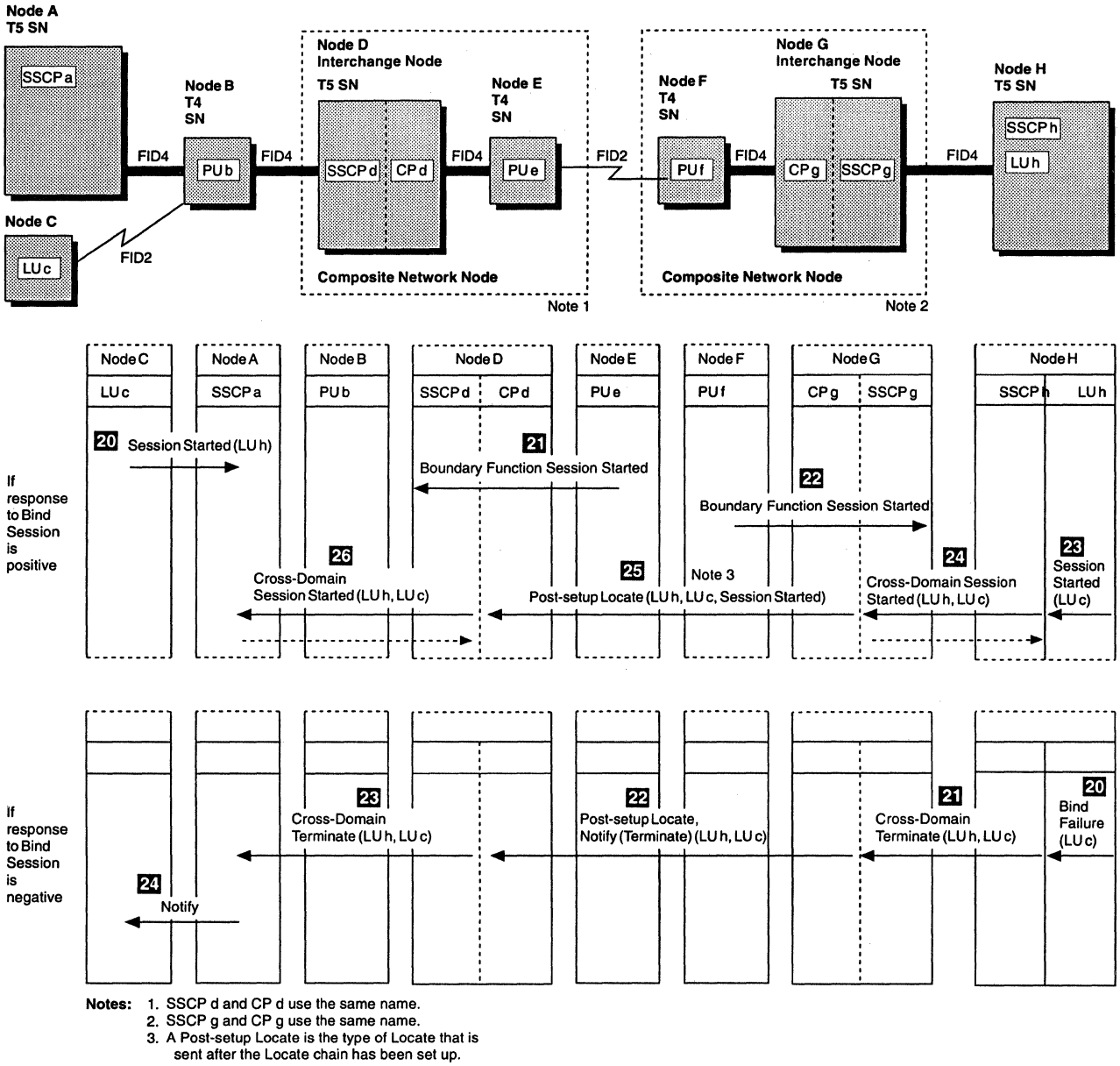


Notes: 1. SSCP d and CP d use the same name.
2. SSCP g and CP g use the same name.

Figure 169 (Part 1 of 4). PLU-Initiated Subarea-APPN-Subarea Session

1. LU h sends SSCP h an Initiate-Other to initiate a session between LU h and LU c.
2. SSCP h sends a Cross-Domain Initiate request to (the SSCP g portion of) Node G. The mode name for the session between LU h and SSCP g is carried with the Cross-Domain Initiate request.
3. Node G transforms the Cross-Domain Initiate request to a Locate request. The request includes the RSCV. The RSCV is precalculated due to the cross-domain subarea portions in this configuration. (The CP g portion of) Node G sends the Locate request to (the CP d portion) of Node D.
4. Node D transforms the Locate request into a Cross-Domain Initiate request. (The SSCP d portion of) Node D sends the Cross-Domain Initiate request to SSCP a. The mode name for the session between SSCP d and LU c is carried within the Cross-Domain Initiate request.
5. SSCP a sends a response to the Cross-Domain Initiate to (the SSCP d portion of) Node D. The mode name for the session is resolved to a class of service (COS) for the portion of the session between SSCP d and LU c at the SSCP acting as SSCP of the SLU, that is, SSCP a.
6. SSCP a tells SSCP d the session rules for the session between LU c and LU h. At this time the COS is mapped to a virtual route list at the SSCP acting as SSCP for the PLU, that is SSCP d (for the portion of the session between SSCP d and LU c).
7. Node D transforms the Cross-Domain Initiate response into a Locate reply. (The CP d portion of) Node D sends the Locate reply to (the CP g portion of) Node G.
8. Node G transforms the Locate reply into a Cross-Domain Initiate Response. (The SSCP g portion of) Node G sends the Cross-Domain Initiate response to SSCP h. The mode name for the session is resolved to a class of service (COS) for the portion of the session between SSCP g and LU h at the SSCP acting as SSCP of the SLU, that is SSCP g.
9. SSCP h sends LU h a positive response to the Initiate-Other, which was sent in step 1.
10. SSCP g tells SSCP h the session rules for the session between LU c and LU h. At this time, the COS is mapped to a virtual route list at the SSCP acting as SSCP for the PLU, that is SSCP h (for the portion of the session between between SSCP g and LU h).
11. SSCP h sends a Control Initiate to LU h to tell LU h to activate a session with LU c using the session rules passed by SSCP g in step 10.
12. LU h initiates session activation by sending a Bind Session to PU f using the rules given by SSCP g in step 10.
13. PU f sends SSCP g (PU f's owning SSCP) a Boundary Function Initiate to obtain the RSCV for the portion of the session between PU f and PU e.
14. SSCP g sends PU f a Boundary Function Control Initiate with the RSCV.
15. PU f transforms the FID4 Bind Session into a FID2 Bind Session. PU f forwards the Bind Session (with RSCV) to PU e.
16. PU e sends SSCP d (PU e's owning SSCP) a Boundary Function Initiate to continue session setup.
17. SSCP d sends PU e a Boundary Function Control Initiate to tell PU e the session rules for the portion of the session between SSCP d and PU e.
18. PU e transforms the FID2 Bind Session into a FID4 Bind Session. PU e forwards the Bind Session to PU b.
19. PU b transforms the FID4 Bind Session into a FID2 Bind Session and forwards the Bind Session to LU c.

Figure 169 (Part 2 of 4). PLU-Initiated Subarea-APPN-Subarea Session



Notes: 1. SSCP d and CP d use the same name.
2. SSCP g and CP g use the same name.
3. A Post-setup Locate is the type of Locate that is sent after the Locate chain has been set up.

Figure 169 (Part 3 of 4). PLU-Initiated Subarea-APPN-Subarea Session

If the session is activated successfully:

20 LU c informs SSCP a that LU h has activated a session with LU c.

21 PU e informs SSCP d that the Bind Session response has reached PU e.

22 PU f informs SSCP g that the Bind Session response has reached PU f.

23 LU h informs SSCP h that LU h has activated a session with LU c.

24 SSCP h sends a Cross-Domain Session Started to (the SSCP g portion of) Node G to inform Node G of this fact.

25 Node G transforms the Cross-Domain Session Started into a post-setup Locate. (The CP g portion of) Node G sends the Locate to (the CP d portion of) Node D.

26 Node D transforms the Locate back into a Cross-Domain Session Started. (The SSCP d portion of) Node D sends the Cross-Domain Session Started to SSCP a.

If the session is not activated successfully:

20 LU h informs SSCP h that LU h has failed in its attempt to activate a session with LU c.

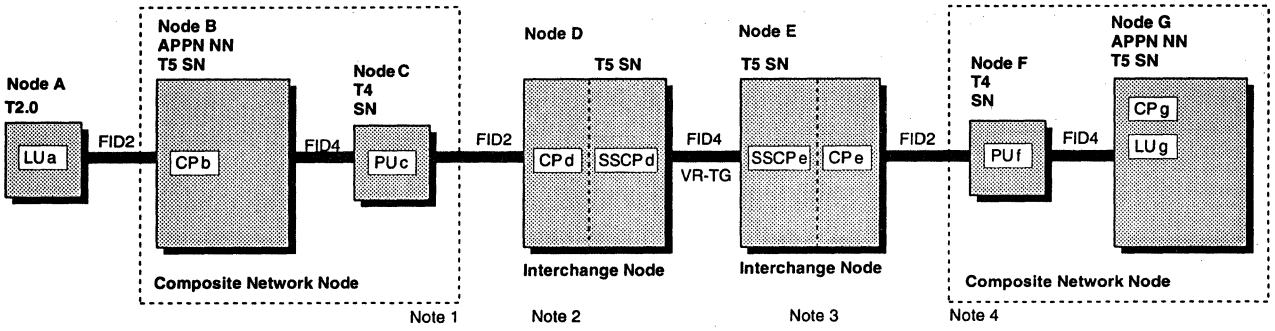
21 SSCP h sends a Cross-Domain Terminate to (the SSCP g portion of) Node G to inform Node G of this fact.

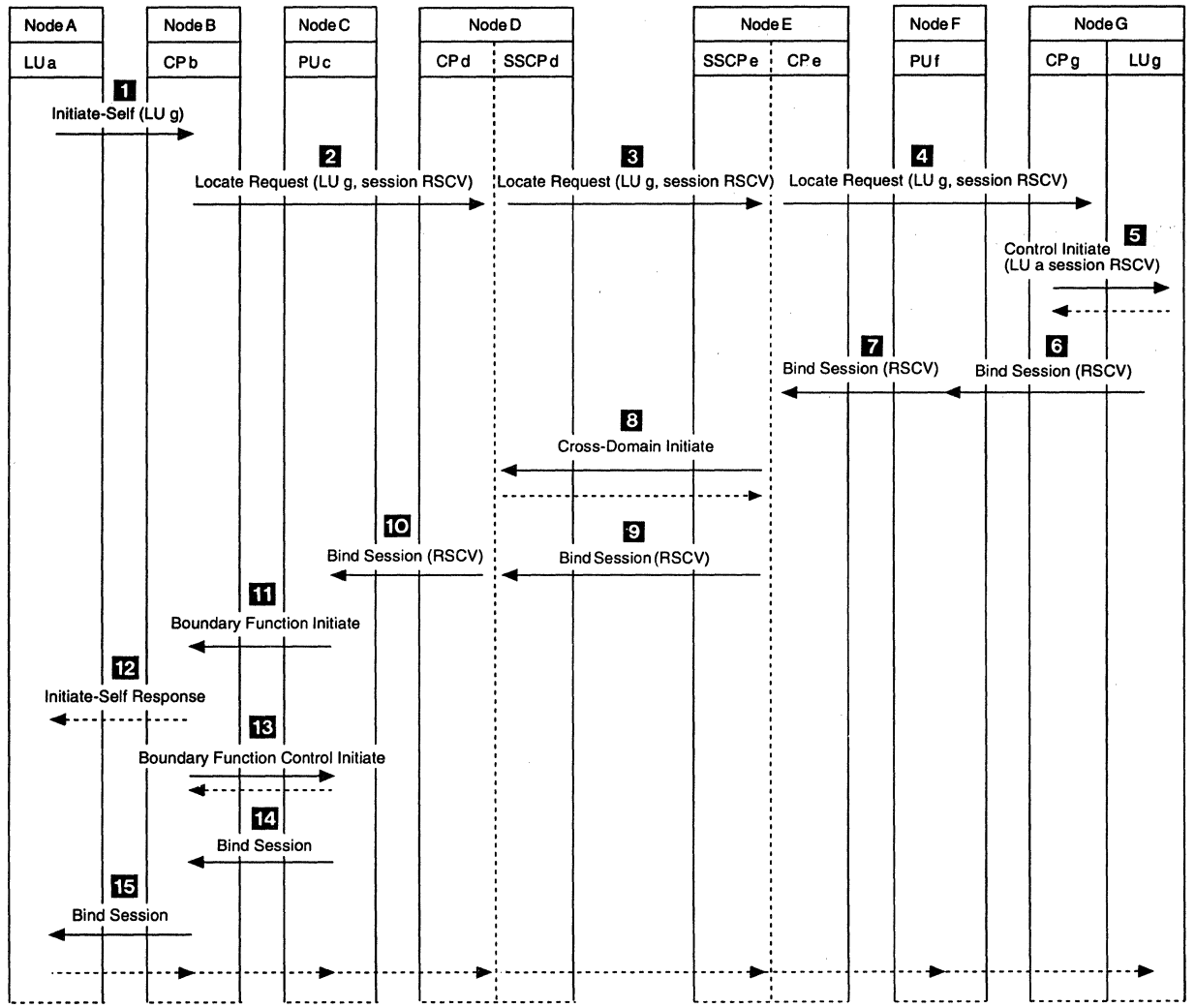
22 Node G transforms the Cross-Domain Terminate into a Locate,Notify(Terminate). (The CP g portion of) Node G sends the Locate,Notify(Terminate) to Node D.

23 Node D transforms the Locate,Notify(Terminate) back into a Cross-Domain Terminate. (The SSCP d portion of) Node D sends the Cross-Domain Terminate to SSCP a.

24 SSCP a tells LU c that LU h's attempt to activate the session between LU h and LU c has failed.

Figure 169 (Part 4 of 4). PLU-Initiated Subarea-APPN-Subarea Session





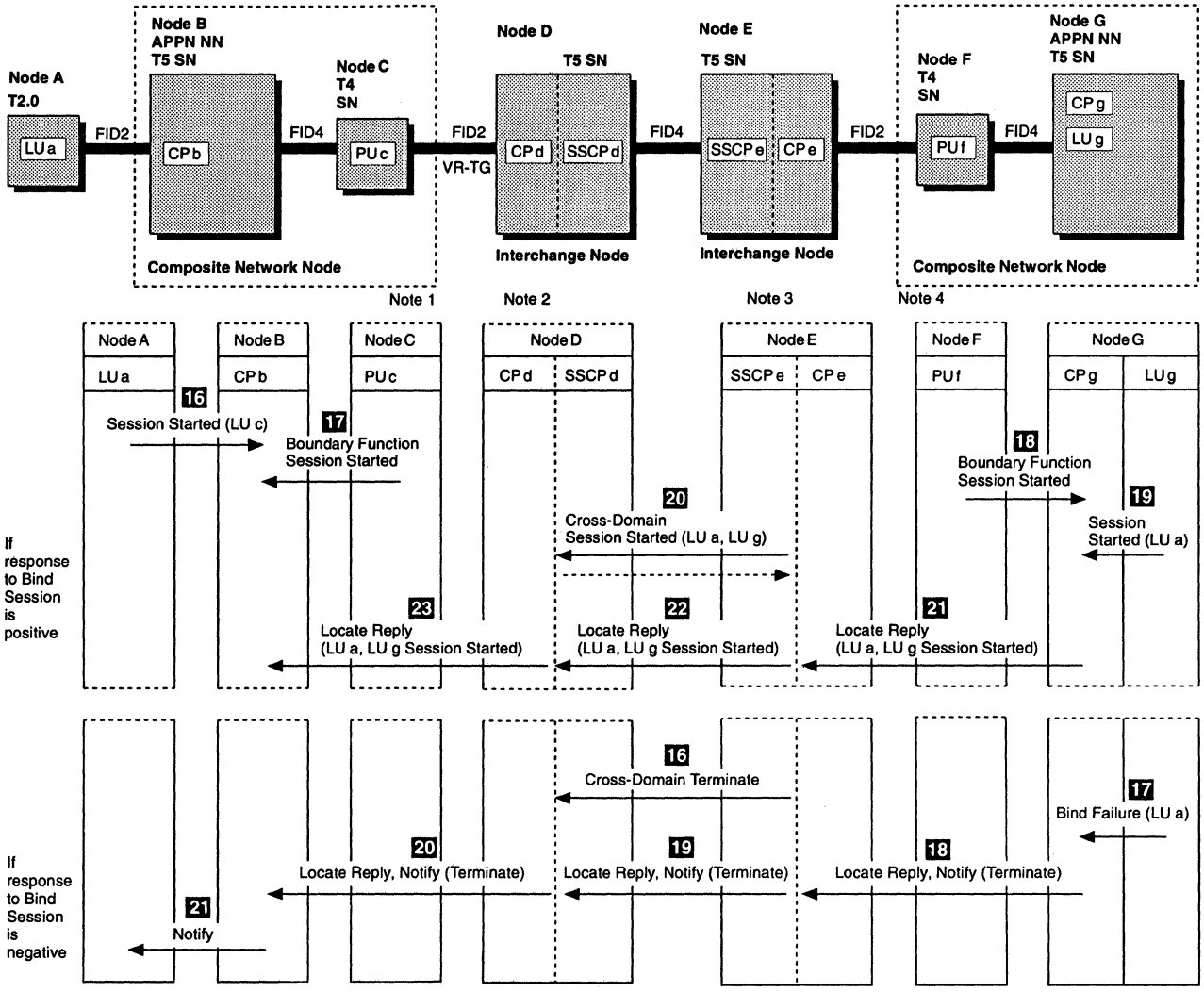
Notes:

1. Node B and Node C comprise a composite network node. Node B contains an SSCP portion which is used only for SSCP-PU and SSCP-LU sessions. It is not used for SSCP-SSCP sessions.
2. CP d and SSCP d use the same name.
3. CP e and SSCP e use the same name.
4. Node F and Node G comprise a composite network node. Node G contains an SSCP portion which is used only for SSCP-PU and SSCP-LU sessions. It is not used for SSCP-SSCP sessions.

Figure 170 (Part 1 of 4). SLU-Initiated APPN-Subarea-APPN Session Using a VR-Based TG

1. LU a sends CP b an Initiate-Self to request that CP b help set up a session between LU a and LU g.
2. CP b sends (the CP d portion of) Node D a Locate request to search for LU g. The request includes the RSCV. The RSCV is precalculated due to the cross-domain subarea portions in this configuration. The Locate request is sent toward LU g. The Locate request carries the mode name for the session between LU g and LU a.
3. (The SSCP d portion of) Node D forwards the Locate request to (the SSCP e portion of) Node E.
4. (The CP e portion of) Node E forwards the Locate request to CP g.
5. CP g sends a Control Initiate request containing the RSCV to LU g.
6. LU g initiates session activation by sending a Bind Session to PU f.
7. PU f forwards the Bind Session to (the CP e portion of) Node E.
8. (The SSCP e portion of) Node E sends the Cross-Domain Initiate to (the SSCP e portion of) Node D.
9. (The SSCP e portion of) Node E forwards the Bind Session to (the SSCP d portion of) Node D.
10. (The CP d portion of) Node D forwards the Bind Session to PU c.
11. PU c sends Node B a Boundary Function Initiate request to continue session setup.
12. CP b responds to the Initiate-Self request issued by LU a in step 1.
13. Node B sends a Boundary Function Control Initiate request to PU c to tell PU c the session rules for the session between Node B and PU c.
14. PU c forwards the Bind Session to CP b.
15. CP b forwards the Bind Session to LU a.

Figure 170 (Part 2 of 4). SLU-Initiated APPN-Subarea-APPN Session Using a VR-Based TG



Notes:

1. Node B and Node C comprise a composite network node. Node B contains an SSCP portion which is used only for SSCP-PU and SSCP-LU sessions. It is not used for SSCP-SSCP sessions.
2. CP d and SSCP d use the same name.
3. CP e and SSCP e use the same name.
4. Node F and Node G comprise a composite network node. Node G contains an SSCP portion which is used only for SSCP-PU and SSCP-LU sessions. It is not used for SSCP-SSCP sessions.

Figure 170 (Part 3 of 4). SLU-Initiated APPN-Subarea-APPN Session Using a VR-Based TG

If the session is activated successfully:

16 LU a informs CP b that LU g has activated a session with LU a.

17 PU c informs CP b that the Bind Session response has reached PU c.

18 PU f informs CP g that the Bind Session response has reached PU f.

19 LU g informs CP g that LU g has activated a session with LU a.

20 (The SSCP e portion of) Node E send a Cross-Domain Session Started to (the SSCP d portion of) Node D.

21 CP g sends (the CP e portion of) Node E a Locate reply in reponse to the Locate request, which was sent in step 2. The Locate reply contains notification that a session was started.

22 (The SSCP e portion of) Node E forwards the Locate reply to (the SSCP d portion of) Node D.

23 (The CP d portion of) Node D forwards the Locate reply to CP b.

If the session is not activated successfully:

16 (The SSCP e portion of) Node E sends a Cross-Domain Terminate to (the SSCP d portion of) Node D.

17 LU g informs CP g that LU g has failed in its attempt to activate a session with LU a.

18 CP g sends (the CP e portion of) Node E a Locate reply with notification that the session was not established.

19 (The SSCP e portion of) Node E forwards the Locate reply, Notify(Terminate) to (the SSCP d portion of) Node D.

20 (The CP d portion of) Node D forwards the Locate reply, Notify(Terminate) to CP b.

21 CP b tells LU a that LU g's attempt to activate the session between LU g and LU a has failed.

Figure 170 (Part 4 of 4). SLU-Initiated APPN-Subarea-APPN Session Using a VR-Based TG

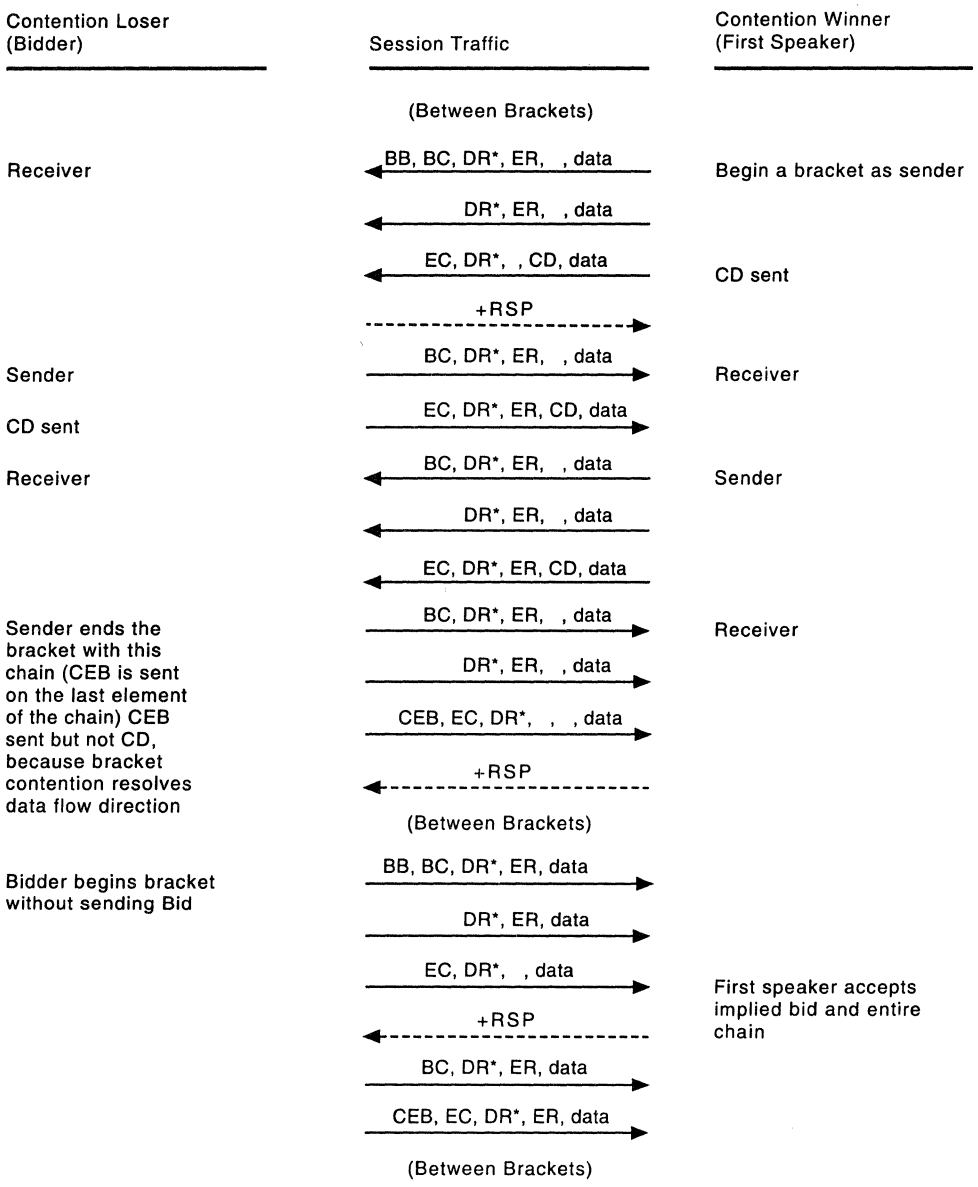


Figure 171 (Part 1 of 2). Communication Using Brackets in a Half-Duplex Flip-Flop Mode

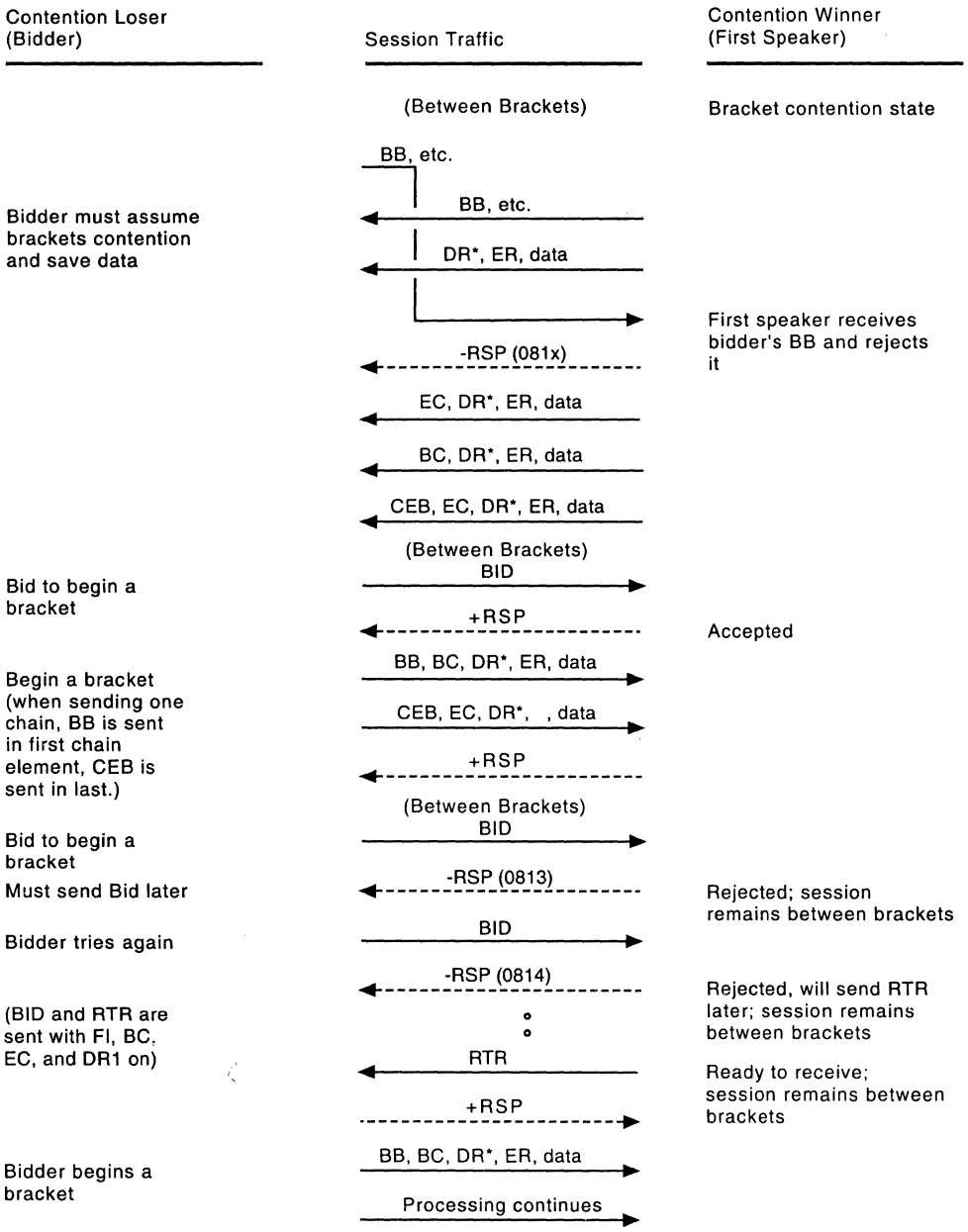


Figure 171 (Part 2 of 2). Communication Using Brackets in a Half-Duplex Flip-Flop Mode

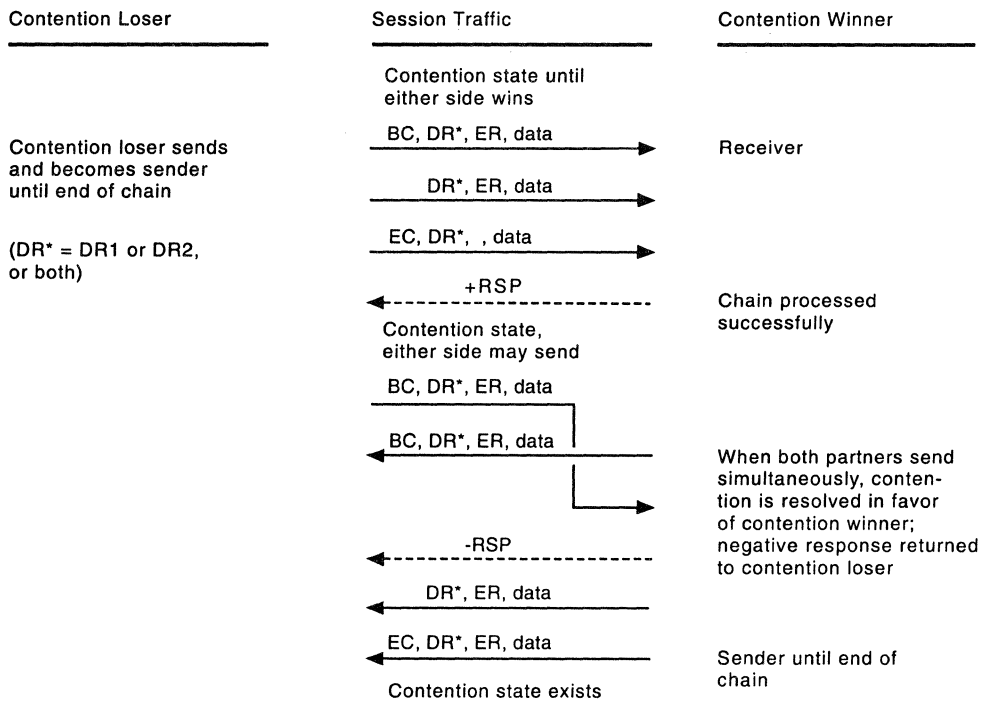


Figure 172. Communication Using Half-Duplex Contention Protocols

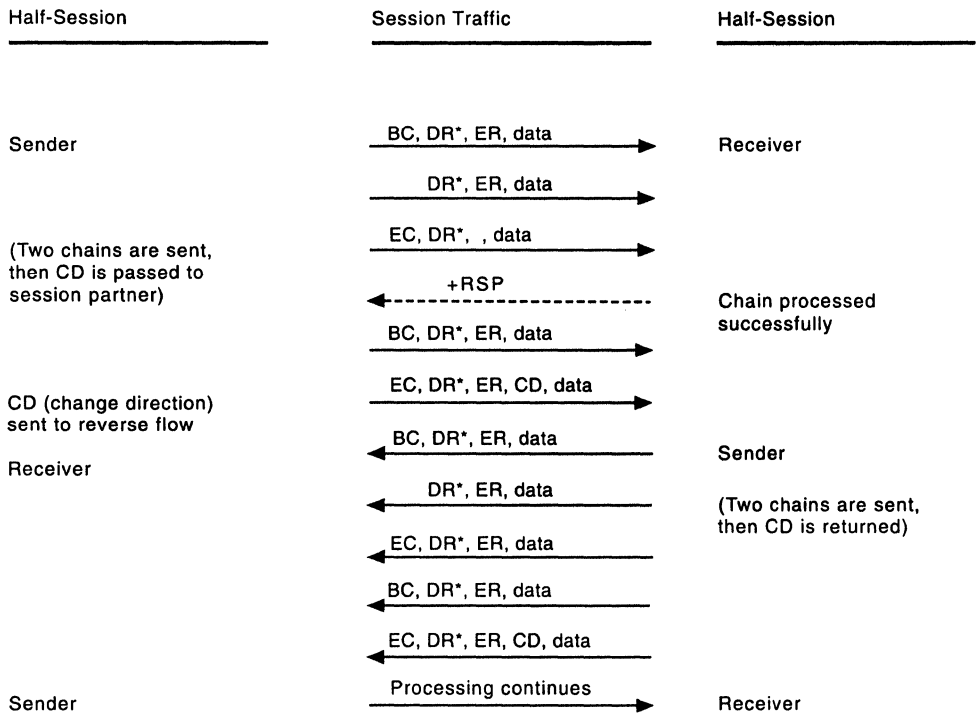


Figure 173. LU-LU Communication Using Half-Duplex Flip-Flop Protocols

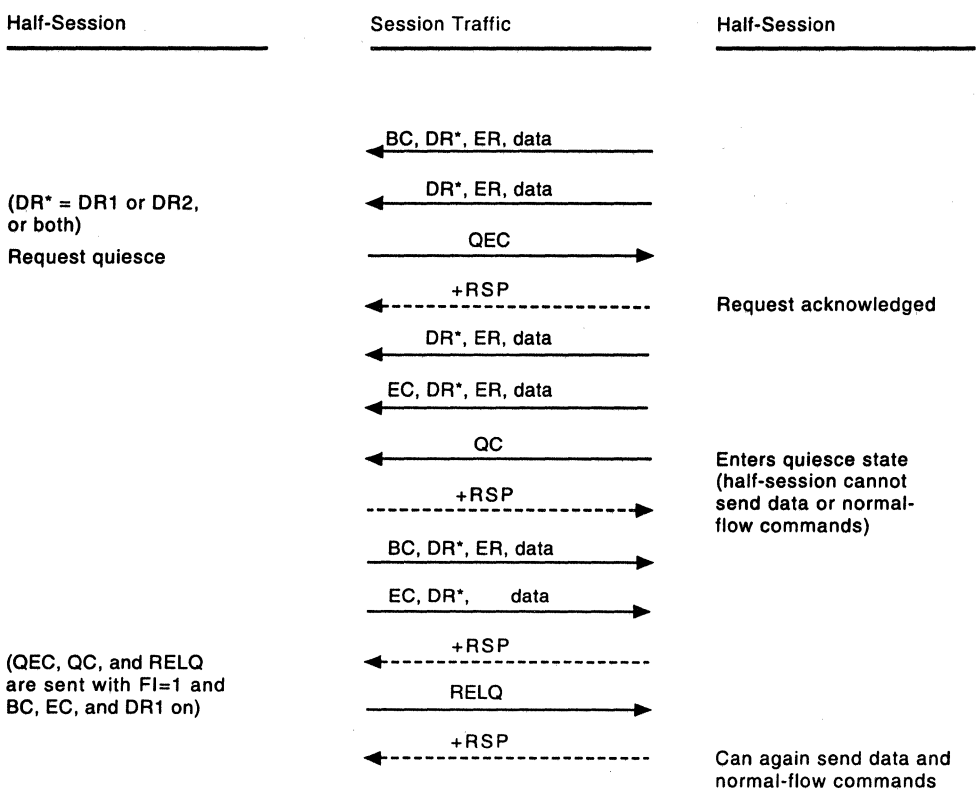
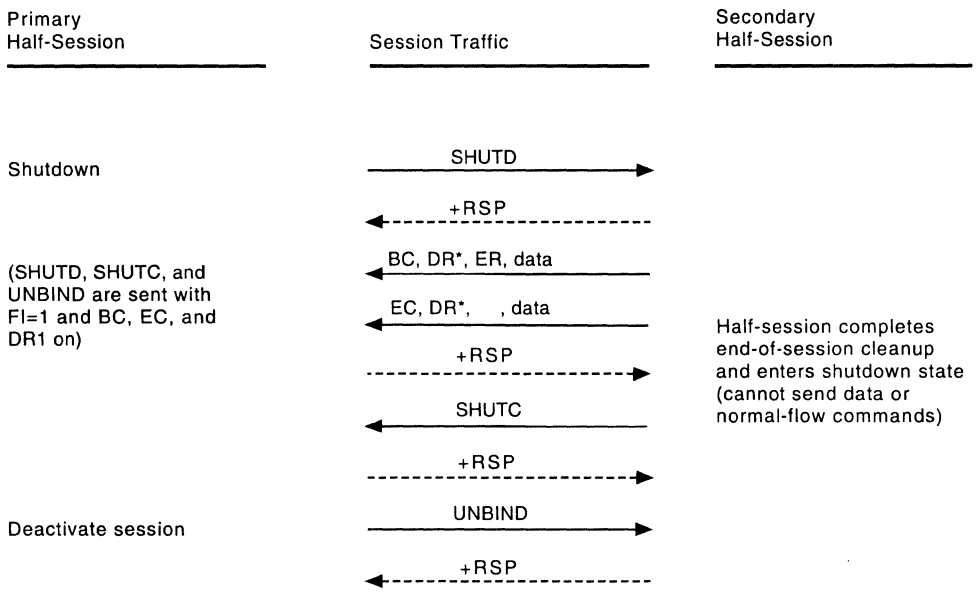
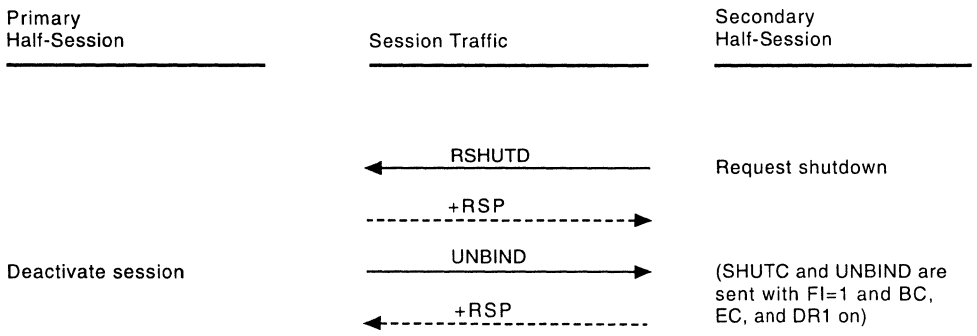


Figure 174. Protocols for Quiescing Data Flow



(a) Orderly shutdown initiated by primary half-session.



(b) Orderly shutdown initiated by secondary half-session.

Figure 175. Protocols for Terminating LU-LU Sessions

Glossary, Bibliography, and Index

Glossary

This glossary includes terms and definitions from:

- The *American National Standard Dictionary for Information Systems*, ANSI X3.172-1990, copyright 1990 by the American National Standards Institute (ANSI). Copies may be purchased from the American National Standards Institute, 11 West 42nd Street, New York, New York 10036. Definitions are identified by the symbol (A) after the definition.
- The ANSI/EIA Standard—440-A, *Fiber Optic Terminology*. Copies may be purchased from the Electronic Industries Association, 2001 Pennsylvania Avenue, N.W., Washington, DC 20006. Definitions are identified by the symbol (E) after the definition.
- The *Information Technology Vocabulary*, developed by Subcommittee 1, Joint Technical Committee 1, of the International Organization for Standardization and the International Electrotechnical Commission (ISO/IEC JTC1/SC1). Definitions of published parts of this vocabulary are identified by the symbol (I) after the definition; definitions taken from draft international standards, committee drafts, and working papers being developed by ISO/IEC JTC1/SC1 are identified by the symbol (T) after the definition, indicating that final agreement has not yet been reached among the participating National Bodies of SC1.
- The Network Working Group Request for Comments: 1208.

The following cross-references are used in this glossary:

Contrast with: This refers to a term that has an opposed or substantively different meaning.

Synonym for: This indicates that the term has the same meaning as a preferred term, which is defined in its proper place in the glossary.

Synonymous with: This is a backward reference from a defined term to all other terms that have the same meaning.

See: This refers the reader to multiple-word terms that have the same last word.

See also: This refers the reader to terms that have a related, but not synonymous, meaning.

Deprecated term for: This indicates that the term should not be used. It refers to a preferred term, which is defined in its proper place in the glossary.

A

access method. (1) A technique, implemented in software, that controls the flow of information through a network. (2) A technique for moving data between main storage and input/output devices.

ACF/NCP. Advanced Communications Function for the Network Control Program. Synonym for *NCP*.

ACF/VTAM. Advanced Communications Function for the Virtual Telecommunications Access Method. Synonym for *VTAM*.

adaptive pacing. Synonym for *adaptive session-level pacing* and *virtual route pacing*.

adaptive rate-based (ARB) flow/congestion control. An HPR algorithm in which a sender regulates the flow of data over an RTP connection by adaptively changing the rate at which it sends data into the network, based on the rate at which the receiver processes the data. While sending the data, the sender regularly solicits processing-rate information from the receiver.

adaptive session-level pacing. A form of session-level pacing in which session components exchange pacing windows that may vary in size during the course of a session. This allows transmission within a network to adapt dynamically to variations in availability and demand of buffers on a session-by-session basis. Session-level pacing occurs within independent stages along the session path according to local congestion at the intermediate and endpoint nodes. Synonymous with *adaptive pacing* and *adaptive session pacing*. See *pacing*, *session-level pacing*, and *virtual route pacing*.

adaptive session pacing. Synonym for *adaptive session-level pacing*.

address space. A set of addresses used to uniquely identify network accessible units, sessions, adjacent link stations, and links in a node for each network in which the node participates. An APPN node has one address space for intranode routing and one for each transmission group on which it can send message units.

address space manager (ASM). A component in an APPN or LEN node that assigns and frees session addresses.

adjacent link station (ALS). (1) In SNA, a link station directly connected to a given node by a link connection over which network traffic can be carried.

Note: Several secondary link stations that share a link connection do not exchange data with each other and

therefore are not adjacent to each other. (2) With respect to a specific node, a link station partner in an adjacent node.

adjacent nodes. Two nodes connected together by at least one path that connects no other node. (T)

Advanced Peer-to-Peer Networking (APPN). An extension to SNA featuring (a) greater distributed network control that avoids critical hierarchical dependencies, thereby isolating the effects of single points of failure; (b) dynamic exchange of network topology information to foster ease of connection, reconfiguration, and adaptive route selection; (c) dynamic definition of network resources; and (d) automated resource registration and directory lookup. APPN extends the LU 6.2 peer orientation for end-user services to network control and supports multiple LU types, including LU 2, LU 3, and LU 6.2.

Advanced Peer-to-Peer Networking (APPN) end node. A node that provides a broad range of end-user services and supports sessions between its local control point (CP) and the CP in an adjacent network node. It uses these sessions to dynamically register its resources with the adjacent CP (its network node server), to send and receive directory search requests, and to obtain management services. An APPN end node can also attach to a subarea network as a peripheral node or to other end nodes.

Advanced Peer-to-Peer Networking (APPN) network. A collection of interconnected network nodes and their client end nodes.

Advanced Peer-to-Peer Networking (APPN) network node. A node that offers a broad range of end-user services and that can provide the following:

- Distributed directory services, including registration of its domain resources to a central directory server
- Topology database exchanges with other APPN network nodes, enabling network nodes throughout the network to select optimal routes for LU-LU sessions based on requested classes of service
- Session services for its local LUs and client end nodes
- Intermediate routing services within an APPN network

Advanced Peer-to-Peer Networking (APPN) node. An APPN network node or an APPN end node.

advanced program-to-program communication (APPC). (1) The general facility characterizing the LU 6.2 architecture and its various implementations in products. (2) Sometimes used to refer to the LU 6.2 architecture and its product implementations as a whole, or to an LU 6.2 product feature in particular, such as an APPC application program interface.

agent. (1) In OSI management, a user that, for a particular interaction, has assumed an agent role. (2) An OSI application entity that can accept a request to retrieve information from or to change a particular resource in a system environment. (3) A system that assumes an agent role.

alert. A message sent to a management services focal point in a network to identify a problem or an impending problem.

alias address. An address used by a gateway NCP and a gateway system services control point (SSCP) in one network to represent a logical unit (LU) or SSCP in another network.

alias name. A name that is defined in one network to represent a logical unit name in another interconnected network. The alias name does not have to be the same as the real name; if these names are not the same, translation is required.

ALS. Adjacent link station.

ANR. Automatic network routing.

API. Application program interface.

APPC. Advanced program-to-program communication.

application program interface (API). A functional interface supplied by the operating system or by a separately orderable licensed program that allows an application program written in a high-level language to use specific data or functions of the operating system or the licensed program.

application transaction program. A program written for or by a user to process the user's application; in an SNA network, an end user of a type 6.2 logical unit. Contrast with *service transaction program.*

APPN. Advanced Peer-to-Peer Networking.

APPN connection. A link over which APPN protocols are used.

APPN end node. See *Advanced Peer-to-Peer Networking (APPN) end node.*

APPN intermediate routing. The capability of an APPN network node to accept traffic from one adjacent node and pass it on to another, with awareness of session affinities in controlling traffic flow and outage notifications.

APPN intermediate routing network. The portion of an APPN network consisting of the network nodes and their interconnections.

APPN network. See *Advanced Peer-to-Peer Networking (APPN) network.*

APPN network node. See *Advanced Peer-to-Peer Networking (APPN) network node.*

APPN node. See *Advanced Peer-to-Peer Networking (APPN) node.*

APPN Topology and Accounting Manager (APPNTAM). A feature of the NetView program that allows a user, from a central management station, to view APPN network topologies and gather APPN network utilization data.

APPNTAM. Advanced Peer-to-Peer Networking Topology and Accounting Manager.

ARB. Adaptive rate-based.

ASM. Address space manager.

asynchronous (ASYNC). (1) Pertaining to two or more processes that do not depend upon the occurrence of specific events such as common timing signals. (T) (2) Without regular time relationship; unexpected or unpredictable with respect to the execution of program instructions.

automatic network routing (ANR). In HPR, a lower-layer routing mechanism that minimizes processing cycles and storage requirements for routing packets through intermediate nodes by employing source routing.

B

backbone. A set of nodes and their interconnecting links providing the primary data path across a network.

backup session. The session that replaces the failing primary extended recovery facility (XRF) session between a terminal user and the active subsystem.

basic conversation. An LU 6.2 conversation type specified by the allocating transaction program. Transaction programs using basic conversation have available to them a wider variety of LU 6.2 functions, but they are responsible for more of their own error recovery and must manage details of the data stream used on the conversation. Contrast with *mapped conversation.*

basic information unit (BIU). In SNA, the unit of data and control information passed between half-sessions. It consists of a request/response header (RH) followed by a request/response unit (RU).

basic link unit (BLU). In SNA, the unit of data and control information transmitted over a link by data link control.

basic transmission unit (BTU). In SNA, the unit of data and control information passed between path control components. A BTU can consist of one or more path information units (PIUs). See also *blocking of PIUs.*

BF. Boundary function.

bidder. Synonym for *bidder session.*

bidder session. The half-session defined at session activation as having to request and receive permission from the other half-session to begin a bracket. Contrast with *first-speaker session.* Synonym for *contention-loser session.* Synonymous with *bidder.*

binary synchronous communication (BSC). A form of telecommunication line control that uses a standard set of transmission control characters and control character sequences, for binary synchronous transmission of binary-coded data between stations. Contrast with *Synchronous Data Link Control (SDLC).*

BIND pacing. A technique by which the address space manager (ASM) at one node controls the rate of transmission of BIND requests of a sending ASM at another node. BIND pacing can be used to prevent BIND standoff, in which each of two nodes has reserved most of its resources for sessions it is attempting to initiate through the other and thus rejects any BINDs received from the other.

BIU. Basic information unit.

BIU segment. In SNA, the portion of a basic information unit (BIU) that is contained within a path information unit (PIU). It consists of either a request/response header (RH) followed by all or part of a request/response unit (RU), or of only a part of an RU.

blocking of PIUs. In SNA, an optional function of path control that combines multiple path information units (PIUs) in a single basic transmission unit (BTU).

Note: When blocking is not done, a BTU consists of one PIU.

BLU. Basic link unit.

border node. An APPN network node that interconnects APPN networks having independent topology databases in order to support LU-LU sessions between these networks. See *extended border node* and *peripheral border node.*

boundary function. (1) In SNA, a capability of a subarea node to provide protocol support for attached

peripheral nodes, such as: (a) interconnecting subarea path control and peripheral path control elements, (b) performing session sequence numbering for low-function peripheral nodes, and (c) providing session-level pacing support. (2) In SNA, the component that provides these capabilities.

boundary node (BN). In SNA, a subarea node with boundary function.

Note: A subarea node may be a boundary node, an intermediate routing node, both, or neither, depending on how it is used in the network.

bracket. In SNA, one or more chains of request units and their responses that are exchanged between two session partners and that represent a transaction between them. A bracket must be completed before another bracket can be started. Examples of brackets are database inquiries/replies, update transactions, and remote job entry output sequences to workstations. ring to the using node's storage, maintaining error counters, observing Medium Access Control (MAC) sublayer protocols (for address acquisition, error reporting, or other duties), and (in the full-function native mode) directing frames to the correct Data Link Control link station. A ring station is an instance of a MAC sublayer in a node attached to a ring.

bracket protocol. In SNA, a data flow control protocol in which exchanges between two session partners are achieved through the use of brackets, with one partner designated at session activation as the first speaker and the other as the bidder. The bracket protocol involves bracket initiation and termination rules.

broadcast search. The simultaneous propagation of a search request to all network nodes in an APPN network. This type of search may be used when the location of a resource is unknown to the requester. Contrast with *directed search.*

BSC. Binary synchronous communication.

BTU. Basic transmission unit.

C

cache. An optional part of the directory database in network nodes where frequently used directory information may be stored to speed directory searches.

casual connection. (1) In a subarea network, a connection in which type 5 nodes are attached through the boundary function using low-entry networking (LEN). Therefore, the nodes appear as LEN nodes rather than subarea nodes. (2) In an APPN network, a connection between an end node and a network node with different network identifiers.

CCITT. International Telegraph and Telephone Consultative Committee.

CDS. Central directory server.

central directory. A repository for storing resource location information centrally registered by network nodes or cached as the result of network searches.

central directory server (CDS). A network node that provides a repository for information on network resource locations; it also reduces the number of network searches by providing a focal point for queries and broadcast searches and by caching the results of network searches to avoid later broadcasts for the same information.

central resource registration. A process in which an APPN network node sends information about itself and its client end nodes to a central directory server.

chain. (1) A group of logically linked user data records processed by LU 6.2. (2) A group of request units delimited by begin-chain and end-chain. Responses are always single-unit chains. See *RU chain.*

change management. The process of planning (for example, scheduling) and controlling (for example, distributing, installing, and tracking) changes in an SNA network.

channel. (1) A path along which signals can be sent, for example, data channel, output channel. (A) (2) A functional unit, controlled by the processor, that handles the transfer of data between processor storage and local peripheral equipment.

class of service (COS). A set of characteristics (such as route security, transmission priority, and bandwidth) used to construct a route between session partners. The class of service is derived from a mode name specified by the initiator of a session.

client end node. An end node for which the network node provides network services.

client-server. In communications, the model of interaction in distributed data processing in which a program at one site sends a request to a program at another site and awaits a response. The requesting program is called a client; the answering program is called a server.

cluster. (1) A station that consists of a control unit (a cluster controller) and the terminals attached to it. (2) A group of APPN nodes that have the same network ID and the same topology database. A cluster is a subset of a NETID subnetwork.

CMIP. Common management information protocol.

common management information protocol (CMIP). OSI standard protocol defined in ISO/IEC 9596-1 for the interaction between managers and agents that use the common management information service element (CMISE).

communication controller. A type of communication control unit whose operations are controlled by one or more programs stored and executed in the unit. It manages the details of line control and the routing of data through a network.

composite LEN node. A type 5 node and its subordinate type 4 nodes that support LEN protocols and appear to an attached APPN or LEN node as a single LEN node.

composite network node. A type 5 node and its subordinate type 4 nodes that support APPN network node protocols and appear to an attached APPN or LEN node as a single network node.

configuration. (1) The manner in which the hardware and software of an information processing system are organized and interconnected. (T) (2) The devices and programs that make up a system, subsystem, or network.

configuration services. One of the types of network services in a control point (SSCP, NNCP, or ENCP). Configuration services activates, deactivates, and records the status of physical units, links, and link stations.

congestion. See *network congestion.*

connect phase. An optional phase of link activation during which initial communication is established. It includes "dialing" and "answering" on switched links and can include modem equalization. The connect phase is followed by the optional prenegotiation phase or by the contact phase.

connection network. A representation within an APPN network of a shared-access transport facility (SATF), such as a token ring, that allows nodes identifying their connectivity to the SATF by a common virtual routing node to communicate without having individually defined connections to one another.

contact phase. A phase of link activation during which negotiation-proceeding XID3s are exchanged between the connected link stations to establish the primary and secondary roles of the link stations, the TG number to be used, and other characteristics of the link, and during which the mode-setting command is sent and acknowledged after the primary and the secondary roles are established. Link activation may consist only of the contact phase, or it may also have either a connect phase or a prenegotiation phase or both preceding the contact phase.

contention. In a session, a situation in which both NAUs attempt to initiate the same action at the same time, such as when both attempt to send data in a half-duplex protocol (half-duplex contention), or both attempt to start a bracket (bracket contention). At session initiation, one NAU is defined to be the contention winner; its action will take precedence when contention occurs. The contention loser must get explicit or implicit permission from the contention winner to begin its action.

contention-loser session. To an NAU, a session for which it was defined during session initiation to be the contention loser.

contention-winner session. To an NAU, a session for which it was defined during session initiation to be the contention winner.

control point (CP). (1) A component of an APPN or LEN node that manages the resources of that node. In an APPN node, the CP is capable of engaging in CP-CP sessions with other APPN nodes. In an APPN network node, the CP also provides services to adjacent end nodes in the APPN network. (2) A component of a node that manages resources of that node and optionally provides services to other nodes in the network. Examples are a system services control point (SSCP) in a type 5 subarea node, a network node control point (NNCP) in an APPN network node, and an end node control point (ENCP) in an APPN or LEN end node. An SSCP and an NNCP can provide services to other nodes.

control point management services (CPMS). A component of a control point, consisting of management services function sets, that provides facilities to assist in performing problem management, performance and accounting management, change management, and configuration management. Capabilities provided by the CPMS include sending requests to physical unit management services (PUMS) to test system resources, collecting statistical information (for example, error and performance data) from PUMS about the system resources, and analyzing and presenting test results and statistical information collected about the system resources. Analysis and presentation responsibilities for problem determination and performance monitoring can be distributed among multiple CPMSs.

conversation. A logical connection between two transaction programs using an LU 6.2 session. Conversations are delimited by brackets to gain exclusive use of a session.

conversation-level security. See *session-level security.* See also *end-user verification.*

COS. Class of service.

CP. Control point.

CP capabilities. The level of network services provided by the control point (CP) in an APPN end node or network node. CP capabilities information is exchanged during the activation of CP-CP sessions between two nodes. A node's CP capabilities are encoded in the CP capabilities (X'12C1') GDS variable.

CP-CP sessions. The parallel sessions between two control points, using LU 6.2 protocols and a mode name of CPSVCMG, on which network services requests and replies are exchanged. Each CP of a given pair has one contention-winner session and one contention-loser session with the other.

CP name. A network-qualified name of a control point (CP), consisting of a network ID qualifier identifying the network (or name space) to which the CP's node belongs, and a unique name within the scope of that network ID identifying the CP. Each APPN or LEN node has one CP name, assigned to it at system-definition time.

CPMS. Control point management services.

cross-domain. In SNA, pertaining to control or resources involving more than one domain.

cross-network. In SNA, pertaining to control or resources involving more than one network.

cross-network LU-LU session. In SNA, a session between logical units (LUs) in different networks.

cross-network session. An LU-LU or SSCP-SSCP session whose path traverses more than one SNA network.

cryptography. The transformation of data to conceal its meaning.

Customer Information Control System (CICS). An IBM licensed program that enables transactions entered at remote terminals to be processed concurrently by user-written application programs. It includes facilities for building, using, and maintaining databases.

D

data channel. Synonym for *input/output channel.*

data circuit-terminating equipment (DCE). In a data station, the equipment that provides the signal conversion and coding between the data terminal equipment (DTE) and the line. (I)

Notes:

1. The DCE may be separate equipment or an integral part of the DTE or of the intermediate equipment.
2. A DCE may perform other functions that are usually performed at the network end of the line.

Data Encryption Standard (DES). In computer security, the National Institute of Standards and Technology (NIST) Data Encryption Standard, adopted by the U.S. government as Federal Information Processing Standard (FIPS) Publication 46, which allows only hardware implementations of the data encryption algorithm.

data flow control (DFC). In SNA, a request/response unit (RU) category used for requests and responses exchanged between the data flow control layer in one half-session and the data flow control layer in the session partner.

data flow control (DFC) layer. In SNA, the layer within a half-session that controls whether the half-session can send, receive, or concurrently send and receive, request units (RUs); groups related RUs into RU chains; delimits transactions via the bracket protocol; controls the interlocking of requests and responses in accordance with control modes specified at session activation; generates sequence numbers; and correlates requests and responses.

data link control (DLC). A set of rules used by nodes on a data link (such as an SDLC link or a token ring) to accomplish an orderly exchange of information.

data link control (DLC) layer. In SNA, the layer that consists of the link stations that schedule data transfer over a link between two nodes and perform error control for the link. Examples of data link control are SDLC for serial-by-bit link connection and data link control for the System/370 channel.

Note: The DLC layer is usually independent of the physical transport mechanism and ensures the integrity of data that reaches the higher layers.

data link layer. In the Open Systems Interconnection reference model, the layer that provides services to transfer data between entities in the network layer over a communication link. The data link layer detects and possibly corrects errors that may occur in the physical layer. (T)

data link level. In the hierarchical structure of a data station, the conceptual level of control or processing logic between high level logic and the data link that maintains control of the data link. The data link level performs such functions as inserting transmit bits and deleting receive bits; interpreting address and control fields; generating, transmitting, and interpreting commands and responses; and computing and interpreting

frame check sequences. See also *higher level*, *packet level*, and *physical level*.

data stream. (1) All information (data and control commands) sent over a data link usually in a single read or write operation. (2) A continuous stream of data elements being transmitted, or intended for transmission, in character or binary-digit form, using a defined format.

data terminal equipment (DTE). That part of a data station that serves as a data source, data sink, or both. (I) (A)

DCE. Data circuit-terminating equipment.

decipher. (1) In computer security, to convert ciphertext into plaintext by means of a cipher system. (2) To convert enciphered data into clear data. Contrast with *encipher*.

definite response (DR). In SNA, a protocol requested in the form-of-response-requested field of the request header that directs the receiver of the request to return a response unconditionally, whether positive or negative, to that request chain. Contrast with *exception response* and *no response*.

delayed-request mode. In SNA, an operational mode in which the sender may continue sending request units on the normal flow after sending a definite-response request chain on that flow, without waiting to receive the response to that chain. See also *delayed-response mode*. Contrast with *immediate-request mode*.

delayed-response mode. In SNA, an operational mode in which the receiver of request units can return responses to the sender in a sequence different from that in which the corresponding request units were sent. An exception is the response to a CHASE request, which is returned only after all responses to requests sent before the CHASE have been returned. Contrast with *immediate-response mode*.

dependent LU. See *SSCP-dependent LU*.

dependent LU requester (DLUR). An APPN end node or an APPN network node that owns dependent LUs, but requests that a dependent LU server provide the SSCP services for those dependent LUs. Contrast with *dependent LU server*.

dependent LU server (DLUS). An APPN network node that provides SSCP services for a dependent LU in its own or another APPN network. Contrast with *dependent LU requester*.

DES. Data Encryption Standard.

destination logical unit (DLU). A logical unit that is the target of a Locate search request as part of a session initiation sequence. See also *initiating logical unit (ILU)* and *origin logical unit (OLU)*.

DFC. Data flow control.

directed Locate search. A search request sent to a specific destination node known to contain a resource, such as a logical unit, to verify the continued presence of the resource at the destination node and to obtain the node's connectivity information for route calculation. Contrast with *broadcast search*.

directed search. Synonym for *directed Locate search*.

directory. (1) A database in an APPN node that lists names of resources (in particular, logical units) and records the CP name of the node where each resource is located. See *distributed directory database* and *local directory database*.

directory services (DS). A control point component of an APPN node that maintains knowledge of the location of network resources.

distributed directory database. The complete listing of all the resources in the network as maintained in the individual directories scattered throughout an APPN network. Each node has a piece of the complete directory, but it is not necessary for any one node to have the entire list. Entries are created, modified, and deleted through system definition, operator action, automatic registration, and ongoing network search procedures. Synonymous with *distributed network directory* and *network directory database*.

distributed network directory. Synonym for *distributed directory database*.

DLC. Data link control.

DLU. Destination logical unit.

DLUR. Dependent LU requester.

DLUS. Dependent LU server.

domain. In SNA, see *end node domain*, *network node domain*, and *system services control point domain*.

domain search. A search initiated by a network node to all its authorized client APPN end nodes (that allow themselves to be searched) when it receives a search request for which it has no entry in its database.

downstream. In the direction of data flow from the host to the end user. Contrast with *upstream*.

DS. Directory services.

DTE. Data terminal equipment. (A)

duplex. Pertaining to communication in which data can be sent and received at the same time. Synonymous with *full duplex*. Contrast with *half duplex*.

dynamic. Pertaining to an operation that occurs at the time it is needed rather than at a predetermined or fixed time.

dynamic definition of LUs. The ability of VTAM to dynamically create a definition of an LU upon receiving a session-initiation request from that LU.

dynamic reconfiguration (DR). The process of changing the network configuration (peripheral PUs and LUs) without regenerating complete configuration tables or deactivating the affected major node.

E

element address. A value in the element address field of the network address identifying a specific resource within a subarea. See *subarea address*.

EN. End node.

encipher. To scramble data or to convert data to a secret code that masks the meaning of the data to any unauthorized recipient. Contrast with *decipher*. Synonymous with *encrypt*.

ENCP. End-node control point.

encrypt. Synonym for *encipher*. (T)

encryption. In computer security, the process of transforming data into an unintelligible form in such a way that the original data either cannot be obtained or can be obtained only by using a decryption process.

end node (EN). See *Advanced Peer-to-Peer Networking (APPN) end node* and *low-entry networking (LEN) end node*.

end node domain. An end node control point, its attached links, and its local LUs.

end user. The ultimate source or destination of application data flowing through an SNA network. An end user can be an application program or a workstation operator.

end-user verification. For logical unit (LU) 6.2, checking the identification of end users by means of identifiers and passwords on attach function-management headers (FMHs). See *partner-LU verification*. See also *conversation-level security*.

endpoint node. A node that is at the end of only one branch. Synonymous with *peripheral node*. (T)

Enterprise Systems Connection (ESCON). A set of IBM products and services that provide a dynamically connected environment within an enterprise.

entry point (EP). In SNA, a type 2.0, type 2.1, type 4, or type 5 node that provides distributed network management support. It sends network management data about itself and the resources it controls to a focal point for centralized processing, and it receives and executes focal-point initiated commands to manage and control its resources.

ER. (1) Explicit route. (2) Exception response.

ESCON. Enterprise Systems Connection.

Ethernet. A 10-Mbps baseband local area network that allows multiple stations to access the transmission medium at will without prior coordination, avoids contention by using carrier sense and deference, and resolves contention by using collision detection and transmission. Ethernet uses carrier sense multiple access with collision detection (CSMA/CD).

exception response (ER). In SNA, a protocol requested in the form-of-response-requested field of a request header that directs the receiver to return a response only if the request is unacceptable as received or cannot be processed; that is, a negative response, but not a positive response, can be returned. Contrast with *definite response* and *no response*.

exchange identification (XID). A specific type of basic link unit that is used to convey node and link characteristics between adjacent nodes. XIDs are exchanged between link stations before and during link activation to establish and negotiate link and node characteristics, and after link activation to communicate changes in these characteristics.

explicit route (ER). In SNA, a series of one or more transmission groups that connect two subarea nodes. An explicit route is identified by an origin subarea address, a destination subarea address, an explicit route number, and a reverse explicit route number. Contrast with *virtual route (VR)*.

extended bind. A bind request that includes the Fully Qualified Procedure Correlation Identifier (FQPCID) control vector.

extended border node. A border node that interconnects (a) APPN networks having different network identifiers or (b) separate partitions of the same APPN network, where the partitioning is to allow isolated topology subnetworks (or clusters). An extended border node supports intermediate network routing, allowing it to support LU-LU sessions that do not terminate in its native network. Contrast with *peripheral border node*.

extended network addressing. The network addressing system that splits the address into an 8-bit subarea and a 15-bit element portion. The subarea portion of the address is used to address host processors or communication controllers. The element portion is used to permit processors or controllers to address resources.

extended recovery facility (XRF). A facility that minimizes the effect of failures in MVS, VTAM, the host processor, or high availability applications during sessions between high availability applications and designated terminals. This facility provides an alternate subsystem to take over sessions from the failing subsystem.

extended subarea addressing. A network addressing system that is used in a network with more than 255 subareas.

F

FDDI. Fiber Distributed Data Interface.

Fiber Distributed Data Interface (FDDI). An American National Standards Institute (ANSI) standard for a 100-megabit-per-second LAN using optical fiber cables.

FID. Format identification.

first speaker. See *first-speaker session.*

first-speaker session. The half-session defined at session activation as: (a) able to begin a bracket without requesting permission from the other half-session to do so, and (b) winning contention if both half-sessions attempt to begin a bracket simultaneously. Synonym for *contention-winner session.* Contrast with *bidder session.*

flow control. In SNA, the process of managing the rate at which data traffic passes between components of the network. The purpose of flow control is to optimize the rate of flow of message units with minimum congestion in the network; that is, to neither overflow the buffers at the receiver or at intermediate routing nodes, nor leave the receiver waiting for more message units. See also *adaptive session-level pacing, pacing,* and *session-level pacing.*

FMD. Function management data.

FMH. Function management header.

focal point (FP). (1) See *management services focal point (MSFP).* (2) In the NetView program, the focal point domain is the central host domain. It is the central control point for any management services element containing control of the network management data.

format identification (FID) field. In SNA, a field in each transmission header (TH) that indicates the format of the TH; that is, the presence or absence of certain fields. TH formats differ in accordance with the types of nodes between which they pass. Following are the six FID types:

- FID0, used for traffic involving non-SNA devices between adjacent subarea nodes when either or both nodes do not support explicit route and virtual route protocols
- FID1, used for traffic involving SNA devices between adjacent subarea nodes when either or both nodes do not support explicit route and virtual route protocols
- FID2, used for traffic between a subarea node and an adjacent type 2 peripheral node
- FID3, used for traffic between a subarea node and an adjacent type 1 peripheral node
- FID4, used for traffic between adjacent subarea nodes when both nodes support explicit route and virtual route protocols
- FIDF, used for certain commands (for example, for transmission group control) sent between adjacent subarea nodes when both nodes support explicit route and virtual route protocols.

FQPCID. Fully qualified procedure correlator identifier.

frame relay. (1) An interface standard describing the boundary between a user's equipment and a fast-packet network. In frame-relay systems, flawed frames are discarded; recovery comes end-to-end rather than hop-by-hop.

frame-relay network. A network that consists of frame-relay frame handlers (FRFH) and in which frames are passed from one frame-relay terminal equipment (FRTE) station to another through a series of one or more FRFHs.

full duplex (FDX). Synonym for *duplex.*

fully qualified procedure correlator identifier (FQPCID). A network-unique identifier that is used for the following:

- Correlating messages sent between nodes, such as correlating a Locate search request with its replies
- Identifying a session for problem determination and resolution
- Identifying a session for accounting, auditing, and performance monitoring purposes

This identifier is normally assigned at the node that contains the LU for which a procedure or session is initiated, except when that node is an end node, in which case its network node server may assign it. The

FQPCID consists of a fixed-length correlator concatenated with the network-qualified name of the control point that generated the correlator.

function management data (FMD). An RU category used for end-user data exchanged between logical units (LUs) and for requests and responses exchanged between network services components of LUs, PUs, and control points.

function management header (FMH). One or more headers, optionally present in the leading request units (RUs) of an RU chain, that allow one LU to (a) select a transaction program or device at the session partner and control the way in which the end-user data it sends is handled at the destination, (b) change the destination or the characteristics of the data during the session, and (c) transmit between session partners status or user information about the destination (for example, a program or device). Function management headers can be used with LU type 1, 4, and 6.2 protocols.

G

garbage collection. The process of identifying unused areas of main storage. (A)

gateway. The combination of machines and programs that provide address translation, name translation, and system services control point (SSCP) rerouting between independent SNA networks to allow those networks to communicate. A gateway consists of one gateway NCP and at least one gateway VTAM.

gateway NCP. An NCP that performs address translation to allow cross-network session traffic. The gateway NCP connects two or more independent SNA networks. Synonymous with *gateway node.*

gateway node. Synonym for *gateway NCP.*

gateway SSCP. Synonym for *gateway VTAM.*

gateway VTAM. An SSCP that is capable of cross-network session initiation, termination, takedown, and session outage notification. A gateway VTAM is in session with the gateway NCP; it provides network name translation and assists the gateway NCP in setting up alias network addresses for cross-network sessions. Synonymous with *gateway SSCP.*

GDS. General data stream.

general data stream (GDS). The data stream used for conversations in LU 6.2 sessions.

general data stream (GDS) variable. A type of RU substructure that is preceded by an identifier and a length field and includes either application data, user control data, or SNA-defined control data.

GMFHS. Graphic Monitor Facility host subsystem.

graphic monitor. The graphical user interface of the NetView Graphic Monitor Facility.

Graphic Monitor Facility host subsystem (GMFHS). A NetView feature that manages configuration and status updates for non-SNA resources.

H

half-duplex (HD, HDX). In data communication, pertaining to transmission in only one direction at a time. Contrast with *duplex.* See also *half-duplex operation* and *half-duplex transmission.*

half-duplex operation. A mode of operation of a data link in which data can be transmitted in both directions, one way at a time. (T)

half-duplex transmission. Data transmission in either direction, one direction at a time. (I) (A)

half-session. A session-layer component consisting of the combination of data flow control and transmission control components comprising one end of a session. See also *session connector.*

header. (1) System-defined control information that precedes user data. (2) The portion of a message that contains control information for the message such as one or more destination fields, name of the originating station, input sequence number, character string indicating the type of message, and priority level for the message.

hierarchy. The resource types, display types, and data types that make up the organization, or levels, in a network.

high-performance routing. Addition to APPN that enhances data routing performance and session reliability.

hop. In APPN, a portion of a route that has no intermediate nodes. It consists of only a single transmission group connecting adjacent nodes.

HPR. High-performance routing.

I

ILU. Initiating logical unit.

immediate-request mode. In SNA, an operational mode in which the sender stops sending request units (RUs) on a normal or expedited flow after sending a definite-response request chain on that flow until a

response is made to the chain. Contrast with *delayed-request mode*. See also *immediate-response mode*.

immediate-response mode. In SNA, an operational mode in which the receiver responds to request units (RUs) on a given normal flow in the order it receives them, that is, in a first-in, first-out sequence. Contrast with *delayed-response mode*. See also *immediate-request mode*.

independent LU. See *SSCP-independent LU*.

initiating logical unit (ILU). The logical unit that first requests that a session be initiated. The ILU can be one of the session participants, in which case it is also called the origin logical unit (OLU), or it can be a third-party logical unit that identifies the participants and chooses the OLU.

interchange node. A VTAM node that acts as both an APPN network node and a type 5 subarea node to transform APPN protocols to subarea protocols and vice versa. Contrast with *migration data host*.

interconnected networks. SNA networks connected by gateways.

interconnection. See *SNA network interconnection (SNI)*.

intermediate network node. In APPN, a node that is part of a route between an origin LU (OLU) and a destination LU (DLU) but neither contains the OLU or the DLU nor serves as the network server for either the OLU or DLU.

intermediate node. A node that is at the end of more than one branch. (T)

intermediate routing function (IRF). A capability within a node that allows it to receive and route path information units (PIUs) that neither originate from, nor are destined for, network accessible units (NAUs) in that node.

intermediate routing network. See *APPN intermediate routing network*.

intermediate routing node (IRN). A node containing intermediate routing function.

intermediate session routing (ISR). A type of routing function within an APPN network node that provides session-level flow control and outage reporting for all sessions that pass through the node but whose end points are elsewhere.

International Organization for Standardization (ISO). An organization of national standards bodies from various countries established to promote development of standards to facilitate international exchange of goods and services, and develop cooperation in intellectual, scientific, technological, and economic activity.

Internode routing. The capability of path control to route PIUs from half-sessions to data link control and from data link control to half-sessions for sessions between NAUs that reside in different nodes.

intranode routing. The capability of path control to route PIUs for sessions between NAUs that reside in the same node.

IPM. Isolated pacing message.

IPR. Isolated pacing response.

ISO. International Organization for Standardization.

ISR. Intermediate session routing.

L

LAN. Local area network.

layer. In SNA, a grouping of related functions that are logically separate from the functions in other groups. Implementation of the functions in one layer can be changed without affecting functions in other layers.

LEN. Low-entry networking.

LEN connection. A link over which LEN protocols are used.

LEN end node. See *low-entry networking (LEN) end node*.

LEN node. A node that supports independent LU protocols but does not support CP-CP sessions. It may be a peripheral node attached to a boundary node in a subarea network, an end node attached to an APPN network node in an APPN network, or a peer-connected node directly attached to another LEN node or APPN end node. See also *low-entry networking (LEN) end node*.

LFSID. Local-form session identifier.

limited resource. A connection facility that causes a session traversing it to be terminated if no session activity is detected for a specified period of time. See also *limited-resource session*.

limited-resource session. A session that traverses a limited-resource link. This session is terminated if no session activity is detected for a specified period of time.

link. The combination of the link connection (the transmission medium) and two link stations, one at each end

of the link connection. A link connection can be shared among multiple links in a multipoint or token-ring configuration.

link connection. The physical equipment providing two-way communication between one link station and one or more other link stations; for example, a telecommunication line and data circuit-terminating equipment (DCE). Synonymous with *data circuit*.

link station. The hardware and software components within a node representing a connection to an adjacent node over a specific link. For example, if node A is the primary end of a multipoint line that connects to three adjacent nodes, node A will have three link stations representing the connections to the adjacent nodes. See also *adjacent link station*.

LLC. Logical link control.

local address. In SNA, an address used in a peripheral node in place of a network address and transformed to or from a network address by the boundary function in a subarea node.

local area network (LAN). (1) A computer network located on a user's premises within a limited geographical area. Communication within a local area network is not subject to external regulations; however, communication across the LAN boundary may be subject to some form of regulation. (T) (2) A network in which a set of devices are connected to one another for communication and that can be connected to a larger network. See also *Ethernet* and *token ring*. (3) Contrast with *metropolitan area network (MAN)* and *wide area network (WAN)*.

local directory database. That set of resources (LUs) in the network known at a particular node. The resources included are all those in the node's domain as well as any cache entries.

local-form session identifier (LFSID). A dynamically assigned value used at a type 2.1 node to identify traffic for a particular session using a given transmission group (TG). The LFSID is encoded in the ODAI, OAF', and DAF' fields of the transmission headers that accompany session messages exchanged over the TG.

local topology database. A database in an APPN or LEN node containing an entry for each transmission group (TG) having at least one end node for an endpoint. In an end node, the database has one entry for each TG connecting to the node. In a network node, the database has an entry for each TG connecting the network node to an end node. Each entry describes the current characteristics of the TG that it represents. A network node has both a local and a network topology database while an end node has only a local topology database.

Locate. Synonym for *Locate/CD-Initiate*.

Locate chain. A temporary logical connection, spanning a series of CP-CP sessions, between the control point at a search initiator and the control point at the search destination. It is set up when a search is initiated, and ends on completion of the search. It is used to transport directory search control traffic and allows route-outage reporting to the search endpoints if an outage occurs during the search.

Locate search. The means directory services in a node uses to find a resource that is not in that node. The Locate search enables directory services to ask the directory services components in other APPN nodes for information on the target resource. See also *broadcast search* and *directed Locate search*.

Locate search reply. A Locate, a Found Resource, and a Cross-Domain Initiate GDS variable used when a network resource has been located.

Locate search request. A Locate, a Find Resource, and a Cross-Domain Initiate GDS variable used to search for a network resource.

Locate/CD-Initiate. An abbreviated term for a message exchanged between APPN nodes that contains one of the following sets of general data stream (GDS) variables:

- A Locate, a Find Resource, and a Cross-Domain Initiate GDS variable used for a network search request
- A Locate, a Found Resource, and a Cross-Domain Initiate GDS variable used for a search reply when a network resource has been located

These message structures correspond to the CP components that perform the search of the distributed network directory and establish the session. The Locate GDS variable contains information used to control the delivery of the search messages in the network. The Find and Found GDS variables contain information used in the directories: origin cache data (control point information) and search arguments (destination LU name), and located resource information, respectively. The Cross-Domain Initiate GDS variable contains endpoint TG vector information to be used in selecting the route for the session. The length of the Locate/CD-Initiate message is limited to 1024 bytes.

logical link control (LLC). The data link control (DLC) LAN sublayer that provides two types of DLC operation for the orderly exchange of information. The first type is connectionless service, which allows information to be sent and received without establishing a link. The LLC sublayer does not perform error recovery or flow control for connectionless service. The second type is connection-oriented service, which requires establishing a link prior to the exchange of information. Connection-

oriented service provides sequenced information transfer, flow control, and error recovery.

logical unit (LU). A type of network accessible unit that enables end users to gain access to network resources and communicate with each other.

logical unit (LU) 6.2. A type of logical unit that supports general communication between programs in a distributed processing environment. LU 6.2 is characterized by (a) a peer relationship between session partners, (b) efficient utilization of a session for multiple transactions, (c) comprehensive end-to-end error processing, and (d) a generic application program interface (API) consisting of structured verbs that are mapped into a product implementation.

logical unit of work (LUW). The changes to protected resources that are committed or backed out as a unit as a result of a sync point operation. The protected resources may be distributed among different LUs joined together by conversations.

low-entry networking (LEN). A capability of nodes to attach directly to one another using basic peer-to-peer protocols to support multiple and parallel sessions between logical units.

low-entry networking (LEN) end node. A LEN node receiving network services from an adjacent APPN network node.

low-entry networking (LEN) node. A node that provides a range of end-user services, attaches directly to other nodes using peer protocols, and derives network services implicitly from an adjacent APPN network node, that is, without the direct use of CP-CP sessions.

LU. Logical unit.

LU-LU session. A logical connection between two logical units (LUs) in an SNA network that typically provides communication between two end users.

LU type. The classification of an LU in terms of the specific subset of SNA protocols and options it supports for a given session, namely:

- The mandatory and optional values allowed in the session activation request
- The usage of data stream controls, function management headers (FMHs), request unit parameters, and sense data values
- Presentation services protocols such as those associated with FMH usage

LU types 0, 1, 2, 3, 4, 6.1, 6.2, and 7 are defined.

LU type 6.2 (LU 6.2). A type of logical unit that supports general communication between programs in a distributed processing environment. LU 6.2 is characterized by (a) a peer relationship between session partners, (b) efficient utilization of a session for multiple transactions, (c) comprehensive end-to-end error processing, and (d) a generic application program interface consisting of structured verbs that are mapped into a product implementation.

LU 6.2. Logical unit 6.2.

LU 6.2 session. A session that is initiated by VTAM on behalf of a logical unit (LU) 6.2 application program, or a session initiated by a remote LU in which the application program specifies that VTAM is to control the session by using the APPCCMD macroinstruction.

LU 6.2 verb. A syntactical unit in the LU 6.2 application program interface representing an operation.

LUW. Logical unit of work.

M

MAC. Medium access control.

MAN. Metropolitan area network.

management services (MS). (1) One of the types of network services in control points (CPs) and physical units (PUs). Management services are the services provided to assist in the management of SNA networks, such as problem management, performance and accounting management, configuration management, and change management. (2) Services that assist in the management of systems and networks in areas such as problem management, performance management, business management, operations management, configuration management, and change management.

management services focal point (MSFP). For any given management services discipline (for example, problem determination or response time monitoring), the control point that is responsible for that type of network management data for a sphere of control. This responsibility may include collecting, storing or displaying the data or all of these. (For example, a problem determination focal point is a control point that collects, stores, and displays problem determination data.)

mapped conversation. An LU 6.2 conversation type specified by the allocating transaction program. Transaction programs using a mapped conversation can exchange messages of arbitrary format regardless of the underlying data stream. System-defined or user-defined mappers can perform data transformation for the transaction programs. See also *conversation*. Contrast with *basic conversation*.

Mbps. One million bits per second.

MDS. Multiple-domain support.

medium access control (MAC). In LANs, the sublayer of the data link control layer that supports medium-dependent functions and uses the services of the physical layer to provide services to the logical link control (LLC) sublayer. The MAC sublayer includes the method of determining when a device has access to the transmission medium.

medium access control (MAC) sublayer. In a local area network, the part of the data link layer that applies a medium access method. The MAC sublayer supports topology-dependent functions and uses the services of the physical layer to provide services to the logical link control sublayer. (T)

message unit. In SNA, the unit of data processed by any layer; for example, a basic information unit (BIU), a path information unit (PIU), or a request/response unit (RU).

metropolitan area network (MAN). A network formed by the interconnection of two or more networks which may operate at higher speed than those networks, may cross administrative boundaries, and may use multiple access methods. (T) Contrast with *local area network (LAN)* and *wide area network (WAN).*

migration. The installation of a new version or release of a program to replace an earlier version or release.

migration data host. A VTAM node that acts as both an APPN end node and a type 5 subarea node. Contrast with *interchange node.*

MLTG. Multilink transmission group.

mode name. The name used by the initiator of a session to designate the characteristics desired for the session, such as traffic pacing values, message-length limits, sync point and cryptography options, and the class of service within the transport network.

MS. Management services.

MSFP. Management services focal point.

MU. Message unit.

multilink transmission group (MLTG). See *transmission group (TG).*

multipath channel (MPC). A channel protocol that uses multiple unidirectional subchannels for VTAM-to-VTAM bidirectional communication.

multiple-domain network. (1) A network with more than one system services control point. (2) An APPN network with more than one network node.

multiple-domain support (MDS). A technique for transporting management services data between management services function sets over LU-LU and CP-CP sessions. See also *multiple-domain support message unit (MDS-MU).*

Multiple Virtual Storage (MVS). See *MVS.*

MVS. Multiple Virtual Storage. Implies MVS/370, the MVS/XA product, and the MVS/ESA product.

N

native network. The subnetwork whose network identifier a node uses for its own network-qualified resource names.

NAU. (1) Network accessible unit. (2) Network addressable unit.

navigate. In the NetView Graphic Monitor Facility, to move between levels in the view hierarchy.

NC. Network control.

NCP. Network Control Program.

negotiable BIND. In SNA, a capability that allows two half-sessions to negotiate the parameters of a session when the session is being activated.

NetView Graphic Monitor Facility (NGMF). A function of the NetView program that provides the network operator with a graphic topological presentation of a network controlled by the NetView program and that allows the operator to manage the network interactively.

network. (1) An arrangement of nodes and connecting branches. (T) (2) A configuration of data processing devices and software connected for information interchange. (3) A group of nodes and the links interconnecting them.

network accessible unit (NAU). A logical unit (LU), physical unit (PU), control point (CP), or system services control point (SSCP). It is the origin or the destination of information transmitted by the path control network. Synonymous with *network addressable unit.*

network address. (1) In a subarea network, an address, consisting of subarea and element fields, that identifies a link, link station, physical unit, logical unit, or system services control point. Subarea nodes use network addresses; peripheral nodes use local addresses or local-form session identifiers (LFSIDs). The boundary function in the subarea node to which a peripheral node is attached transforms local addresses or LFSIDs to network addresses and vice versa. Contrast with *network name.*

network addressable unit (NAU). Synonym for *network accessible unit.*

network congestion. An undesirable overload condition caused by traffic in excess of what a network can handle.

network control (NC). In SNA, a request/response unit (RU) category used for requests and responses exchanged between physical units (PUs) for such purposes as activating and deactivating explicit and virtual routes and sending load modules to adjust peripheral nodes. See also *data flow control*, *function management data*, and *session control*.

network control program. A program, generated by the user from a library of IBM-supplied modules, that controls the operation of a communication controller.

Network Control Program (NCP). An IBM licensed program that provides communication controller support for single-domain, multiple-domain, and interconnected network capability.

network directory database. Synonym for *distributed directory database*.

network identifier. A 1- to 8-byte customer-selected name or an 8-byte IBM-registered name that uniquely identifies a specific subnetwork.

network layer. In Open Systems Interconnection (OSI) architecture, the layer that is responsible for routing, switching, and link-layer access across the OSI environment.

network layer packet. In HPR, the message unit used to carry data through the network.

network management vector transport (NMVT). A management services request/response unit (RU) that flows over an active session between physical unit management services and control point management services (SSCP-PU session).

network name. The symbolic identifier by which end users refer to a network accessible unit, a link, or a link station within a given subnetwork. In APPN networks, network names are also used for routing purposes. Contrast with *network address*.

network node (NN). Synonym for *Advanced Peer-to-Peer Networking (APPN) network node*.

network-node domain. An APPN network-node control point, its attached links, the network resources for which it answers directory search requests (namely, its local LUs and adjacent LEN end nodes), the adjacent APPN end nodes with which it exchanges directory search requests and replies, and other resources (such as a local storage device) associated with its own node or an adjacent end node for which it provides management services.

network node server. An APPN network node that provides network services for its local LUs and client end nodes.

network operator. A person who controls the operation of all or part of a network.

network-qualified name. A name that uniquely identifies a specific resource within a specific network. It consists of a network identifier and a resource name, each of which is a 1- to 8-byte symbol string.

network services. (1) The services within network accessible units that control network operation through SSCP-SSCP, SSCP-PU, SSCP-LU, and CP-CP sessions. (2) The session services (directory and route-selection functions) and management services provided by an APPN network-node control point to its domain.

network topology database. The representation of the current connectivity between the network nodes within an APPN network. It includes (a) entries for all network nodes and the transmission groups interconnecting them and (b) entries for all virtual routing nodes to which network nodes are attached.

NGMF. NetView Graphic Monitor Facility.

NLP. Network layer packet.

NMVT. Network management vector transport.

NN. Network node.

NNCP. Network node control point.

no response. In SNA, a protocol requested in the form-of-response-requested field of the request header that directs the receiver of the request not to return any response, regardless of whether or not the request is received and processed successfully. Contrast with *definite response* and *exception response*.

node. (1) Any device, attached to a network, that transmits and receives data. (2) An endpoint of a link or a junction common to two or more links in a network. Nodes can be processors, communication controllers, cluster controllers, or terminals. Nodes can vary in routing and other functional capabilities.

node type. A designation of a node according to the protocols it supports or the role it plays in a network. Node type was originally denoted numerically (as 1, 2.0, 2.1, 4, and 5) but is now characterized more specifically by protocol type (APPN network node, LEN node, subarea node, and interchange node, for example) because type 2.1 nodes and type 5 nodes support multiple protocol types and roles.

nonnative network. (1) A subnetwork whose network identifier differs from the network identifier that a node

uses for its own network-qualified resource names. (2) Any network attached to a gateway NCP that does not contain that NCP's resources.

O

ODAI. Origin-Destination Assignor indicator, a bit in a FID2 transmission header used to divide the address space so that an address space manager (ASM) in one node may use all possible combinations of OAF', DAF' with the ODAI having one setting and the ASM in the adjacent node may use all possible combinations of OAF', DAF' with the ODAI having the complementary setting.

OLU. Origin logical unit.

one-way bracket. A bracket in which data is sent from one NAU to another in a single chain with begin bracket, conditional end bracket, and exception response requested. When one-way brackets are used on CP-CP sessions, they are always sent on the contention-winner session.

one-way conversation. A conversation in which data is sent from one transaction program (the source) to another (the target) with no response requested and that is released after the data is sent. If the source TP terminates as soon as it releases the conversation, the data may still be in transit; thus, the source and target TPs are not necessarily active at the same time.

Open Systems Interconnection (OSI). (1) The interconnection of open systems in accordance with standards of the International Organization for Standardization (ISO) for the exchange of information. (T) (A) (2) The use of standardized procedures to enable the interconnection of data processing systems.

origin logical unit (OLU). A logical unit that is the source of a Locate search request as part of a session initiation sequence. See also *destination logical unit (DLU)* and *initiating logical unit (ILU)*.

OSI. Open Systems Interconnection.

P

pacing. A technique by which a receiving component controls the rate of transmission of a sending component to prevent overrun or congestion. See *session-level pacing*, *send pacing*, and *virtual route (VR) pacing*. See also *flow control*.

pacing group. Synonym for *pacing window*.

pacing window. (1) The path information units (PIUs) that can be transmitted on a virtual route before a virtual-route pacing response is received, indicating that the virtual route receiver is ready to receive more PIUs on the route. (2) The requests that can be transmitted on the normal flow in one direction on a session before a session-level pacing response is received, indicating that the receiver is ready to accept the next group of requests. (3) Synonymous with *pacing group*.

parallel links. In SNA, two or more links between adjacent subarea nodes.

parallel sessions. Two or more concurrently active sessions between the same two network accessible units (NAUs) using different pairs of network addresses or local-form session identifiers. Each session can have independent session parameters.

parallel transmission groups. Multiple transmission groups between adjacent nodes, with each group having a distinct transmission group number.

path. (1) In a network, any route between any two nodes. A path may include more than one branch. (T) (2) The series of transport network components (path control and data link control) that are traversed by the information exchanged between two network accessible units. See also *explicit route (ER)*, *route extension*, and *virtual route (VR)*.

path control (PC). The function that routes message units between network accessible units in the network and provides the paths between them. It converts the basic information units (BIUs) from transmission control (possibly segmenting them) into path information units (PIUs) and exchanges basic transmission units containing one or more PIUs with data link control. Path control differs by node type: some nodes (APPN nodes, for example) use locally generated session identifiers for routing, and others (subarea nodes) use network addresses for routing.

path control network. Synonym for *transport network*.

path information unit (PIU). A message unit consisting of a transmission header (TH) alone, or a TH followed by a basic information unit (BIU) or a BIU segment. See also *transmission header*.

PCID. Procedure-correlation identifier.

peer. In network architecture, any functional unit that is in the same layer as another entity. (T)

peripheral border node. A border node that interconnects adjacent APPN networks having different network identifiers in order to support LU-LU sessions that have one partner LU in its native network. Contrast with *extended border node*.

peripheral link. In SNA, a link between a subarea and a peripheral node. See also *route extension (REX)*.

peripheral logical unit (LU). In SNA, a logical unit in a peripheral node.

peripheral LU. Peripheral logical unit.

peripheral node. A node that uses local addresses for routing and therefore is not affected by changes in network addresses. A peripheral node requires boundary-function assistance from an adjacent subarea node. A peripheral node can be a type 1, 2.0, or 2.1 node connected to a subarea boundary node.

peripheral PU. In SNA, a physical unit (PU) in a peripheral node.

physical unit (PU). The component that manages and monitors the resources (such as attached links and adjacent link stations) associated with a node, as requested by an SSCP via an SSCP-PU session. An SSCP activates a session with the physical unit in order to indirectly manage, through the PU, resources of the node such as attached links. This term applies to type 2.0, type 4, and type 5 nodes only. See also *peripheral PU* and *subarea PU*.

PIU. Path information unit.

PLU. Primary logical unit.

prenegotiation phase. An optional phase of link activation that occurs after physical connection of the link has been established. During this phase, polling may occur to determine if the adjacent link station is active, and prenegotiation XID3s are exchanged to allow each node to verify the identity of the adjacent node by examining the CP name in the Network Name control vector appended to the XID3 or the Node Identification field in the fixed part of the XID3. See also *connect phase* and *contact phase*.

primary logical unit (PLU). The logical unit (LU) that sends the BIND to activate a session with its partner LU. Contrast with *secondary logical unit*.

problem determination. The process of determining the source of a problem; for example, a program component, machine failure, telecommunication facilities, user or contractor-installed programs or equipment, environmental failure such as a power loss, or user error.

procedure-correlation identifier (PCID). In SNA, a value used to correlate all requests and replies associated with a given procedure.

protocol. In SNA, the meanings of, and the sequencing rules for, requests and responses used for managing the network, transferring data, and synchronizing the states of network components. Synonymous with *line control discipline* and *line discipline*. See *bracket protocol* and *link protocol*.

protocol boundary. The signals and rules governing interactions between two components within a node.

R

rapid-transport protocol. In HPR, used to establish the RTP connection and to transport APPN data over this connection in such a way that intermediate nodes are not aware of the SNA sessions, or of the RTP connection itself.

request header (RH). The control information that precedes a request unit (RU). See also *request/response header (RH)*.

request/response header (RH). Control information associated with a particular RU. The RH precedes the request/response unit (RU) and specifies the type of RU (request unit or response unit).

request/response unit (RU). A generic term for a request unit or a response unit. See *request unit (RU)* and *response unit (RU)*.

request unit (RU). A message unit that contains control information, end-user data, or both.

resource hierarchy. In VTAM, the relationship among network resources in which some resources are subordinate to others as a result of their position in the network structure and architecture; for example, the logical units (LUs) of a peripheral physical unit (PU) are subordinate to that PU, which, in turn, is subordinate to the link attaching it to its subarea node.

Resource Object Data Manager (RODM). A component of the NetView program that operates as a cache manager and that supports automation applications. RODM provides an in-memory cache for maintaining real-time data in an address space that is accessible by multiple applications.

resource registration. The process of identifying names of resources, such as LUs, to a network node server or a central directory server.

resource sequence number (RSN). A value that identifies an update of a resource in a network topology database.

response header (RH). A header, optionally followed by a response unit (RU), that indicates whether the response is positive or negative and that may contain a pacing response. See also *negative response*, *pacing response*, and *positive response*.

response unit (RU). A message unit that acknowledges a request unit. It may contain prefix information received in a request unit. If positive, the response unit

may contain additional information (such as session parameters in response to BIND SESSION). If negative, the response unit contains sense data defining the exception condition.

resync. Recovery processing that is performed by sync point services when a failure of a session, transaction program, or LU occurs during sync point processing. The purpose of resync is to return protected resources to consistent states.

RH. Request/response header.

ring station. The functions necessary for connecting to the local area network and for operating with the token-ring protocols. These include token handling, transferring copied frames from the ring to the using node's storage, maintaining error counters, observing Medium Access Control (MAC) sublayer protocols (for address acquisition, error reporting, or other duties), and (in the full-function native mode) directing frames to the correct Data Link Control link station. A ring station is an instance of a MAC sublayer in a node attached to a ring.

RODM. Resource Object Data Manager.

route. (1) An ordered sequence of nodes and transmission groups (TGs) that represent a path from an origin node to a destination node traversed by the traffic exchanged between them. (2) The path that network traffic uses to get from source to destination.

Route Selection control vector (RSCV). A control vector that describes a route within an APPN network. The RSCV consists of an ordered sequence of control vectors that identify the TGs and nodes that make up the path from an origin node to a destination node.

route selection services (RSS). A subcomponent of the topology and routing services component that determines the preferred route between a specified pair of nodes for a given class of service.

RSCV. Route Selection control vector.

RSN. Resource sequence number.

RSS. Route selection services.

RTP. Rapid-transport protocol.

RU. Request/response unit.

S

SAA. Systems Application Architecture.

same-domain. Pertaining to communication between entities in the same SNA domain. Contrast with *cross-domain*. See also *single-domain network*.

same-domain LU-LU session. In SNA, an LU-LU session between logical units (LUs) in the same domain. Contrast with *cross-domain LU-LU session*.

SATF. Shared-access transport facility.

SDLC. Synchronous Data Link Control.

secondary logical unit (SLU). In SNA, the logical unit (LU) that contains the secondary half-session for a particular LU-LU session. An LU may contain secondary and primary half-sessions for different active LU-LU sessions. Contrast with *primary logical unit (PLU)*.

segment. (1) Synonym for *BIU segment*.

segmentation. A process by which path control (PC) divides basic information units (BIUs) into smaller units, called BIU segments, to accommodate smaller buffer sizes in adjacent nodes. Both segmentation and segment assembly are optional PC features. The support for either or both is indicated in the BIND request and response.

session. (1) In network architecture, for the purpose of data communication between functional units, all the activities which take place during the establishment, maintenance, and release of the connection. (T) (2) A logical connection between two network accessible units (NAUs) that can be activated, tailored to provide various protocols, and deactivated, as requested. Each session is uniquely identified in a transmission header (TH) accompanying any transmissions exchanged during the session.

session connector. A session-layer component in an APPN network node or in a subarea node boundary or gateway function that connects two stages of a session. Session connectors swap addresses from one address space to another for session-level intermediate routing, segment session message units as needed, and (except for gateway function session connectors) adaptively pace the session traffic in each direction. See also *half-session*.

session cryptography key. In SNA, a data encrypting key used to encipher and decipher function management data (FMD) requests transmitted in an LU-LU session that uses cryptography.

session-level pacing. A flow control technique that permits a receiving half-session or session connector to

control the data transfer rate (the rate at which it receives request units) on the normal flow. It is used to prevent overloading a receiver with unprocessed requests when the sender can generate requests faster than the receiver can process them. See *pacing* and *virtual route pacing*.

session-level security. For logical unit (LU) 6.2, partner LU verification and session cryptography. See *conversation-level security*.

session limit. The maximum number of concurrently active LU-LU sessions that a particular logical unit (LU) can support.

session monitor. The component of the NetView program that collects and correlates session-related data and provides online access to this information.

session path. The half-sessions delimiting a given session and their interconnection (including any intermediate session connectors).

session services. One of the types of network services in the control point (CP) and in the logical unit (LU). These services provide facilities for an LU or a network operator to request that a control point (an ENCP, NNCP, or SSCP) assist with initiating or terminating sessions between logical units. Assistance with session termination is needed only by SSCP-dependent LUs. See *configuration services*, *maintenance services*, and *management services*.

session stage. The portion of a session path consisting of two session-layer components that are logically adjacent (no other session-layer components between them), and their interconnection. An example is the paired session-layer components in adjacent type 2.1 nodes and their interconnection over the link between them. Examples include paired BF session connectors and their interconnection over a virtual route, and the paired session-layer components in adjacent type 2.1 nodes and their interconnection over the link between them. A session path may consist of one stage, as between LUs in two physically adjacent nodes; two stages, as in a session having one intermediate boundary function; or more than two stages, as in an APPN network where the number of stages equals one less than the number of nodes in the path.

share limit. In SNA, the maximum number of control points that can control a network resource concurrently.

shared-access transport facility (SATF). A transmission facility, such as a multipoint link connection, a public switched network, or a token-ring, on which multiple pairs of nodes can form concurrently active links.

single-domain network. In SNA, a network with one system services control point (SSCP). Contrast with *multiple-domain network*.

SLU. Secondary logical unit.

SNA. Systems Network Architecture.

SNA Management Services (SNA/MS). The services provided to assist in management of SNA networks.

SNA network. The part of a user-application network that conforms to the formats and protocols of Systems Network Architecture. It enables reliable transfer of data among end users and provides protocols for controlling the resources of various network configurations. The SNA network consists of network accessible units (NAUs), boundary function, gateway function, and intermediate session routing function components; and the transport network.

SNA network interconnection (SNI). The connection, by gateways, of two or more independent SNA networks to allow communication between logical units in those networks. The individual SNA networks retain their independence.

SNA/MS. SNA Management Services.

SNI. SNA network interconnection.

SOC. Sphere of control.

source routing. In LANs, a method by which the sending station determines the route the frame will follow and includes the routing information with the frame. Bridges then read the routing information to determine whether they should forward the frame.

sphere of control (SOC). The set of control point domains served by a single management services focal point.

sphere of control (SOC) node. A node directly in the sphere of control of a focal point. A SOC node has exchanged management services capabilities with its focal point. An APPN end node can be a SOC node if it supports the function to exchange management services capabilities.

SPM. Sync point manager.

SS. Session services.

SSCP. System services control point.

SSCP-dependent LU. An LU that requires assistance from a system services control point (SSCP) in order to initiate an LU-LU session. It requires an SSCP-LU session.

SSCP-independent LU. An LU that is able to activate an LU-LU session (that is, send a BIND request) without assistance from an SSCP. It does not have an

SSCP-LU session. Currently, only an LU 6.2 can be an independent LU.

SSCP-LU session. In SNA, a session between a system services control point (SSCP) and a logical unit (LU). The session enables the LU to request the SSCP to help initiate LU-LU sessions.

SSCP-PU session. In SNA, a session between a system services control point (SSCP) and a physical unit (PU); SSCP-PU sessions allow SSCPs to send requests to and receive status information from individual nodes in order to control the network configuration.

SSCP-SSCP session. In SNA, a session between the system services control point (SSCP) in one domain and the SSCP in another domain. An SSCP-SSCP session is used to initiate and terminate cross-domain LU-LU sessions.

SSCP takeover. See *resource takeover*.

subarea. A portion of the SNA network consisting of a subarea node, attached peripheral nodes, and associated resources. Within a subarea node, all network accessible units (NAUs), links, and adjacent link stations (in attached peripheral or subarea nodes) that are addressable within the subarea share a common subarea address and have distinct element addresses.

subarea address. A value in the subarea field of the network address that identifies a particular subarea. See also *element address*.

subarea host node. A host node that provides both subarea function and an application program interface (API) for running application programs. It provides system services control point (SSCP) functions and subarea node services, and it is aware of the network configuration. See *boundary node*, *node*, *peripheral node*, and *subarea node*. See also *boundary function* and *node type*.

subarea LU. A logical unit that resides in a subarea node. Contrast with *peripheral LU*.

subarea network. Interconnected subareas, their directly attached peripheral nodes, and the transmission groups that connect them.

subarea node (SN). A node that uses network addresses for routing and maintains routing tables that reflect the configuration of the network. Subarea nodes can provide gateway function to connect multiple subarea networks, intermediate routing function, and boundary function support for peripheral nodes. Type 4 and type 5 nodes can be subarea nodes.

subarea path control. The function in a subarea node that routes message units between network accessible units (NAUs) and provides the paths between them. See *path control*. See also *boundary function*, *peripheral node*, and *subarea node*.

subarea PU. In SNA, a physical unit (PU) in a subarea node.

subnet. Synonym for *subnetwork*.

subnetwork. Any group of nodes that have a set of common characteristics, such as the same network ID. Synonymous with *subnet*.

sync point. An intermediate or end point during processing of a transaction at which an update or modification to one or more of the transaction's protected resources is logically complete and error free. Synonymous with *synchronization point*.

sync point manager (SPM). The component of the node that implements two-phase commit and resynchronization processing. The subcomponents of the SPM are sync point services (SPS) and the protection managers (the conversation resource protection managers and the local resource protection managers).

sync point services (SPS). The component of the sync point manager that is responsible for coordinating the managers of protected resources during sync point processing. SPS coordinates two-phase commit protocols, resync protocols, and logging.

synchronization point. Synonym for *sync point*.

synchronous. Pertaining to two or more processes that depend upon the occurrence of specific events such as common timing signals. (T)

Synchronous Data Link Control (SDLC). A discipline conforming to subsets of the Advanced Data Communication Control Procedures (ADCCP) of the American National Standards Institute (ANSI) and High-level Data Link Control (HDLC) of the International Organization for Standardization, for managing synchronous, code-transparent, serial-by-bit information transfer over a link connection. Transmission exchanges may be duplex or half-duplex over switched or nonswitched links. The configuration of the link connection may be point-to-point, multipoint, or loop. (I) Contrast with *binary synchronous communication (BSC)*.

system definition. The process, completed before a system is put into use, by which desired functions and operations of the system are selected from various available options. Synonymous with *system generation*.

system generation. Synonym for *system definition*.

system management facility (SMF). A standard feature of MVS that collects and records a variety of system and job-related information.

system services control point (SSCP). A component within a subarea network for managing the configuration, coordinating network operator and problem determination requests, and providing directory services and other session services for end users of the network. Multiple SSCPs, cooperating as peers with one another, can divide the network into domains of control, with each SSCP having a hierarchical control relationship to the physical units and logical units within its own domain.

system services control point (SSCP) domain. The system services control point, the physical units (PUs), the logical units (LUs), the links, the link stations, and all the resources that the SSCP has the ability to control by means of activation and deactivation requests.

Systems Application Architecture (SAA) solution. A set of IBM software interfaces, conventions, and protocols that provide a framework for designing and developing applications that are consistent across systems.

Systems Network Architecture (SNA). The description of the logical structure, formats, protocols, and operational sequences for transmitting information units through, and controlling the configuration and operation of, networks. The layered structure of SNA allows the ultimate origins and destinations of information, that is, the end users, to be independent of and unaffected by the specific SNA network services and facilities used for information exchange.

T

takeover. The process by which the failing active subsystem is released from its extended recovery facility (XRF) sessions with terminal users and replaced by an alternate subsystem. See *resource takeover.*

TDU. Topology database update.

TG. Transmission group.

TG vector. See *transmission group vector.*

TH. Transmission header.

token. (1) In a local area network, the symbol of authority passed successively from one data station to another to indicate the station temporarily in control of the transmission medium. (2) In LANs, a sequence of bits passed from one device to another along the transmission medium. When the token has data appended to it, it becomes a frame.

token ring. (1) According to IEEE 802.5, network technology that controls media access by passing a token (special packet or frame) between media-attached stations. (2) A FDDI or IEEE 802.5 network with a ring topology that passes tokens from one attaching ring station (node) to another. (3) See also *local area network (LAN).*

token-ring network. (1) A ring network that allows unidirectional data transmission between data stations, by a token passing procedure, such that the transmitted data return to the transmitting station. (T) (2) A network that uses a ring topology, in which tokens are passed in a circuit from node to node. A node that is ready to send can capture the token and insert data for transmission.

topology. In communications, the physical or logical arrangement of nodes in a network, especially the relationships among nodes and the links between them.

topology and routing services (TRS). An APPN control point component that manages the topology database, computes routes, and provides a Route Selection control vector (RSCV) that specifies the best route through the network for a given session based on its requested class of service.

topology database. See *local topology database* and *network topology database.*

topology database update (TDU). A message about a new or changed link or node that is broadcast among APPN network nodes to maintain the network topology database, which is fully replicated in each network node. A TDU contains information that identifies the following:

- The sending node
- The node and link characteristics of various resources in the network
- The sequence number of the most recent update for each of the resources described.

topology subnetwork. A group of APPN nodes that share a common topology database.

TP. Transaction program.

transaction program. A program that processes transactions in an SNA network. There are two kinds of transaction programs: application transaction programs and service transaction programs. See also *conversation.*

transmission group (TG). (1) A connection between adjacent nodes that is identified by a transmission group number. See also *parallel transmission groups.* (2) In a subarea network, a single link or a group of links between adjacent nodes. When a transmission group consists of a group of links, the links are viewed as a single logical link, and the transmission group is called a *multilink transmission group (MLTG).* (3) In an APPN network, a single link between adjacent nodes.

transmission group (TG) vector. A representation of an endpoint TG in a T2.1 network, consisting of two control vectors: the TG Descriptor (X'46') control vector and the TG Characteristics (X'47') control vector.

transmission header (TH). Control information, optionally followed by a basic information unit (BIU) or a BIU segment, that is created and used by path control to route message units and to control their flow within the network. See also *path information unit*.

transmission priority. A rank assigned to a message unit that determines its precedence for being selected by the path control component in each node along a route for forwarding to the next node in the route.

transport layer. In the Open Systems Interconnection reference model, the layer that provides a reliable end-to-end data transfer service. There may be relay open systems in the path. (T) See also *Open Systems Interconnection reference model*.

transport network. The part of the SNA network that includes the data link control and path control layers. Synonymous with *path control network*.

TRS. Topology and routing services.

type 2.0 node. A node that attaches to a subarea network as a peripheral node and provides a range of end-user services but no intermediate routing services.

type 2.1 node. A node that can be an APPN network node, an APPN end node, or a LEN node. It can also attach as a peripheral node to a subarea boundary node in the same way as a type 2.0 node.

type 4 node. A node that is controlled by one or more type 5 nodes. It can be a subarea node, or, together with other type 4 nodes and their owning type 5 node, it can be included in a group of nodes forming a composite LEN node or a composite network node.

type 5 node. A node that can be any one of the following:

- APPN end node
- APPN network node
- LEN node
- Interchange node
- Migration data host (a node that acts as both an APPN end node and a subarea node)
- Subarea node (with an SSCP)

Together with its subordinate type 4 nodes, it can also form a composite LEN node or a composite network node.

U

upstream. In the direction of data flow from the end user to the host. Contrast with *downstream*.

user variable. See *USERVAR*.

USERVAR. A variable whose value is the name of the logical unit to which VTAM routes session-establishment requests.

V

verb. See *LU 6.2 verb*.

virtual machine (VM). In VM, a functional equivalent of a computing system. On the 370 Feature of VM, a virtual machine operates in System/370 mode. On the ESA Feature of VM, a virtual machine operates in System/370, 370-XA, ESA/370, or ESA/390 mode. Each virtual machine is controlled by an operating system. VM controls the concurrent execution of multiple virtual machines on an actual processor complex.

virtual node. Synonym for *virtual routing node*.

virtual route (VR) pacing. In SNA, a flow control technique used by the virtual route control component of path control at each end of a virtual route to control the rate at which path information units (PIUs) flow over the virtual route. VR pacing can be adjusted according to traffic congestion in any of the nodes along the route. See also *pacing* and *session-level pacing*.

virtual-route-based transmission group. A transmission group that represents the virtual routes connecting the domains of the following nodes across a subarea network: a) two interchange nodes, b) an interchange node and a migration data host, or c) two migration data hosts.

virtual routing node. A representation of a node's connectivity to a connection network defined on a shared-access transport facility, such as a token ring. Synonymous with *virtual node*.

Virtual Telecommunications Access Method (VTAM). An IBM licensed program that controls communication and the flow of data in an SNA network. It provides single-domain, multiple-domain, and interconnected network capability.

VM. Virtual machine.

VR. Virtual route.

VTAM. Virtual Telecommunications Access Method. Synonymous with *ACF/VTAM*.

W

WAN. Wide area network.

wide area network (WAN). (1) A network that provides communication services to a geographic area larger than that served by a local area network or a metropolitan area network, and that may use or provide public communication facilities. (T) (2) A data communications network designed to serve an area of hundreds or thousands of miles; for example, public and private packet-switching networks, and national telephone networks. Contrast with *local area network (LAN)* and *metropolitan area network (MAN)*.

wild-card entry. An entry in a network directory database that implicitly represents all resources not explicitly listed.

wild-card routing. The forwarding of session-activation requests to unknown logical units by a network node having a wild-card entry in its network directory database.

X

X.25. An International Telegraph and Telephone Consultative Committee (CCITT) recommendation for the interface between data terminal equipment and packet-switched data networks. See also *packet switching*.

XID. Exchange identification.

Bibliography

- *Architectures for Object Interchange*, GG24-3296
- *Data Security Through Cryptography*, GC22-9062
- *Distributed Data Management Architecture: General Information Manual*, GC21-9527
- *Document Interchange Architecture: Technical Reference*, SC23-0781
- *Guide to SNA Publications*, GC30-3438
- *IBM SDLC Concepts*, GA27-3093
- *IBM 3270 Data Stream Programmer's Reference*, GA23-0059
- *Information Processing Systems - Open Systems Interconnection - Basic Reference Model*, ISO 7498
- *Networking Blueprint Executive Overview*, GC31-7057
- *NetView APPNTAM Feature Agent Implementation Guide*, SC31-7129
- *NetView APPNTAM Feature Diagnosis*, LY43-0019
- *NetView APPNTAM Feature Implementation Guide*, SC31-7050
- *NetView APPNTAM Feature Topology Data Model Reference*, SC31-7051
- *NetView Resouce Object Definition Manager Programming Guide*, SC31-7095
- *Non-SNA Interconnection General Information Manual*, GC33-2023
- *Systems Application Architecture: Common Programming Interface Communications Reference*, SC26-4399
- *Common Programming Interface for Communications Specification*, SC31-6180
- *Systems Network Architecture Distribution Services Reference*, SC30-3098
- *Systems Network Architecture File Services Reference*, SC31-6807
- *Systems Network Architecture Format and Protocol Reference Manual: Architectural Logic*, SC30-3112
- *Systems Network Architecture Format and Protocol Reference Manual: Architecture Logic for LU Type 6.2*, SC30-3269
- *Systems Network Architecture LU 6.2 Reference—Peer Protocols*, SC31-6808
- *Systems Network Architecture Transaction Programmer's Reference Manual for LU Type 6.2*, GC30-3084
- *Systems Network Architecture—Sessions Between Logical Units*, GC20-1868
- *Systems Network Architecture Format and Protocol Reference Manual: SNA Network Interconnection*, SC30-3339
- *Systems Network Architecture Formats*, GA27-3136
- *Systems Network Architecture Network Product Formats*, LY43-0081
- *Systems Network Architecture Management Services Reference*, SC30-3346
- *Systems Network Architecture APPN Architecture Reference*, SC30-3422
- *The X.25 Interface for Attaching Packet-Switched Data Networks General Information Manual*, GA27-3345
- *X.25 1984/1988 Interface for Attaching SNA Nodes to PSDNs General Information*, GA27-3761
- *X.25 1984/1988 Interface for Attaching SNA Nodes to PSDNs Architecture Reference*, SC30-3409
- *Token-Ring Network Architecture Reference*, SC30-3374

Index

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

R

S

T

U

V

W

X

Communicating Your Comments to IBM

Systems Network Architecture
Technical Overview

Publication No. GC30-3073-04

If you especially like or dislike anything about this book, please use one of the methods listed below to send your comments to IBM. Whichever method you choose, make sure you send your name, address, and telephone number if you would like a reply.

Feel free to comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this book. However, the comments you send should pertain to only the information in this manual and the way in which the information is presented. To request additional publications, or to ask questions or make comments about the functions of IBM products or systems, you should talk to your IBM representative or to your IBM authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

If you are mailing a readers' comment form (RCF) from a country other than the United States, you can give the RCF to the local IBM branch office or IBM representative for postage-paid mailing.

- If you prefer to send comments by mail, use the RCF at the back of this book.
- If you prefer to send comments by FAX, use this number:

  United States and Canada: **1-800-227-5088**

- If you prefer to send comments electronically, use this network ID:
  - IBM Mail Exchange: **USIB2HPD at IBMMAIL**
  - IBMLink: **CIBMORCF at RALVM13**
  - Internet: **USIB2HPD@VNET.IBM.COM**

Make sure to include the following in your note:

- Title and publication number of this book
- Page number or topic to which your comment applies.

Help us help you!

Systems Network Architecture
Technical Overview

Publication No. GC30-3073-04

We hope you find this publication useful, readable and technically accurate, but only you can tell us! Your comments and suggestions will help us improve our technical publications. Please take a few minutes to let us know what you think by completing this form.

Overall, how satisfied are you with the information in this book?	Satisfied	Dissatisfied
	☐	☐

How satisfied are you that the information in this book is:	Satisfied	Dissatisfied
Accurate	☐	☐
Complete	☐	☐
Easy to find	☐	☐
Easy to understand	☐	☐
Well organized	☐	☐
Applicable to your task	☐	☐

Specific Comments or Problems:

Please tell us how we can improve this book:

Thank you for your response. When you send information to IBM, you grant IBM the right to use or distribute the information without incurring any obligation to you. You of course retain the right to use the information in any way you choose.

Name

Address

Company or Organization

Phone No.

Help us help you!
GC30-3073-04

IBM®

Cut or
Along

Fold and Tape | **Please do not staple** | Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation
Information Development
Department E15
PO BOX 12195
RESEARCH TRIANGLE PARK, NORTH CAROLINA 27709-9990

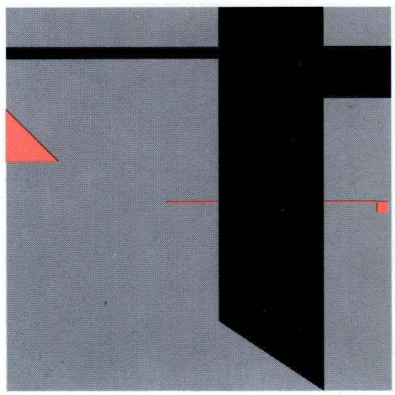
Fold and Tape | **Please do not staple** | Fold and Tape

GC30-3073-04

Cut or F
Along Li

IBM®

Printed in U.S.A.



GC30-3073-04