SV40-1010-04

Systems Network Architecture
Advanced Peer-to-Peer Networking

Dependent LU Requester Architecture Reference

Version 1.5

IBM

Systems Network Architecture
Advanced Peer-to-Peer Networking

SV40-1010-04

Dependent LU Requester Architecture Reference

Version 1.5

Note

Before using this document, read the general information under “Notices” on page xi.

Fifth Edition (September 2000)

This edition, SV40-1010-04, is a revision of the previous edition, SV40-1010-03, and obsoletes that edition; it applies until otherwise indicated in a new edition.

Additional copies of this publication can be downloaded via anonymous FTP from Internet node ftp.networking.ibm.com under the name of aiw/dlur/dlur-04.psbin - the file is a PostScript file, but treat it as binary when you get it and print it. There are also compressed versions of the file (aiw/dlur/dlur-04.zipbin and aiw/dlur/dlur-04.exebin) as well as a BookManager * version (aiw/dlur/dlur-04.bookbin).

You can check the AIW Web site at http://www.networking.ibm.com/app/aiwhome.htm, and more specifically, the Sources of Architecture Softcopy Documentation page, to see if this edition has been superceded by a more recently published edition.

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

Any comments or questions relative to this document can be sent to the following Internet address: **appn@us.ibm.com**. In addition, you may address comments to:

IBM Corporation
APPN Architecture
Department BRQA
P.O. Box 12195
Research Triangle Park, North Carolina 27709-9990
U.S.A.

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, 500 Columbus Avenue, Thornwood, NY 10594, United States.

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	Library Reader
APPN	NetView
BookManager	VTAM

About This Book

This book defines the protocols and operation of the APPN Dependent LU Requester (DLUR) functional entity, which interacts with a Dependent LU Server (DLUS). The DLUR function is an APPN enhancement for an end node or network node that supports dependent LUs. The DLUS function is a product feature of an interchange node or a T5 network node supporting session services extensions.

How This Book Is Organized

Chapter 1, “Introduction” provides an overview of existing APPN dependent LU support and an introduction to dependent LU requester/server (DLUR/S) architecture.

Chapter 2, “Requirements” provides explanation of the objective of DLUR architecture.

Chapter 3, “Functional Overview” provides a basic explanation of how DLUR/S works.

Chapter 4, “Network Management” describes how an APPN network with DLUR is managed including configuration management, problem management, operational control, and change management.

Chapter 5, “CP-SVR Pipe” explains the concept of the CP-SVR pipe. Scenarios of CP-SVR pipe activation and deactivation, error recovery, and race conditions are described.

Chapter 6, “SSCP-PU/SSCP-LU Session Encapsulation” explains how the SSCP-PU and SSCP-LU sessions relate to the CP-SVR pipe. Scenarios of activation and deactivation of both types of SSCP sessions are included.

Chapter 7, “Unformatted Session Services Management” explains, through a scenario, what happens after the SSCP-LU session is activated.

Chapter 8, “LU-LU Session Routing” describes the RUs involved in BIND routing and the concept of DLUR End Node (EN) tail vector registration.

Chapter 9, “Dependent LU Session Establishment Flows” describes, through scenarios, how LU-LU sessions are established using the DLUR/S option sets.

Chapter 10, “Dependent LU Session Awareness Management” describes how session awareness is provided to the SSCP in the DLUR/S environment.

Chapter 11, “Multi-Subnet DLUR/S” describes how the DLUR architecture works when the DLUR, DLUS, or PLU reside in different APPN subnets.

Chapter 12, “DLUR Option Set Content” describes DLUR option sets.

Chapter 13, “Format Changes” describes changes to RUs, control vectors, GDS variables, and sense data for DLUR architecture.

Appendix A, "Alternate Solutions" describes an alternate solution for LU-LU session route calculation.

Appendix B, "Sample Network Configurations" contains examples of sample network configurations.

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

Softcopy

This publication is available only as a softcopy book.

CD-ROM

The softcopy book is on an electronic bookshelf and is part of the *IBM Online Library Omnibus Edition: OS/390 Collection* (SK2T-6700) on compact disc read-only memory (CD-ROM). In the collection kit this book's file name is D50A5104.

You can view and search softcopy books by using IBM BookManager READ and BookServer products or by using the IBM Softcopy Reader or IBM Library Reader* product included on the collection. For more information on CD-ROMs and softcopy books, see *Installing, Managing, and Using the Online Library* (GC31-8311) and IBM BookManager READ and BookServer documentation.

Internet

Additional copies of this publication can be downloaded via anonymous FTP from Internet node ftp.networking.ibm.com under the name of aiw/dlur/dlur-04.psbin - the file is a PostScript file, but treat it as binary when you get it and print it. There are also compressed versions of the file (aiw/dlur/dlur-04.zipbin and aiw/dlur/dlur-04.exebin) as well as a BookManager version (aiw/dlur/dlur-04.bookbin).

You can also view softcopy books using a World Wide Web browser. If you go to the APPN Implementers Workshop home page (the URL is http://www.networking.ibm.com/app/aiwhome.htm), you will find a link to a viewer for this softcopy book.

Where to Find More Information

The following publications about related topics may be of interest:

- *SNA APPN Architecture Reference*, SC30-3422
- *SNA APPN Branch Extender Architecture Reference*, SV40-0129
- *SNA APPN Extended Border Node Architecture Reference*, SV40-0128
- *SNA APPN High Performance Routing Architecture Reference*, SV40-1018
- *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 Formats*, GA27-3136

- *Systems Network Architecture Management Services Formats*, GC31-8302
- *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 Technical Overview*, GC30-3073
- *Inside APPN: The Essential Guide To The Next-Generation SNA*, SG24-3669
- *Definitions of Managed Objects for DLUR using SMIv2*, RFC 2232
- *The Simple Book: An Introduction to Management of TCP/IP-Based Internets*, Marshall T. Rose, Prentice Hall, Englewook Cliffs, NJ, ISBN 0-13-812611-9, 1991
- *Understanding SNMP MIBs*, David Perkins and Evan McGinnis, Prentice Hall PTR, Upper Saddle River, NJ, ISBN 0-13-437708-7, 1997

Summary Of Changes

The following changes appear in this edition. All of these changes are identified with change bars (|s).

- A new section 10.3, "DLUR HPR Session Awareness" on page 10-41 was added, defining the DLUR HPR Session Awareness (DLUR HPR SAW) optional function set.
- The reference to the DLUR MIB, *Definitions of Managed Objects for DLUR using SMIv2*, was updated to point to the IETF Proposed Standard.
- The Acknowledgments section and the detailed change log were removed.
- Since this update coincides with updates to *Systems Network Architecture Formats* and *Systems Network Architecture Management Services Formats*, the Formats section has been removed from this document.

Chapter 1. Introduction

It is well known that while Advanced Peer-to-Peer Networking* (APPN) protocols originally provided enhanced networking services for independent logical units (LUs), APPN protocols before the advent of DLUR/S did not provide services for dependent LUs. Dependent LUs are those LUs requiring assistance from a System Services Control Point (SSCP) in order to activate and manage LU-LU sessions. As Systems Network Architecture (SNA) products have moved from the current subarea-based protocols to APPN-based protocols, it has been increasingly more important to provide dependent LU services in APPN. This is particularly true when one considers the enormous customer investment in dependent LU types and the anticipated long migration period toward independent LU communications.

This document describes a method by which dependent LU support can now be provided in APPN networks with minimal implementation cost and minimal customer impact. It leverages the existing investment in dependent LU architecture while taking advantage of the dynamics available in APPN networks. The architecture described in this document is intended to provide the strategic method of dependent LU support in APPN networks.

1.1 Original APPN Dependent LU Support Architecture

The architecture extensions to APPN that orignally provided support for dependent LUs are in several option sets. [1] They are as follows:

1060 - Prerequisites for Session Services Extensions Control Point (CP) Support

1061 - Prerequisites for Session Services Extensions Network Node Server (NNS) Support

1062 - Session Services Extensions Control Point (CP) Support

1063 - Session Services Extensions Network Node Server (NNS) Support

1064 - Session Services Extensions Primary Logical Unit (PLU) Node Support

1065 - Session Services Extensions CP(SLU) (SSCP) Support

These option sets cover extensions to APPN to handle the additional session initiation types used by dependent LUs (i.e., Initiate or Queue, Initiate or Notify, Queue only, Autologon, and Third-party Initiate). The basic design premise of these option sets is that current SSCP support for dependent LUs will be moved out of the large mainframe host and distributed throughout the network in the peripheral type 2.1 (T2.1) nodes. This approach moves dependent LU ownership and management out of the mainframe and into the peripheral nodes. In addition, this approach requires the SSCP code to be duplicated in peripheral nodes. And lastly, this approach moves the system definition responsibilities for dependent LUs to peripheral nodes.

This approach is a fundamental change to the way customers design, define, and manage their existing dependent LUs. In addition, the cost of implementing SSCP function in the peripheral nodes is prohibitive to most products implementing such architecture. The next section describes the newer DLUR/S approach.

[1] For more information on option sets, please refer to *SNA APPN Architecture Reference*.

1.2 Alternative Approach: Dependent LU Requester/Server Architecture

Since SSCP function already resides in the existing Virtual Telecommunications Access Method (VTAM*) subarea product, this document describes an alternative solution that leverages this support to enable dependent LUs in an APPN network. In this solution, an LU6.2 session pipe is established between the node supporting dependent secondary logical units (SLUs) and the node providing SSCP function (initially either VTAM as pure network node (NN) or as interchange node). In this scenario, the node supporting dependent SLUs is called the Dependent LU Requester (***DLUR*** - option set 1067) and the node providing SSCP function is known as the Dependent LU Server (***DLUS*** - option set 1066). Once a pair of LU6.2 sessions (using a new mode known as CPSVRMGR) have been brought up between the DLUR and DLUS, dependent physical unit/logical unit (PU/LU) flows (SSCP-PU and SSCP-LU sessions) are encapsulated over the LU6.2 sessions between the DLUR and DLUS SSCP. These LU6.2 sessions are known as the CP-to-server (CP-SVR) pipe. In this way, SSCP services are provided from VTAM without requiring the distribution of SSCP code or definition. The remainder of this document provides more detail on the architecture to provide such SSCP services to remote APPN dependent LUs.

Chapter 2. Requirements

2.1 Objectives

It is the objective of this architecture to:

- Leverage the current investments in dependent LU support to the advantage of both the implementer and its customers.
- Reduce the impact to customers in supporting dependent LUs in an APPN environment.
- Develop a solution to supporting dependent LUs in APPN that can and will be readily implemented by APPN products.

In addition to the objectives listed above, it is the intent of this architecture to be compatible with existing dependent LU support solutions. This enables full support of dependent LUs including: USS Messages, Interpret Function, Autolog Support, and Session Queuing/Notification, SLU initiation, PLU initiation, third party initiation, etc. It is also a requirement to provide both Unformatted and Formatted Session Services support. Another important requirement is the ability to handle SSCP takeover scenarios. Lastly, there is a requirement to provide session routing independent of the SSCP providing dependent LU services. In this way, more optimal routes can be achieved between the PLU and SLU.

Chapter 3. Functional Overview

3.1 Overview

Dependent LU Requester/Server architecture consists of three separate APPN option sets:

- Dependent LU Server (***DLUS***) option set (1066)
- Dependent LU Requester (***DLUR***) option set (1067)
- DLUS-served LU registration option set (1116)

The intent of these option sets is to provide dependent SLU support by establishing an LU 6.2 session pipe (known as the CP-SVR pipe) between the requester and its server. Once this pipe has been established, the requester can obtain SSCP services for its dependent LUs from the server. This is done by multiplexing the necessary SSCP-PU and SSCP-LU sessions over the CP-SVR pipe and sending the dependent LU session initiation and management flows directly between the SSCP server and the PLU node.

In this way, the dependent LU appears to be located within the domain of the serving SSCP. Session initiation flows will be emulated from the SSCP server, but session BIND and data paths will be calculated directly between the dependent LU and its session partner. This path may or may not traverse the serving SSCP.

Since the serving SSCP will be a VTAM APPN NN, it will be able to provide both the standard SSCP-SSCP subarea and Session Services Extensions APPN session initiation flows on behalf of the served dependent SLUs. When the PLU exists in the subarea network, the server will use SSCP-SSCP flows to establish a session between the PLU and served SLU. When the PLU exists in the APPN network, the server will use Session Services Extensions flows to establish a session between the PLU and served SLU.

In this sense, DLUR/S architecture will not eliminate the need for Session Services Extensions within an APPN network. For many products, however, DLUR/S architecture will eliminate the need to implement Session Services Extensions. Products that only require dependent SLU support can implement the DLUR option set of this architecture and receive the Session Services Extensions support from their SSCP server via the CP-SVR pipe. As such, most products should only be required to provide the DLUR option set, while VTAM will provide both the DLUS option set and full Session Services Extensions support.

Every NN product should consider providing the DLUS-served LU registration option set if it can act as a NNS for an end node (EN) DLUR (see 11.3.4, “Multi-Subnet EN DLUR Capabilities and Resource Registration” on page 11-6 for a description of the function in this option set).

3.1.1 Notations Used in Diagrams

The following is a list of notations and their meanings as used in this document:

Table 3-1. Diagram Notations

BN	border node
DLUR	dependent LU requester
DLUS	dependent LU server
EN	end node
LU	logical unit
NCP	network control program
NN	network node
NNS	network node server
PLU	primary logical unit
PU	physical unit
SLU	secondary logical unit
SSCP	system services control point
VTAM	virtual telecommunications access method

DLUR/S architecture is illustrated in the following figure:

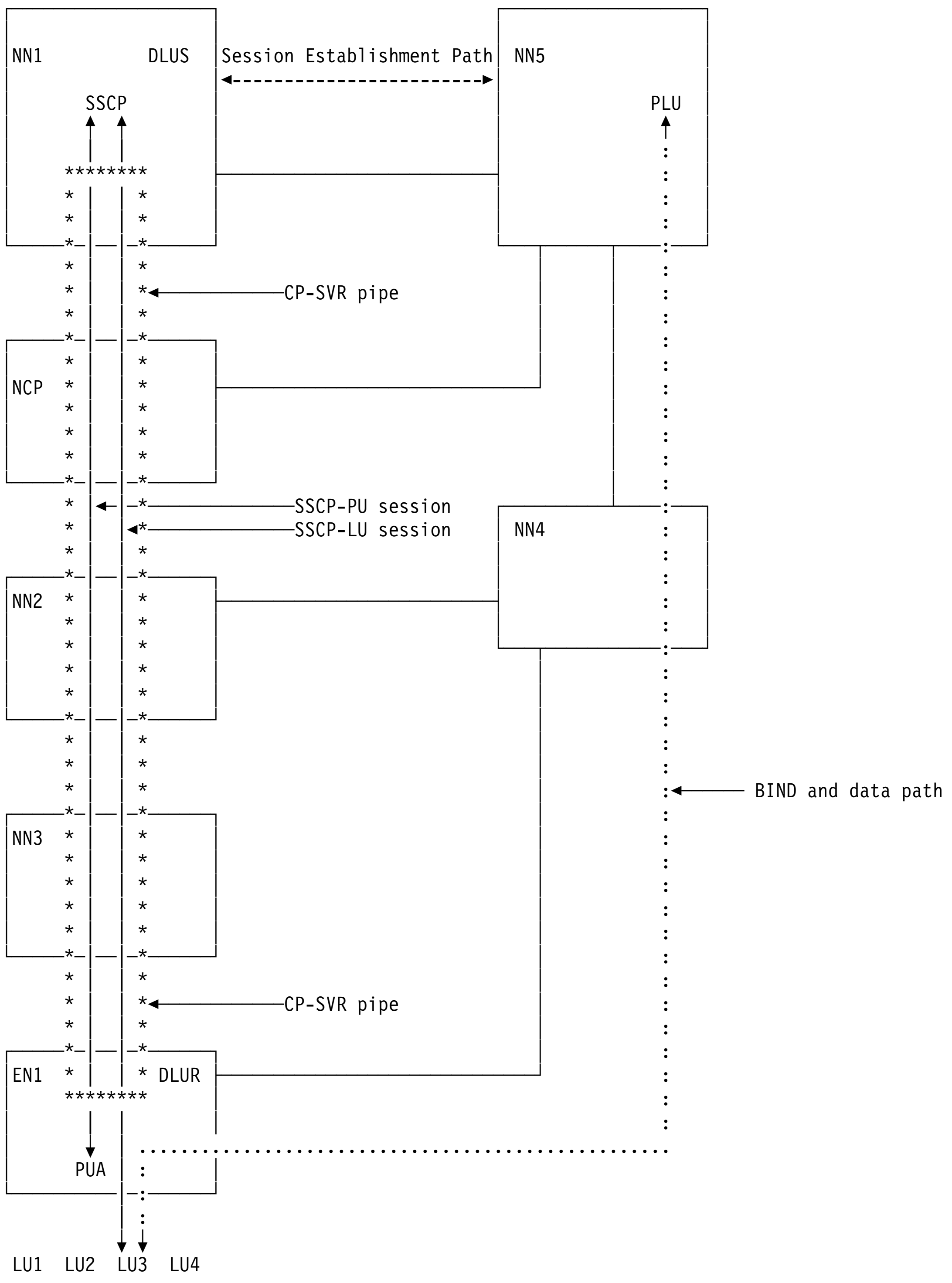


Figure 3-1. Dependent LU Requester/Server Architecture

In this figure, EN1 is supporting dependent SLUs: LU1, LU2, LU3, and LU4. EN1 supports these dependent SLUs with an internal physical unit type 2 (PU2.0) known as PUA. A CP-SVR pipe is established between EN1 and the interchange node, NN1. The SSCP-PU and SSCP-LU sessions are then encapsulated and multiplexed over the CP-SVR pipe. Session initiation flows for the dependent SLUs on EN1 are sent over the SSCP-LU session to NN1. NN1 then uses Session Services Extensions flows to initiate a session with the PLU located in the APPN network on NN5. NN5 must support the Session Services Extensions PLU functions for the specified session initiation type. Once the PLU has been located on NN5 and the necessary resource reservations, session queuing management, etc. have taken place, a BIND is issued in the APPN network directly from NN5 to EN1. In this scenario, the DLUS option set is located on NN1, the DLUR option set is located on EN1, and Session Services Extensions function is located at both NN1 and NN5. Assuming that in this case NN3 is the NNS for EN1, NN3 may or may not have the DLUS-served LU registration option set.

3.2 Assumptions And Dependencies

This architecture assumes that the DLUS architecture will be implemented by VTAM in its APPN NN implementation. VTAM APPN NNs will have the capability to provide both APPN and SSCP control point services.

Other products are assumed to implement the DLUR function to support either local or downstream (e.g., local area network (LAN)-attached disk operating system (DOS) workstations or directly attached PU2.0s) dependent SLUs.

Products that wish to provide dependent PLU support or dependent SLU support in the absence of the DLUR/S architecture must use Session Services Extensions.

APPN nodes implementing the DLUR function for ***local*** dependent LUs must implement PU2.0 images. Because these PU2.0 images will be part of the logical network, they must be uniquely named in flows or network management displays that concern them. This implies that they will not be the CP name of the DLUR node or any other node in the network for that matter.

DLUS-served resources will be represented as switched LUs which are coupled to the CP-SVR pipe. In essence, the CP-SVR pipe is the LINE for the session, although internally it cannot be represented as a LINE, since the control point name will already be represented to DLUS as an application.

3.2.1 DLUR Base/Option Set Dependencies

The DLUR option set (1067) can be implemented on either Version 1 or Version 2 Base APPN, and on either NN or EN.

3.2.2 Configuration Services XID Exit Support

In order to dynamically create a PU definition for APPN dependent LUs, the DLUS node must provide the Configuration Services Exchange Identification (XID) Exit support provided as part of the Dynamic Network Access function of VTAM Version 3 Release 4. This function will allow the DLUS node to dynamically create a PU definition for the DLUR PUs based upon information sent on the CP-SVR pipe.

LUs can also be dynamically defined using this facility, but the newer Dynamic Definition of Dependent LUs Support (see 3.2.3, “Dynamic Definition of Dependent LUs Support”) should be used.

As a customer option, dependent LUs may also be predefined at the DLUS node; however, this requires that the corresponding PU also be predefined.

3.2.3 Dynamic Definition of Dependent LUs Support

VTAM Version 3 Release 4.1 introduces Dynamic Definition of Dependent LUs (DDDLU) Support, which uses the Product Set Identifier (PSID) information sent on SSCP-PU sessions to dynamically create LU definitions for downstream dependent LUs. DLUR/S architecture requires this support in both the DLUS and the PU node if dynamic LU definition is desired. The PU node must provide PSID information to the DLUS, who will interpret it and dynamically create LU definitions for the APPN dependent SLUs at or downstream from the DLUR node. If this function does not exist at one of the nodes, then predefinition of dependent LUs is required. For further information on DDDLU function see SNA Networking Update: Dynamic Definition of Dependent Logical Units (published by SNA Networking Vendor Enablement).

3.2.4 Network Management

Most of the network management concerns for dependent LU supported via the DLUR/S architecture will be met by existing NMVT support. Since the SSCP-PU flows remain intact and are simply encapsulated on the CP-SVR pipe, all dependent LU management of this type will be preserved. DLUR/S architecture, however, does present a unique management concern with respect to the CP-SVR pipe. Management Services components will be required to understand the relationship between SSCP-PU and SSCP-LU sessions, the CP-SVR pipe, and the real resources that support them.

3.2.5 DLUS-Served LU Registration Base/Option Set Dependencies

The DLUS-served LU registration function is implemented in two parts:

- the EN DLUR impacts are incorporated into the base DLUR architecture (option set 1067)
- the NNS impacts are incorporated into option set 1116.

EN DLURs will support NNSs with and without option set 1116.

The DLUS-served LU registration option set (1116) can be implemented on either Version 1 or Version 2 Base APPN for NN only.

For details about the DLUS-served LU registration function, refer to 11.3.4, “Multi-Subnet EN DLUR Capabilities and Resource Registration” on page 11-6.

Chapter 4. Network Management

The Dependent LU Requester architecture provides a migration path for existing SNA/Management Services (SNA/MS) management of PU resources from an SSCP. All the NetView* applications for management of dependent resources that were developed for the pure subarea environment continue to work unchanged on a system with the Dependent LU Server function. However, for simple network management protocol (SNMP)-managed APPN nodes, a management information base (MIB) is needed to represent the DLUR functions in the node. The relationship between NetView management of the PU and LU resources and SNMP management of the DLUR node is illustrated in Figure 4-1.

Note: Refer to Table 3-1 on page 3-2 for an explanation of the notation used in the diagrams in this chapter.

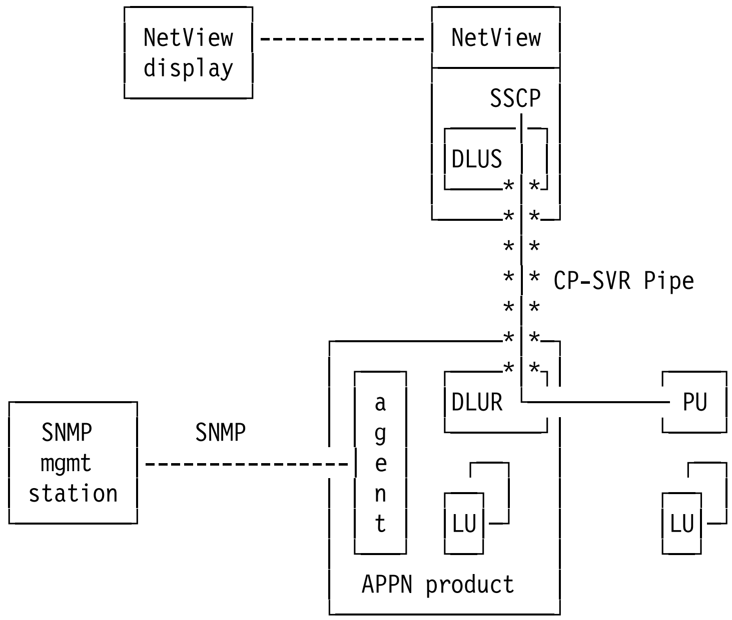


Figure 4-1. Relationship of SNA/MS Management to SNMP Management for DLUR

The DLUR MIB supports, from the SNMP management station, viewing of DLUR configuration information, and status monitoring of the PUs supported by the DLUR, as well as of the DLUR′s relationship(s) with its serving DLUS(s).

4.1 Naming Of DLUS-Supported Resources

The DLUR node learns the names of the PU and LU resources when the DLUS node activates them via ACTPU and ACTLU over the CP-SVR pipe. These names flow in the Network Name (X′0E′) control vector (CV). The Network Name control vector on ACTPU has the resource type type X′F1′ (PU name), while the one on ACTLU has resource type X′F3′ (LU name). The Network Name control vector is optional in ACTPU and ACTLU themselves; it is, however, always included when these RUs are sent over the CP-SVR pipe. The PU name in the CV X′0E′ is network-qualified when the PU network identifier (NETID) is different from the SSCP′s NETID.

The DLUR BF will usually remove PU and LU names received in ACTPU and ACTLU RUs before forwarding the RUs on to the destination resource. However, a user can indicate to the DLUS that, for a particular PU and its associated LUs, the DLUR should not remove the PU and LU names from the ACTPU and ACTLU RUs when forwarding them (for more details about PU name forwarding, see 6.2.4, “PU/LU Network Name Forwarding” on page 6-25).

The DLUR saves the PU name from this CV X′0E′ for network management.

4.2 Overview of DLUR Management by Discipline

The following sections summarize DLUR management by management discipline.

4.2.1 Configuration Management

The DLUR MIB supports the following categories of configuration information.

1. Information about the capabilities of the DLUR implementation.

   This is the information that the DLUR implementation sends in the DLUR/S Capabilities (X′51′) CV.

2. Information about the status of the CP-SVR pipes.

   The status of the relationships that the DLUR has with each DLUS must be available. This is information in addition to the standard information maintained about every LU-LU session, and returned via the

   the Internet Engineering Task Force (IETF) APPC MIB. When these sessions are ended abnormally, information should be logged by the DLUR product for possible use in problem diagnosis. Mechanisms for local logging of error information are typically product-unique, and not specified in the architecture. For debugging problems, it may be necessary to trace the messages exchanged over these sessions. Product-unique serviceability tools are the most appropriate method for capturing this information when it is needed.

3. Information about each DLUR-supported PU.[1]

   This information includes:

   a. The PU name from the CV X′0E′ received on ACTPU.
   b. Name(s) of the DLUS(s) defined for the PU.
   c. The state of the CP-SVR pipe to the PU′s current DLUS.
   d. Location of the PU (internal to DLUR node or downstream); for a downstream PU, the MIB also identifies the SNA link between the DLUR and the PU.
   e. Detailed state of the PU, reflecting all the intermediate states in the activation and session setup protocols.

1

An earlier version of the DLUR MIB relied on the SNA NAU MIB, *Definitions of Managed Objects for SNA NAUs using SMIv2* (RFC 1666), for returning basic information about DLUR-served PUs. The current version of the DLUR MIB does not have this dependency.

More generally, the original idea of making the SNA NAU MIB a common base that would be extended by the other SNA MIBs (for APPC, APPN, DLUR, etc.) has been discarded. Each of these SNA MIBs is now a stand-alone vehicle for managing its particular set of SNA functions. Even though it is now primarily of historical interest, the SNA NAU MIB continues to be available via anonymous FTP from any of the IETF repositories on Internet.

4.2.2 Problem Management

There are two major elements to problem management for a DLUR: problem reporting (via SNA/MS Alerts or by SNMP traps based upon those Alerts), and the collection of diagnostic information that can be examined after a problem occurs.

1. Problem reports are needed for:

   a. Protocol errors, such as badly formed message units or messages received in inappropriate sequence.
   b. Installation errors, such as capabilities mismatches, incorrect definitions, or invalid configurations that prevent normal operation.
   c. Implementation limits that prevent normal operations, such as exhausted storage for control blocks.

   Problem reports are not appropriate for situations that are handled as normal operations by the DLUR/DLUS components, without the need for operator intervention.

2. Collection of diagnostic information:

   Information about possible problems that may be needed for detailed diagnosis or debugging should be saved for some period of time. Some of this information is made available via the standard SNMP MIBs for SNA (specifically, the APPC, APPN, and DLUR MIBs). Other types utilize the well-established mechanisms for capturing and analyzing historical diagnostic information that already exist for the products implementing DLUR. Some of these product-unique diagnostic tools may be based upon SNMP (via private MIBs), or they may utilize off-line tools (log formatters).

   It is recommended that whenever the CP-SVR sessions are terminated for reasons other than normal operations (operator shutdown or loss of network connectivity), that at least the following information be saved locally:

   Date/time
   PLU name
   SLU name
   Mode name
   Unbind type
   Sense data
   Detecting module name or equivalent
   Copy of any pertinent data (such as bad data received)
   . . .

   Since CP-SVR sessions use LU 6.2, DLUR implementations that implement the APPC MIB can make this information available through the appcHistSessTable, just as they would for other abnormally terminated LU 6.2 session. The PLU and SLU would identify the DLUR and DLUS, and the mode name would always be CPSVRMGR.

4.2.3 Change Management

In SNA/MS, change management refers to the distribution, installation, and change control of software components. In this sense, change management is outside the scope of this document, since a DLUR does nothing to change the way software is controlled, either on the DLUR node or on the supported PU nodes.

4.2.4 Summary

There are two DLUR-unique network management functions defined in this document:

1. A function set providing current session awareness data for dependent LU-LU sessions in an APPN/HPR network.
2. SNA/MS Alerts for reporting problems detected by the DLUR implementation.

In addition to these functions, SNMP-based management for DLUR resources is provided by the DLUR MIB, *Definitions of Managed Objects for DLUR using SMIv2*.

4.3 The DLUR MIB

For more information on the SNMP MIB notation and on SNMP MIB design, please refer to *Understanding SNMP MIBs* written by David Perkins and Evan McGinnis.

The DLUR MIB is available from the IETF as *Definitions of Managed Objects for DLUR using SMIv2* (November 1997).

4.4 SNA/MS Alerts

4.4.1 DLUR Protocol Errors

The following conditions are covered by a single Alert indicating that the DLUR-DLUS communication has been disrupted because a protocol error has been detected.

- GDS Variable Other Than X′1500′ Received After RECEIVE_ENCAP Attach

  If the Receive_Encap_Msg_TP is attached and the GDS variable received is not X′1500′, the DLUR will send an UNBIND request with sense data X′1001 0000′.

- Received GDS Variable X′1500′ Without CV X′60′

  Any path information unit (PIU) request encapsulated in a GDS variable X′1500′ which does not include a CV X′60′ will be discarded. The sense data associated with this condition is X′086C 6000′, which is included in the Alert to describe the problem, but is not actually returned in any message flows.

- Received GDS Variable X′1500′ (ACTPU Encapsulated) With Missing Or Incomplete CV X′46′

  Any ACTPU request encapsulated in a GDS variable X′1500′ which does not include a complete CV X′46′ (e.g., CV missing its signalling information or CV missing completely) will be rejected with sense data X′0806 0000′.

- Received GDS Variable X′1500′ (ACTPU Encapsulated) Without CV X′51′

  Any ACTPU request encapsulated in a GDS variable X′1500′ which does not include a CV X′51′ when one is expected (first ACTPU after the CP-SVR pipe is activated) will be rejected with sense data X′086C 5100′.

- Received GDS Variable X′1500′ (RSP(REQACTPU) Encapsulated) Without CV X′51′

  Any REQACTPU response encapsulated in a GDS variable X′1500′ which does not include a CV X′51′ when one is expected (first REQACTPU after the CP-SVR pipe is activated) will be rejected by rejecting the ensuing corresponding ACTPU with sense data X′086C 5100′.

- CPSVRMGR Session Over Non-TCP/IP LEN Connection

  If a CPSVRMGR session is established over a LEN connection which is not a connection into a TCP/IP network, the DLUS will send an UNBIND request with sense data X′0877 0056′.

4.4.1.1 APPN/DLUR Alert CPDL001

Alert Condition:

A protocol error was detected by the DLUR component of the APPN product, causing communications to be terminated. This Alert covers sense data:

X′0806 0000′
X′086C 5100′
X′086C 6000′
X′0877 0056′
X′1001 0000′

Alert ID Number		X′9E45 2D9C′
Alert Type	X′01′	Permanent
Alert Description	X′3122′	APPN-DLUR protocol error
Probable Causes	X′1023′ X′1022′	Communication program in remote node Communication program
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X′22A1′ X′82′ SF	Communications program in remote node: Data ID=X′82′ (CP)
Actions (note 1)	X′3110′ X′32C0′ X′82′ SF X′82′ SF	Contact communications systems programmer Report the following: Data ID=X′15′ (SNA sense data) Data ID=X′F0′ (Product Alert Reference Code) (note 2)
Additional SVs	X′01′ SV X′05′ SV X′10′ SF X′10′ SV X′48′ SV X′82′ SF X′82′ SF	Date/Time Hierarchy/Resource List Hierarchy Name List Sending node: Name=(CP name) Type=X′F4′ (CP) Alert Sender Product Set ID Supporting Data Correlation (note 3) Data ID=X′D0′ (File Name) Data ID=X′D1′ (Log Record Number)

Notes:

1. The recommended actions listed here are simply *recommended*. An implementation is encouraged to select different recommended actions if they are more meaningful for a particular product. For example, it is possible to identify product-unique configuration parameters in some recommended actions.

2. The Product Alert Reference Code is a label that a product may provide as a key to more detailed documentation about the Alert. This documentation may also include detailed product diagnostic information, such as identifying those program modules involved in the detection of the error.

   An alternative to this latter use of the Product Alert Reference Code could be to include the detecting module name (data ID=X′A2′).

3. The Supporting Data Correlation (X′48′) SV is present to report that diagnostic data for this Alert was stored by the Alert sender. The X′48′ SV provides a great deal of flexibility in terms of how supporting data may be identified. The use of ″file name″ and ″log record number″ shown here is

just one example that may not be appropriate for all implementations. If an implementation is unable to capture any supporting data, this subvector may be omitted.

4.4.2 DLUR Configuration Errors

These are problems caused by incorrect configuration by the network administrator.

4.4.2.1 APPN/DLUR Alert CPDL002

Alert Condition:

While configuring this DLUR product, a network administrator has mistakenly specified as the server for one or more PUs a node that does not support the DLUS function. For example, the specified DLUS may in fact be a DLUR, or it may be an earlier release of a product that subsequently added support for DLUS. This Alert is sent by a DLUR node when a CPSVRMGR session that it initiated is unbound by the session partner with sense data X′088E 0009′. In that scenario, the session partner (the intended DLUS) has rejected the the DLUR/S Capabilities (X′51′) control vector from this node. The most likely explanation is that the configuration error was made at the DLUR that initiated the CPSVRMGR session, not at the node that rejected it.

Alert ID Number		X′EBAA 3C4F′
Alert Type	X′01′	Permanent
Alert Description	X′800A′	APPN-DLUR configuration error
Probable Causes	X′1022′ X′1023′	Communications program Communications program in remote node
User Causes	(none)	
Install Causes	X′80B9′ X′82′ SF	Node (detailed data qualifier) was incorrectly specified as a dependent LU server Data ID=X′82′ (CP)
Actions (note 1)	X′1500′	Correct installation problem
Failure Causes	(none)	
Additional SVs	X′01′ SV X′05′ SV X′10′ SF X′10′ SV	Date/Time Hierarchy/Resource List Hierarchy Name List Sending node: Name=(CP name) Type=X′F4′ (CP) Alert Sender Product Set ID

Notes:

1. The recommended actions listed here are simply *recommendations*. An implementation is encouraged to select different recommended actions if they are more meaningful for a particular product. For example, it is possible to identify product-unique configuration parameters in some recommended actions.

4.4.2.2 APPN/DLUR Alert CPDL003

Alert Condition:

CPSVRMGR Session Limit Exceeded

DLUR products will have the option to limit the number of concurrently active CP-SVR pipes (what number will be selected as the limit will also be a product option). If a limit is implemented and reached, any attempts to activate additional CP-SVR pipes will be rejected by sending an UNBIND request with sense data X′0812 0000′.

When a DLUR product reaches this limit, it should send this alert so that the network administrator will understand the implications of the problem and take the appropriate action.

Alert ID Number		X′754A EAE8′
Alert Type	X′01′	Permanent
Alert Description	X′800A′	APPN-DLUR configuration error
Probable Causes	X′1022′ X′1023′	Communication program Communication program in remote node
User Causes	(none)	
Install Causes	X′171F′	Limit for DLUR control sessions exceeded
Actions (note 1)	X′1500′	Correct installation problem
Failure Causes	(none)	
Additional SVs	X′01′ SV X′05′ SV X′10′ SF X′10′ SV	Date/Time Hierarchy/Resource List Hierarchy Name List Sending node: Name=(CP name) Type=X′F4′ (CP) Alert Sender Product Set ID

Notes:

1. The recommended actions listed here are simply *recommendations*. An implementation is encouraged to select different recommended actions if they are more meaningful for a particular product. For example, it is possible to identify product-unique configuration parameters in some recommended actions.

4.4.2.3 APPN/DLUR Alert CPDL004

Alert Condition:

While configuring this DLUS product, a network administrator has mistakenly specified the name of another DLUS node as the requester for one or more PUs. This alert is sent by a DLUS node when a CPSVRMGR session that it initiated is unbound by the session partner with sense data X′088E 000F′. In that scenario, the session partner (the intended DLUR) has rejected the the DLUR/S Capabilities (X′51′) control vector from this node. The most likely explanation is that the configuration error was made at the DLUS that initiated the CPSVRMGR session, not at the DLUS node that rejected it.

Alert ID Number		X′E9C3 C9C6′
Alert Type	X′01′	Permanent
Alert Description	X′800A′	APPN-DLUR configuration error
Probable Causes	X′1022′ X′1023′	Communication program Communication program in remote node
User Causes	(none)	
Install Causes	X′80BA′ X′82′ SF	Node (detailed data qualifier) was incorrectly specified as a dependent LU requester Data ID=X′82′ (CP)
Actions (note 1)	X′1500′	Correct installation problem
Failure Causes	(none)	
Additional SVs	X′01′ SV X′05′ SV X′10′ SF X′10′ SV	Date/Time Hierarchy/Resource List Hierarchy Name List Sending node: Name=(CP name) Type=X′F4′ (CP) Alert Sender Product Set ID

Notes:

1. The recommended actions listed here are simply *recommendations*. An implementation is encouraged to select different recommended actions if they are more meaningful for a particular product. For example, it is possible to identify product-unique configuration parameters in some recommended actions.

4.4.2.4 APPN/DLUR Alert CPDL005

Alert Condition:

CPSVRMGR Session Across Subnet Boundaries Not Supported

X′088E 000F′ - An attempt was made to establish a CP-SVR pipe across a subnet boundary between a dependent LU server and a dependent LU requester with limited multi-subnet support. The network administrator will be advised of the configuration problem so that it can be corrected.

Alert ID Number		X′E3B7 48C9′
Alert Type	X′01′	Permanent
Alert Description	X′800A′	APPN-DLUR configuration error
Probable Causes	X′1023′ X′1022′	Communication program in remote node Communication program
User Causes	(none)	
Install Causes	X′80BB′ X′82′ SF	Node (detailed data qualifier) does not support a dependent LU server with a different network ID. Data ID=X′82′ (CP)
Actions (note 1)	X′1500′	Correct installation problem
Failure Causes	(none)	
Additional SVs	X′01′ SV X′05′ SV X′10′ SF X′10′ SV	Date/Time Hierarchy/Resource List Hierarchy Name List Sending node: Name=(CP name) Type=X′F4′ (CP) Alert Sender Product Set ID

Notes:

1. The recommended actions listed here are simply *recommendations*. An implementation is encouraged to select different recommended actions if they are more meaningful for a particular product. For example, it is possible to identify product-unique configuration parameters in some recommended actions.

4.4.2.5 APPN/DLUR Alert CPDL006

Alert Condition:

CPSVRMGR Session Not Supported Across a Branch Downlink

X′0877 005B′ - An attempt was made to establish a CP-SVR pipe across a branch downlink. The network administrator will be advised of the configuration problem so that it can be corrected.

Note: A BrNN that supports downstream DLURs does not send this Alert, since it can support CPSVRMGR sessions over its branch downlinks.

Alert ID Number		X′2241 9C9D′
Alert Type	X′01′	Permanent
Alert Description	X′800A′	APPN-DLUR configuration error
Probable Causes	X′1023′ X′1022′	Communication program in remote node Communication program
User Causes	(none)	
Install Causes	X′805A′ X′F001′ X′F0A0′ X′82′ SF	A CPSVRMGR session cannot be established over a path having a branch uplink as an intermediate hop The Alert sender is the owner of the branch uplink Name of the DLUR or DLUS for which the attempt was made to establish the session is Data ID=X′82′ (CP)
Actions (note 1)	X′1500′	Correct installation problem
Failure Causes	(none)	
Additional SVs	X′01′ SV X′05′ SV X′10′ SF X′10′ SV	Date/Time Hierarchy/Resource List Hierarchy Name List Sending node: Name=(CP name) Type=X′F4′ (CP) Alert Sender Product Set ID

Notes:

1. The recommended actions listed here are simply *recommendations*. An implementation is encouraged to select different recommended actions if they are more meaningful for a particular product. For example, it is possible to identify product-unique configuration parameters in some recommended actions.

4.4.3 Switched Major Node at DLUS Not Active

One Alert reports that a DLUS failed to activate a PU when the DLUR requested that it do so.

4.4.3.1 APPN/DLUR Alert CPDL007

Alert Condition:

A request from a DLUR to a DLUS for activation of a PU failed because the switched major node for the PU was not active. The request can be retried once the correct switched major node has been activated.

Alert ID Number		X′1CB5 64E8′
Alert Type	X′02′	Temporary
Alert Description	X′3124′	DLUS failed to activate PU
Probable Causes	X′1023′ X′1022′	Communications program in remote node Communication program
User Causes	X′71A9′ X′82′ SF	PU activation failed because a switched major node at DLUS (detailed data qualifier) was not active Data ID=X′82′ (CP)
Actions (note 1)	X′1350′	Activate switched major node then retry
Install Causes	(none)	
Failure Causes	(none)	
Additional SVs	X′01′ SV X′05′ SV X′10′ SF X′10′ SV	Date/Time Hierarchy/Resource List Hierarchy Name List Sending node: Name=(CP name) Type=X′F4′ (CP) Alert Sender Product Set ID

Notes:

1. The recommended actions listed here are simply *recommendations*. An implementation is encouraged to select different recommended actions if they are more meaningful for a particular product. For example, it is possible to identify product-unique configuration parameters in some recommended actions.

Chapter 5. CP-SVR Pipe

5.1 Overview

CP-SVR pipe initiation is usually tied to PU activation. A DLUS or DLUR node will normally only activate a CP-SVR pipe when a PU requires activation and no CP-SVR pipe between the DLUR and the DLUS (or SSCP) supporting the PU already exists.

- For those situations when a CP-SVR pipe must be activated without an associated PU requiring activation, or when a CP-SVR pipe must remain active after all associated PUs have been deactivated, a ***persistent*** CP-SVR pipe can be activated (for more information about persistent CP-SVR pipes, see 5.8, “Persistent CP-SVR Pipes” on page 5-75).

A PU would require such activation under several circumstances including:

1. Receipt of an XID requesting ACTPU
2. Take over from loss of an active CP-SVR pipe
3. Explicit node operator command

In addition, the CP-SVR pipe may be initiated in one of two directions:

1. From the requester to the server
2. From the server to the requester

It is the intent of this architecture to support dynamic establishment of the CP-SVR pipe with an option to explicitly define the requester/server relationships if so desired. Due to the complexities of such dynamics, however, this version of the document requires that the DLUR and DLUS relationships be predefined. Dynamic determination of a DLUS by a DLUR node as well as dynamic determination of a DLUR by a DLUS node will be addressed by a separate architecture extension.

The following sections will describe how the CP-SVR pipe is established in both the DLUS to DLUR and DLUR to DLUS direction.

5.2 CPSVRMGR Mode

The pair of LU 6.2 sessions which make up a CP-SVR pipe will use a new mode known as CPSVRMGR. CPSVCMG will not be used due to architecture assumptions regarding adjacency of CPSVCMG session partners. SNASVCMG will not be used to avoid contention for the session with other node functions such as network management and change number of sessions (CNOS) negotiation. An additional benefit of using a new mode (CPSVRMGR) is that these sessions can be uniquely identified by the CPSVRMGR mode name, thus allowing the node to provide special session management logic to these sessions as described in 5.5, “CP-SVR Pipe Activation” on page 5-8.

The CPSVRMGR mode will use the SNASVCMG class of service (COS). By using this default COS definition, DLUR nodes will not have to worry about matching COS definitions at the NNS. In this way, a DLUR EN can be supported by an existing base APPN NN.

5.3 DLUR-Initiated CP-SVR Pipe Activation

By definition, all dependent LUs require the services of an SSCP. This is done via SSCP-PU and SSCP-LU sessions. The SSCP-PU session flows between the SSCP and PU2.0 or T2.1 node providing local support for the dependent LU. This session is created when the SSCP sends an ACTPU to the PU2.0/T2.1 node. The SSCP-LU session is created when the SSCP sends an ACTLU over the SSCP-PU session to the dependent LU itself. Normally a T2.1 or PU2.0 device supporting dependent LUs is Boundary Function (BF)-attached to a subarea node and is either provided the SSCP services upon activation or (in the case to T2.1 nodes) requests them via XID3 (this is reflected in the ACTPU suppression indicator: bytes 8-9 bit 8 of XID3).

DLUR nodes are APPN T2.1 nodes (EN or NN) which provide both local and downstream support for dependent LUs. Since DLUR nodes provide BF-like support for these LUs, each DLUR node must signal to the SSCP (or DLUS node in this case) when a PU (either local or downstream) requires SSCP services. For locally supported dependent LUs, the DLUR node requests SSCP services upon node activation. For downstream nodes supporting dependent LUs, the DLUR node waits for the downstream node to request them via XID. When a T2.1 or T2.0 PU exchanges XIDs (or equivalent) with a DLUR node, it signals via XID that it requests SSCP services (this is done by setting the ACTPU suppression indicator, bytes 8-9 bit 8 of XID3, to 0; for XID0 or SNRM without XID, ACTPU is always sent).

DLUR nodes request SSCP services from a DLUS node via the CP-SVR pipe and the REQACTPU RU. When a DLUR node detects the need for SSCP services (as described above), it activates a conwinner CPSVRMGR session (assuming one does not already exist) to a DLUS node and sends a REQACTPU RU. The DLUS node then responds by sending an ACTPU to the T2.0 PU/T2.1 supporting the dependent LUs via a conwinner CPSVRMGR session to the DLUR node. This creates an SSCP-PU session encapsulated in a CP-SVR pipe, enabling the DLUS node to provide SSCP services to the dependent LUs. A DLUR optionally may establish a conwinner CPSVRMGR session with a DLUS prior to receiving any requests for SSCP services from its PUs, e.g., a ***persistent*** CP-SVR pipe can be activated (for more information about persistent CP-SVR pipes, see 5.8, “Persistent CP-SVR Pipes” on page 5-75).

5.3.1 DLUR PU Identification

When the DLUR sends a REQACTPU RU it must identify the PU for which it is requesting SSCP services. The PU will be identified by including with the REQACTPU one or both of the following:

1. ***XID*** - This is either an image of an XID received from a PU externally attached to the DLUR or a DLUR-generated XID image (for an internal PU or an external PU not supporting XID transmission).
2. ***PU TG Descriptor*** - This item consists of one or both of the following:
    a. ***PU Name*** - The PU name can be sent to the DLUS, either to match a predefined PU statement or to be used in the creation of a dynamic PU at the DLUS. The PU name can be sent in a network name (CV X′0E′ - type X′F1′ = PU name) control vector either with or without a NETID:
        1) non-network qualified (without NETID) -in this case the DLUS will match only on PU name.
        2) network qualified (with NETID) - in this case the DLUS will match on NETID.PUname. It is an implementation option for the DLUR to choose how to get the PU name and the NETID to put together a network-qualified PU name to send to the DLUS. The DLUR could require the user to define a network-qualified PU name, or only define a PU name and use its own NETID, or in the case of a downstream PU learn about the NETID via XID (remember: a

downstream PU could have a NETID different from the DLUR′s NETID; it is the PU′s NETID which will be included in the network-qualified PU name).

b. ***Signaling Information*** - This parameter will identify the DLUR-to-PU connectivity. A new group of subfields (X′91′ - X′F0′) is being defined for both subarea Extended DIALNO (to be included in CV X′69′) and DLUR/S (to be included in CV X′46′) in order to carry the descriptions (a.k.a. signaling information). This information must be coordinated between the DLUR and DLUS nodes:

   1) it can be defined completely at the DLUR - in this case some or all of the information can optionally be sent to the DLUS to be used for network management, input into the Configuration Services XID exit for connection verification and accounting, and PU correlation

   2) it can be defined completely at the DLUS - in this case all of the information will be sent to the DLUR on the subsequent ACTPU

   3) it can be defined partially at the DLUS and partially at the DLUR - in this case the DLUR can optionally send the DLUR-defined signaling information (for the reasons described above) while the DLUS-defined signaling information will be sent on the subsequent ACTPU

      - if upon receipt of the ACTPU, the DLUR determines it has incomplete signaling information, it can optionally proceed with the activation (using defaults) or it can fail the activation with sense code X′086D 46mm′, where mm = the subfield key of a missing subfield
      - if upon receipt of the ACTPU, the DLUR determines it has the same signaling information subfield from both the DLUR and DLUS, but the subfields have different values, it will optionally:
         – allow the DLUS values to prevail
         – allow the DLUR values to prevail
         – reject the ACTPU with sense code X′0877 0059′
      - if upon receipt of the REQACTPU, the DLUS determines it has the same signaling information subfield from both the DLUR and DLUS, but the subfields have different values, it can optionally reject the REQACTPU with sense code X′0877 005A′

When both PU name and CP name are sent to the DLUS, these names must be different.

5.3.1.1 DLUS PU Search Order

When a DLUS receives a REQACTPU, the search order to find the correct PU is as follows:

1. PU name - if the DLUR supplies a PU name and if it matches a predefined PU name, the search is over; otherwise,
2. IDBLK/IDNUM - if the DLUR supplies an IDBLK/IDNUM and it matches a predefined IDBLK/IDNUM, the search is over (note that the PU names may not match - the DLUR may accept the DLUS-provided name); otherwise,
3. CP name - if the DLUR supplies a CP name and it matches a predefined CP name, the search is over (note that the PU names may not match - the DLUR may accept the DLUS-provided name); otherwise,
4. user exit - if an exit exists, a dynamic PU will be created (note that the PU names may not match - the DLUR may accept the DLUS-provided name); otherwise, the search will fail

5.3.1.2 DLUR PU Search Order

When a DLUR receives an ACTPU, the search order to find the correct PU is a product option:

- Option A - PU name does not have to match
  1. PU name - this name does not have to match a DLUR-defined name for the same PU, and the DLUS-provided PU name will be accepted
  2. CP name and/or IDBLK/IDNUM - whatever is received must match its corresponding definitions at the DLUR; otherwise, the ACTPU will be rejected (since the CP names and/or IDBLK/IDNUMs do not match)
- Option B - PU name must match
  1. PU name - if the DLUS supplies a PU name and if it matches a predefined PU name, the search is over; otherwise,
  2. IDBLK/IDNUM - if the DLUR supplies an IDBLK/IDNUM and it matches a predefined IDBLK/IDNUM, then
     a. if the predefined IDBLK/IDNUM has no PU name defined with it, the search is over and the DLUS-supplied PU name will be used; otherwise,
     b. if the predefined IDBLK/IDNUM has a PU name defined with it, the ACTPU will be rejected (since the PU names do not match);

     otherwise, the ACTPU will be rejected (since the IDBLK/IDNUMs do not match)

For more information about SSCP-PU related session initiation information, please refer to 6.2, “SSCP-PU Session Activation” on page 6-8.

5.3.2 DLUS Node Determination

In order to establish a CP-SVR pipe with a DLUS node, the DLUR must either ***know*** or ***determine*** the DLUS node to be contacted. The following sections describe how these two scenarios are handled:

5.3.2.1 Predefined DLUS Node Determination

A DLUS node is said to be ***known*** when this information is predefined (sysdefed) at the DLUR node. Since the relationship is between PUs and SSCPs, this means that the DLUR would have to have some product-specific definition that gives the CP name of a DLUS node to be contacted when a given PU requires activation. This can be done in any fashion suitable to the DLUR implementing product (e.g., defaults or operator-defined).

Once defined, the DLUR component will use this information (when necessary) to bring up the CP-SVR pipe to a DLUS node. When a given T2.0 PU/T2.1 node requires an ACTPU, the DLUR node will use the predefined information to determine the DLUS node to be contacted. It must then check to see if a conwinner CPSVRMGR session already exists with that DLUS node. If such a session does not already exist, the DLUR uses the Send_Encap_Msg_TP to issue an ALLOCATE verb with the DLUS′s CP name as the partner LU name. This will trigger the normal APPN session initiation flows to occur (i.e., a search being issued, either broadcast or directed, for the DLUS node) with an eventual BIND message flowing between the DLUR and DLUS node. This session will use the CPSVRMGR mode.

If the CPSVRMGR session to the DLUS node fails for any reason (e.g., DLUS not active or routes not available), the dependent LUs will not be activated. Appropriate notification should be signalled at the DLUR node. If driven by the XID of a downstream dependent LU device, the link between the DLUR and the downstream device will remain active, but the DLUR must either redrive the DLUS request via some product-specific mechanism (e.g., backup DLUS definition, timer-driven retry, event-driven retry, or operator-driven retry), or wait for a DLUS-driven PU activation as described in 5.4, “DLUS-Initiated CP-SVR Pipe Activation” on page 5-6.

5.3.2.2 Dynamic DLUS Node Determination

If no DLUS name is defined at the requester node, then the DLUR node must first find an appropriate DLUS node and then initiate a session to it as described in 5.3.2.1, “Predefined DLUS Node Determination” on page 5-4. Dynamic determination of a DLUS node by a DLUR node will be addressed at a later date by a separate architecture extension.

5.4 DLUS-Initiated CP-SVR Pipe Activation

In the normal case, dependent LUs are activated from the SSCP. In order to simulate the scenario when VTAM activates a PU, the DLUR/S architecture will also support dependent LU activation from a DLUS node. In this scenario, the DLUS node must first determine the DLUR node associated with a given dependent LU. If no conwinner CPSVRMGR session already exists with this node, it will activate one. It may then send an ACTPU over the conwinner CPSVRMGR session to the DLUR node, who will in turn send the ACTPU to the appropriate node (downstream T2.0 PU/T2.1 or DLUR node itself). The DLUR wil activate a conwinner CPSVRMGR session to the DLUS (if one is not already active) and send the RSP(ACTPU) to the DLUS. After this, the ACTLUs may flow over the SSCP-PU session encapsulated in the CP-SVR pipe to activate the dependent LUs.

When activating a predefined PU, however, the DLUS node must know where to send the ACTPU. If the dependent LU is located on or downstream from some DLUR node, the DLUS must know this. As such, DLUR-supported dependent LUs will require new system-defined parameters in the DLUS node. These new parameters will be as follows:

5.4.1 New PU Definition Parameters

- ***DLUR node***

This parameter will identify the CP name of the DLUR node that supports the PU either locally or via some downstream connection. This implies that system administrators must coordinate the dependent LU definitions in a DLUS node with the CP name of the DLUR node supporting them.

- ***PU TG Descriptor***

This parameter will identify the DLUR-to-PU connectivity and is made up of signaling information (see 5.3.1, “DLUR PU Identification” on page 5-2). This parameter must also be coordinated between the DLUR and DLUS nodes:

1. it can be defined completely at the DLUS - in this case all of the information will be sent to the DLUR on the ACTPU
2. it can be defined completely at the DLUR - in this case some or all of the information can optionally be sent to the DLUS to be used for network management, input into the Configuration Services XID exit for connection verification and accounting, and PU correlation
3. it can be defined partially at the DLUS and partially at the DLUR - in this case the DLUR can optionally send the DLUR-defined signaling information (for the reasons described above) while the DLUS-defined signaling information will be sent on the ACTPU
   - if upon receipt of the ACTPU, the DLUR determines it has incomplete signaling information, it can optionally proceed with the activation (using defaults) or it can fail the activation with sense code X′086D 46mm′, where mm = the subfield key of a missing subfield
   - if upon receipt of the ACTPU, the DLUR determines it has the same signaling information subfield from both the DLUR and DLUS, but the subfields have different values, it will optionally:
     - allow the DLUS values to prevail
     - allow the DLUR values to prevail
     - reject the ACTPU with sense code X′0877 0059′
   - if upon receipt of the REQACTPU, the DLUS determines it has the same signaling information subfield from both the DLUR and DLUS, but the subfields have different values, it can optionally reject the REQACTPU with sense code X′0877 005A′

When the PU is locally supported on the DLUR node, this parameter can identify which PU image is associated with the DLUS PU definition. When the PU is located downstream from the DLUR node, this parameter can identify the TG providing connectivity between the DLUR and PU node. In the case of leased line connections, this description can provide the port and, potentially, the polling address of the downstream PU. In the case of a connection via some Shared Access Transport Facility (SATF), this parameter can provide the port, MAC and SAP address of the downstream PU. In the case of switched callout line connections, the description can specify IDBLK and IDNUM values as well as dial digits.

A new group of subfields (X′91′ - X′F0′) is being defined for both subarea Extended DIALNO (to be included in CV X′69′) and DLUR/S (to be included in CV X′46′) in order to carry the descriptions (a.k.a. signaling information).

Whatever signaling information is sent by the DLUR optionally can be echoed back by the DLUS.

When a DLUR receives an ACTPU, the search order to find the correct PU is a product option:

– Option A - PU name does not have to match

  1. PU name - this name does not have to match a DLUR-defined name for the same PU, and the DLUS-provided PU name will be accepted
  2. CP name and/or IDBLK/IDNUM - whatever is received must match its corresponding definitions at the DLUR; otherwise, the ACTPU will be rejected (since the CP names and/or IDBLK/IDNUMs do not match)

– Option B - PU name must match

  1. PU name - if the DLUS supplies a PU name and if it matches a predefined PU name, the search is over; otherwise,
  2. IDBLK/IDNUM - if the DLUR supplies an IDBLK/IDNUM and it matches a predefined IDBLK/IDNUM, then
     a. if the predefined IDBLK/IDNUM has no PU name defined with it, the search is over and the DLUS-supplied PU name will be used; otherwise,
     b. if the predefined IDBLK/IDNUM has a PU name defined with it, the ACTPU will be rejected (since the PU names do not match);

     otherwise, the ACTPU will be rejected (since the IDBLK/IDNUMs do not match)

For more information about SSCP-PU related session initiation information, please refer to 6.2, “SSCP-PU Session Activation” on page 6-8.

5.4.2 DLUS-Driven PU Activation

When a DLUR PU is activated at a DLUS node, the DLUS node will use the DLUR node specification in the PU definition to activate a CP-SVR pipe to the DLUR node. If no such pipe already exists, the DLUS node will initiate one by using the Send_Encap_Msg_TP to issue an ALLOCATE to the Receive_Encap_Msg_TP on the CP_LU of the DLUR node. This will result in a conwinner session being activated between the DLUS and DLUR node.

If the DLUR node is not available at the time the DLUS attempts to initiate a session to it, the PU activation request will fail. At some later time, when the DLUR becomes available, it may initiate a session to a server as described in 5.3, “DLUR-Initiated CP-SVR Pipe Activation” on page 5-2. Otherwise, some other product-specific mechanism will be required to re-attempt PU activation (e.g., operator-driven retry, timer-based retry, or network management exits).

5.4.3 Dynamic DLUR Node Determination

If no DLUR name is defined at the server node, then the DLUS node must first find the appropriate DLUR node and then initiate a session to it as described in 5.4.2, "DLUS-Driven PU Activation." Dynamic determination of a DLUR node by a DLUS node will be addressed at a later date by a separate architecture extension.

5.5 CP-SVR Pipe Activation

The CP-SVR pipe will actually consist of two LU 6.2 sessions between the CPs of the server and requester nodes. These sessions will be very similar to the CP-CP sessions between adjacent APPN Control Points. There will be one conwinner and one conloser session between the control points. CP-SVR sessions will, however, use a new mode name, CPSVRMGR, for reasons cited in 5.2, "CPSVRMGR Mode" on page 5-1. CP-SVR pipe activation will be triggered by the need to activate an SSCP-PU session. This can occur as the result of node activation, error recovery scenarios, XID exchange, or explicit Node Operator Facility interfaces.

The CP-SVR pipes between DLUS and DLUR nodes will exist between the Control Points of these two nodes. They are established when either the DLUS or DLUR node has something to send to the other node. This causes the DLUS or DLUR component to invoke the Send_Encap_Msg_TP to its partner. This TP will allocate a conwinner session to the partner CP name using the CPSVRMGR mode and will attach the Receive_Encap_Msg_TP at the other side.

Because the receipt of flows from one node will necessitate some response to the originator, any DLUS or DLUR node that receives an incoming conloser CPSVRMGR session must bring up a conwinner CPSVRMGR session to its partner. At this point, the CPSVRMGR (actually the encapsulated SSCP-PU) session may be accepted or rejected as described below.

5.5.1 CPSVRMGR Session Usage

Like the SNASVCMG and CPSVCMG modes already architected within the APPN specification, the DLUS and DLUR components will employ a one-way bracket conversation technique to communicate messages between the Send_Encap_Msg_TP and Receive_Encap_Msg_TP. This technique often facilitates better performance within a given implementation and will reduce the storage requirements for a DLUS node that serves many DLUR nodes. Since all products already use this technique on CPSVCMG sessions, it is assumed that the associated Attach/Detach overhead problems have already been internally resolved.

In addition, DLUS and DLUR nodes will not UNBIND CPSVRMGR sessions when no conversation is currently allocated over the session. In general, it is not recommended to BIND CPSVRMGR sessions over limited resources. Limited resources are usually indicated as such so that they are used for short-lived connections. CPSVRMGR sessions are long-lived connections which, if established over limited resources, could be very expensive. Even so, CPSVRMGR sessions are permitted over limited resources; implementations may optionally choose to prohibit CPSVRMGR sessions over limited resources. If a CPSVRMGR session is activated over a limited resource, the DLUS and DLUR nodes should ignore the limited resource characteristic of that session and not deactivate the session when the limited resource logic would normally dictate session deactivation due to inactivity. In general, DLUS and DLUR nodes should not UNBIND a CPSVRMGR session (across a limited resource or not) unless one of the session deactivation conditions specified within 5.6, "CP-SVR Pipe Deactivation" on page 5-20 is encountered.

5.5.2 Accepting Or Rejecting A CPSVRMGR Session

When a DLUR or DLUS node receives an incoming CPSVRMGR BIND from its partner node, it must determine if it accepts the session. Since CPSVRMGR sessions are established for the purpose of carrying SSCP-PU (and SSCP-LU) session traffic, nodes should accept or reject them on the basis of the SSCP to PU session. This session is initiated when either the REQACTPU or ACTPU (see 6.2, "SSCP-PU Session Activation" on page 6-8) flows over the CP-SVR pipe. For this reason, a DLUS or DLUR node receiving a CPSVRMGR BIND (assuming no other problems) should provide a positive RSP(BIND). Next, an incoming Attach for the Receive_Encap_Msg_TP will be received and either a REQACTPU or ACTPU will flow. At this point the DLUS or DLUR node can determine if it accepts the SSCP to PU session. Since a given PU can only be owned by one SSCP, the session may be rejected for this reason. System-defined parameters or other product-specific restrictions may also be used to make this determination.

One new factor in making that determination will be a DLUR/S capabilities control vector. In this initial release the control vector will carry an indication that this is a first release DLUS or DLUR (it is expected that future releases will use this CV in migration support processing). The CV (referred to as DL CAP in the flows) will be included in the FID2 Encapsulation GDS variable which carries the REQACTPU or ACTPU which triggers the activation of the CPSVRMGR session. The DLUS or DLUR receiving the CV will send back its own copy of the CV in the GDS variable carrying the corresponding RSP(REQACTPU) or RSP(ACTPU). If either the DLUS or DLUR detects a capabilities mismatch, it will reject the SSCP-PU session, and the CP-SVR pipe will be deactivated (see 5.6.2, "Abnormal CP-SVR Pipe Deactivation" on page 5-30 for more information on deactivation of the CP-SVR pipe due to DLUR/S capabilities mismatches).

If the SSCP-PU session is accepted, the accepting node must send a +RSP to the message received. This is done by issuing an ALLOCATE from the accepting node, activating the parallel conwinner session to the partner, and sending the +RSP on the session.

If the SSCP-PU session is rejected, the rejecting node will send a -RSP (to the REQACTPU or ACTPU) in the same manner. This terminates the SSCP-PU session activation. After a node has sent the -RSP, the DLUS node must check for any other active SSCP-PU/LU sessions or pending requests on the CP-SVR pipe. If no such encapsulated sessions exist, then the DLUS node must follow by sending UNBINDs for its conwinner and conloser CPSVRMGR sessions to its partner. This will cause the partner DLUR node to send RSP(UNBIND)s for these sessions which completes termination of the CP-SVR pipe. (See 5.6, "CP-SVR Pipe Deactivation" on page 5-20 for more information on deactivation of the CP-SVR pipe.)

Figure 5-1 on page 5-10 through Figure 5-3 on page 5-17 illustrate the CP-SVR pipe activation scenarios described thus far.

Notes:

1. The reader should note that the flows described below contain more detail than has yet been described. This is because of the tight relationship between SSCP-PU and CPSVRMGR session activation. For a more complete understanding of the SSCP-PU related session initiation information, please refer to 6.2, "SSCP-PU Session Activation" on page 6-8.
2. Refer to Table 3-1 on page 3-2 for an explanation of the notations used in the diagrams in this chapter.

5.5.2.1 DLUR-Initiated CP-SVR Pipe Activation

```
     SLU  PU2  DLUR  NNS   NN    NCP  SSCP  DLUS   NN    NNS   EN   PLU
               ENa   NNa   NNb              VTAM   NNc   NNd   ENb

          .XID or external operator command or internal activation signal
000       o──┐
             │Locate(Rq,Find(VTAM),Cdinit(mode=CPSVRMGR))
001          o────►o
                   │Locate(Rq,Find(VTAM))
002                └─────o─────────────────►o
                   . Locate(Rp,Found(VTAM))  │
003                o◄────────────────────────┘
             . Locate(Rp,Found(VTAM),Cdinit(RSCV))
004          o◄────┘
             │BIND(mode=CPSVRMGR)
005          └──────────────────────────────►o
             . RSP(BIND)                     │
006          o◄──────────────────────────────┘
             :Attach(Receive_Encap_Msg_TP=X'22F0F0F6').
007          :──────────────────────────────►o
             : FID2 Encap((TH,RH,REQACTPU),XID,PU TG,DL CAP,FQPCID)
008          └------------------------------►o
                                        . REQACTPU,XID,PU TG,DL CAP,FQPCID .
009                                     o◄───┘
                                        │+RSP(REQACTPU),DL CAP,FQPCID
010                                     │───►o
                   . Locate(Rq,Find(ENa))    │
011          o◄────o────────────────────│────┘
             │Locate(Rp,Found(ENa))     │
012          └──────────────────────────│───►o
             . BIND(mode=CPSVRMGR)      │    │
013          o◄─────────────────────────│────┘
             │RSP(BIND)                 │
014          └──────────────────────────│───►o
             . Attach(Receive_Encap_Msg_TP=X'22F│F0F6':
015          o◄─────────────────────────│────:
             . FID2 Encap((TH,RH,RSP(REQACTPU)),│L CAP, FQPCID) .
016          o◄-------------------------│----┘
                                        │ACTPU(PU NAME),ENa,PU TG,PU CHAR,FQPCID.
017                                     └───►o
             . Attach(Receive_Encap_Msg_TP=X'22F0F0F6')
018          o◄──────────────────────────────┘
             . FID2 Encap((TH,RH,ACTPU(PU NAME)),PU TG,PU CHAR,FQPCID)
019          o◄------------------------------┘
             │
```

Figure 5-1. DLUR-initiated CP-SVR pipe activation (DLUS known)

```
     ┌─────┐ ┌───┐ ┌────┐   ┌────┐   ┌────┐   ┌───┐   ┌────┐┌─────┐  ┌────┐   ┌────┐   ┌────┐ ┌───┐
     │SLU  │ │PU2│ │DLUR│   │NNS │   │NN  │   │NCP│   │SSCP││DLUS │  │NN  │   │NNS │   │EN  │ │PLU│
     │     │ │   │ │ ENa│   │ NNa│   │ NNb│   │   │   │    ││ VTAM│  │ NNc│   │ NNd│   │ ENb│ │   │
     └─────┘ └───┘ └────┘   └────┘   └────┘   └───┘   └────┘└─────┘  └────┘   └────┘   └────┘ └───┘
        .      . ACTPU        .        .        .        .     .        .        .        .     .
020     .      o◄────┘        .        .        .        .     .        .        .        .     .
        .      │      .       .        .        .        .     .        .        .        .     .
        .      │RSP(ACTPU)    .        .        .        .     .        .        .        .     .
021     .      └─────►o       .        .        .        .     .        .        .        .     .
        .      .      :       .        .        .        .     .        .        .        .     .
        .      .      :Attach(Receive_Encap_Msg_TP=X'22F0F0F6').        .        .        .     .
022     .      .      :───────────────────────────────────►o        .        .        .     .
        .      .      :       .        .        .        .     .        .        .        .     .
        .      .      : FID2 Encap((TH,RH,RSP(ACTPU)),FQPCID)  .        .        .        .     .
023     .      .      └-----------------------------------►o        .        .        .     .
        .      .      .       .        .        .        .     :        .        .        .     .
        .      .      .       .        .        .        . RSP(ACTPU),FQPCID     .        .     .
024     .      .      .       .        .        .        o◄----┘        .        .        .     .
        .      .      .       .        .        .        .     .        .        .        .     .
```

1. Upon receipt of an indication that a DLUR-supported PU requires activation (represented in flow 0), the DLUR node determines the DLUS node to be contacted on behalf of this PU. (See 5.3, “DLUR-Initiated CP-SVR Pipe Activation” on page 5-2.) Once the DLUS node has been determined, the DLUR node Allocates a conwinner conversation between its Send_Encap_Msg_TP and the Receive_Encap_Msg_TP (X′22F0F0F6′) at the DLUS node. Since no session using the CPSVRMGR mode between this DLUR and the DLUS already exists, the DLUR node initiates the normal APPN flow sequence to bring up such a session. This begins with the DLUR node sending a Locate Find Cdinit request to its network node server. The LU target of the session is the CP name of the DLUS node=VTAM in this case.
2. The NNS(DLUR) issues either a broadcast or directed search for the CP_LU of the DLUS=VTAM.
3. The DLUS node responds to the search request with a Locate Found reply.
4. The NNS(DLUR) uses the information in the Locate reply to calculate a route to the DLUS node and passes this back to the DLUR in the form of an RSCV.
5. The DLUR node issues a CPSVRMGR BIND to the DLUS node.
6. The DLUS node provides a BIND response.
7. Following the BIND RSP, the DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
8. Once the Attach has been sent, the DLUR must send a REQACTPU to the DLUS node. This will indicate to the DLUS node that an ACTPU is requested on behalf of some local or downstream PU. Before the REQACTPU is sent, however, the DLUR node must generate an FQPCID for this PU. This CV X′60′ will be used to uniquely identify the SSCP-PU sessions when more than one exists on a given CPSVRMGR session pipe. A FID2 Encapsulation GDS variable (X′1500′) is built, including within it
   - the FID2 PIU (TH, RH, REQACTPU RU)
   - one or more of the following used to identify the PU:
     - an XID image (CV X′81′) carrying IDBLK/IDNUM
       - an actual XID image if the PU is externally attached to the DLUR
       - a DLUR-generated XID format 0 for an internal PU or for an external PU which does not support XID
     - the PU TG control vector(s), identifying the connection between the PU and the DLUR and made up of one or more of the following:
       - a PU name (CV X′0E′) control vector
       - a signaling information (CV X′46′) control vector - this could include some or all of the signaling information defined at the DLUR
   - a DL CAP (CV X′51′) control vector, which identifies the DLUR′s capabilities
   - the FQPCID (CV X′60′)

   The FID2 Encapsulation GDS variable X′1500′ is then sent on the CPSVRMGR session to the DLUS node.
9. When the DLUS component of the DLUS node receives the FID2 Encapsulation GDS variable, it removes the encapsulation headers and passes the information within the variable to the SSCP component of the DLUS node.
10. At this point, the DLUS node makes a determination whether it accepts the SSCP-PU session. In this case, the session is accepted and a +RSP to the REQACTPU is generated and sent back to

the DLUS component of the DLUS node. Also included is another DL CAP control vector, this one identifying the DLUS′s capabilities.

11. The DLUS must now send the +RSP to the REQACTPU to the DLUR node. To do this, the DLUS Send_Encap_Msg_TP allocates a conwinner conversation to the Receive_Encap_Msg_TP at the DLUR node. This causes the normal APPN session activation flows to occur between the DLUS and DLUR node. This begins with the DLUS node initiating a Locate Find Cdinit for the CP name of the DLUR (= ENa in this case).

12. The DLUR responds to the Locate request with a Locate Found reply.

13. The DLUS node uses the information contained in the Locate reply to calculate a route to the DLUR node. It then issues a BIND to the DLUR node to establish its conwinner session with the DLUR node.

14. The DLUR node provides a BIND response.

15. The DLUS node then sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.

16. The DLUS node can now send its REQACTPU RSP to the DLUR. It encapsulates in the FID2 Encapsulation GDS variable the FID2 PIU, the DL CAP control vector, and the FQPCID (to identify the related request to the DLUR). The DLUR examines the information in the variable and also agrees to proceed.

17. Meanwhile, the DLUS SSCP has issued an ACTPU in response to the REQACTPU received from the DLUR. The ACTPU RU includes the name of the PU (which could be predefined at the DLUS, provided by the DLUR, or dynamically created by the DLUS).

18. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.

19. The DLUS component of the DLUS node then encapsulates the ACTPU FID2 PIU along with the PU TG control vector (which can optionally include the DLUS-defined signaling information and/or an echo of the DLUR-defined signaling information), PU CHAR (CV X′43′) control vector (used to define such PU characteristics as ANS=STOP|CONT), and the FQPCID of this SSCP-PU relationship with a FID2 Encapsulation GDS variable and sends the flow on its conwinner CPSVRMGR session to the DLUR.

20. The Receive_Encap_Msg_TP at the DLUR receives the encapsulated ACTPU message and removes the encapsulation headers. It uses the FQPCID information and PU TG control vector to identify the PU and its location in order to send the ACTPU to the PU itself. The DLUR removes the PU name from the ACTPU RU and stores it for network management.

    - Before storing the PU name, if the DLUR sent a PU name to the DLUS with the REQACTPU, it should check to see if the same name came back on the ACTPU. If the DLUR receives a PU name on ACTPU different from the name sent with the REQACTPU, it may accept the DLUS-provided PU name (see 5.3.1, “DLUR PU Identification” on page 5-2 for details). This can occur, for example, when the DLUS does not recognize the PU name sent with the REQACTPU; if other information was present, it can find a match based on IDBLK/IDNUM or CP name and return the name of the matching predefined PU.

21. When the PU receives the ACTPU, it will respond to the DLUR component with an ACTPU response.

22. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.

23. The DLUR component then encapsulates the RSP(ACTPU) FID2 PIU along with the FQPCID SSCP-PU session identifier in a FID2 Encapsulation GDS variable and sends it over the DLUR conwinner CPSVRMGR session to the DLUS.

24. The Receive_Encap_Msg_TP at the DLUS receives the message, removes the encapsulation headers, and passes the RSP(ACTPU) to the SSCP component of the DLUS node. The CP-SVR pipe and associated SSCP-PU session have now been activated.

5.5.2.2 DLUS-Initiated CP-SVR Pipe Activation

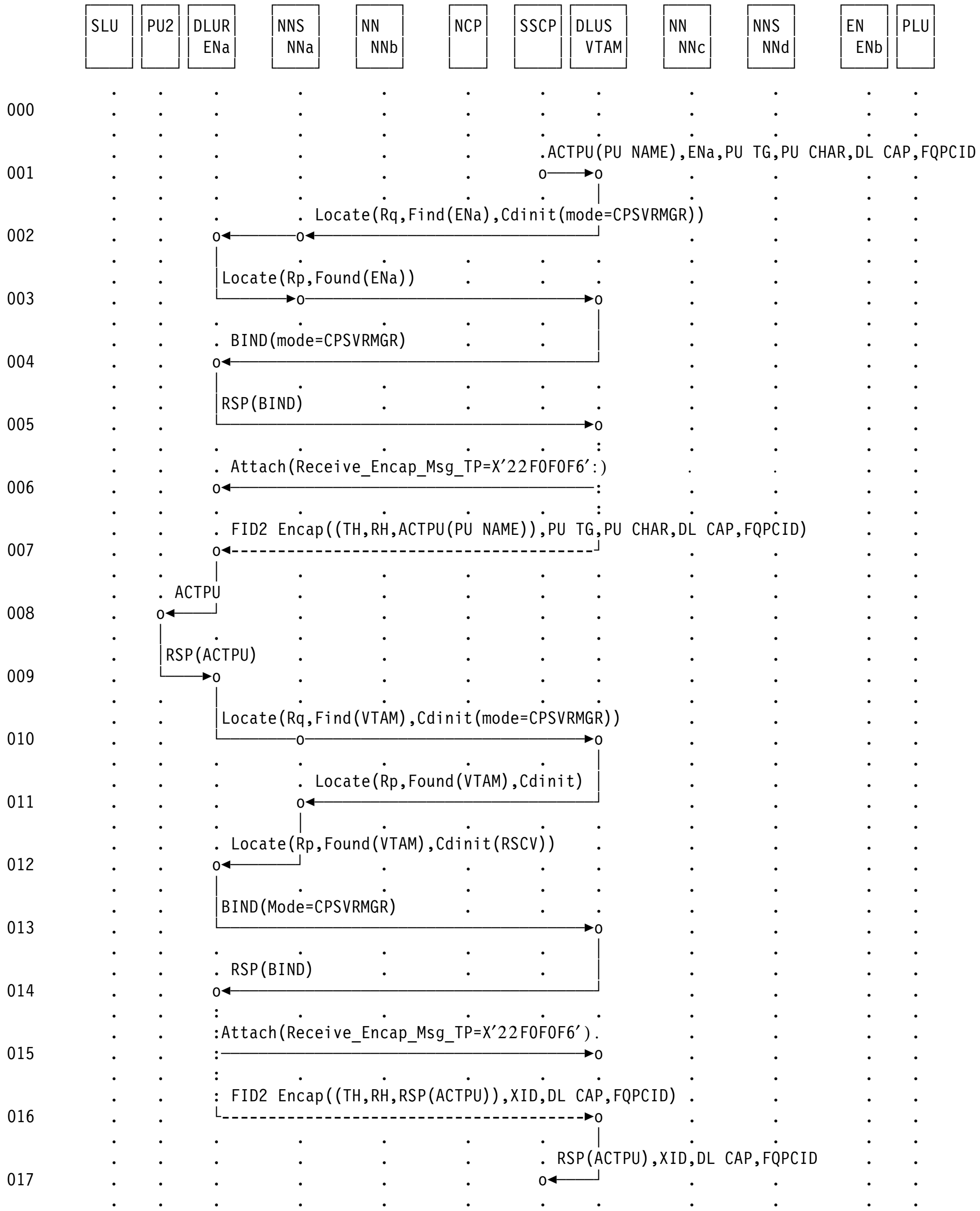


Figure 5-2. DLUS-initiated CP-SVR pipe activation

1. When a DLUS node wishes to activate a PU, the DLUS uses its own internal information (predefined or dynamically generated) to send an ACTPU signal to the DLUS component. This includes information about the PU, its connection to the DLUR, the DLUS′s capabilities, and an FQPCID.
2. The DLUS component uses this same information to determine the DLUR node associated with this PU (see 5.4.1, "New PU Definition Parameters" on page 5-6 for more details). It then allocates a conversation to the Receive_Encap_Msg_TP at the DLUR node. Since no session already exists to carry this conversation, the DLUS node initiates one starting with a Locate Find Cdinit for the CP_LU of the DLUR node = ENa.
3. The DLUR node provides a Locate reply.
4. The DLUS node uses the information in the Locate reply to calculate a route to the DLUR node and send a BIND using the CPSVRMGR mode.
5. The DLUR node provides a BIND response.
6. Once the BIND response has been received, the DLUS node sends an FMH5 to attach the Receive_Encap_Msg_TP at the DLUR node.
7. The DLUS now follows the Attach with an encapsulated ACTPU FID2 PIU. This FID2 Encapsulation GDS variable carries with it an FQPCID that has been generated by the DLUS node to be used as an SSCP-PU session identifier for the DLUR and DLUS nodes, as well as the PU TG (optionally including the DLUS-defined signaling information), PU CHAR, and DL CAP control vectors.
8. When the DLUR node receives the encapsulated message, it removes the headers, validates the capabilities, uses information with the ACTPU (such as the PU TG and PU CHAR control vectors) to determine the receiving PU, removes the PU name (storing it for network management), and sends the ACTPU to this node.
9. The PU responds to the ACTPU with an ACTPU response message to the DLUR.
10. The DLUR must now send the RSP(ACTPU) to the DLUS. In order to do this, it needs to allocate a conwinner conversation between its Send_Encap_Msg_TP and the Receive_Encap_Msg_TP at the DLUS node. Since no DLUR conwinner session already exists, the DLUR node uses the normal APPN flows to create such a session beginning with a Locate Find Cdinit being issued to the DLUS CP_LU=VTAM.
11. The DLUS node responds to the Locate with a Locate reply.
12. The NNS(DLUR) uses the information in the Locate reply to calculate a route to the DLUS node and sends this to the DLUR in the form of an RSCV attached to the Locate reply.
13. The DLUR node now issues a CPSVRMGR BIND to the DLUS node.
14. The DLUS node provides a BIND response.
15. After receiving the RSP(BIND), the DLUR node now sends an Attach for the Receive_Encap_Msg_TP at the DLUS.
16. Following the Attach, the DLUR node can send the encapsulated RSP(ACTPU) to the DLUS. It also includes the XID control vector (optional) as well as a DL CAP control vector which identifies the DLUR′s capabilities.
17. The DLUS component of the DLUS node removes the encapsulation headers, validates the capabilities, and passes the RSP(ACTPU) RU to the SSCP component of the node. The CP-SVR pipe and associated SSCP-PU session have now been activated.

5.5.2.3 SSCP-PU Session Rejection

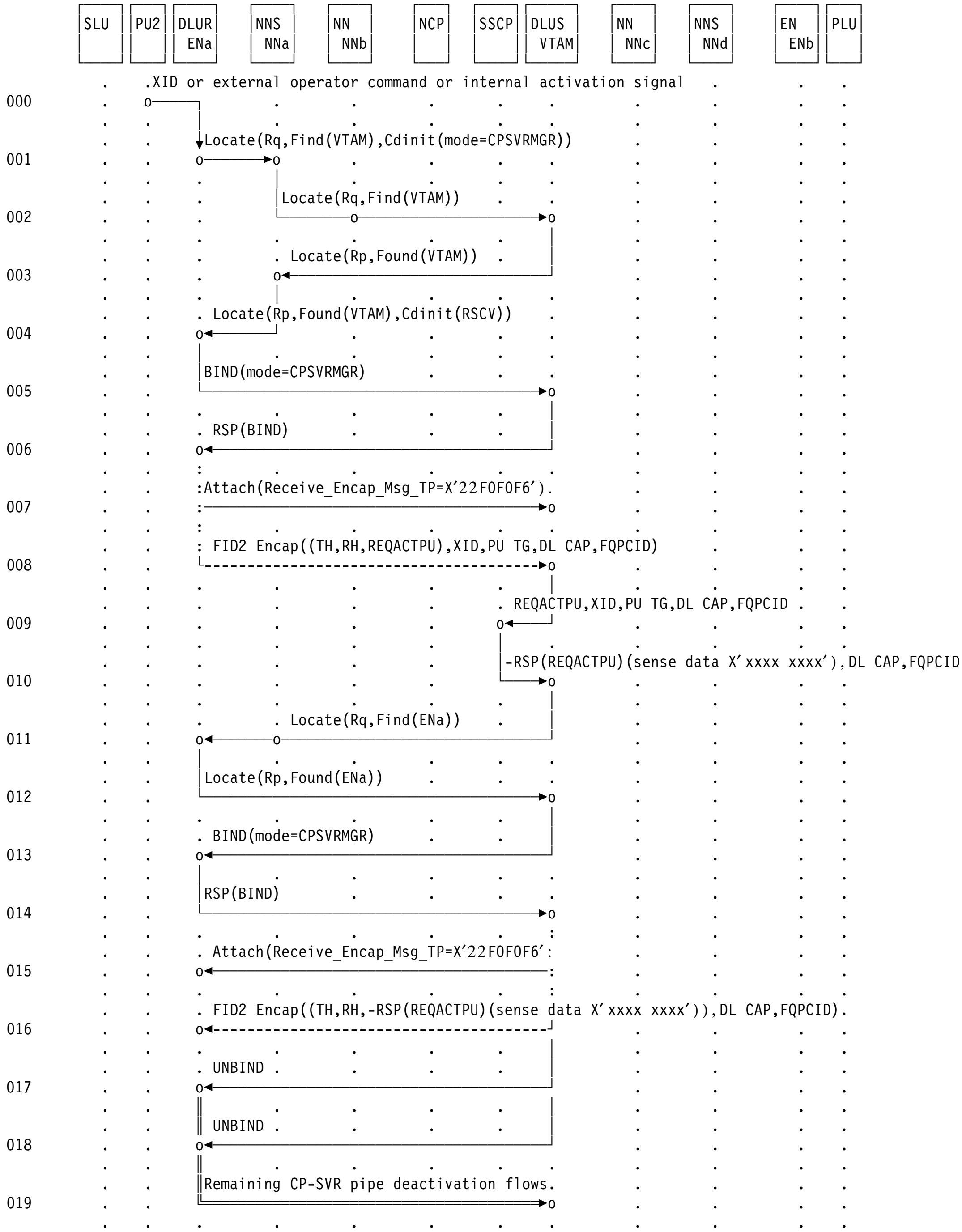


Figure 5-3. DLUR-initiated, DLUS-rejected CP-SVR pipe activation

1. Following a stimulus (flow 0), the DLUR node desires to activate a PU. To do this, the DLUR needs to request an ACTPU from some DLUS node. The DLUR uses predefined information to determine the DLUS node and allocates a conversation to the Receive_Encap_Msg_TP at that DLUS node. Since a conwinner session to that DLUS does not exist, the DLUR node uses normal APPN flows to initiate such a session. This begins with a Locate Find Cdinit for the DLUS node.

2. The NNS(DLUR) issues a broadcast or directed search for the DLUS.

3. The DLUS provides a Locate reply.

4. The NNS(DLUR) uses information in the Locate reply to calculate a route to the DLUS and appends this to the Locate reply in the form of an RSCV.

5. The DLUR node then issues a conwinner BIND to the DLUS node.

6. The DLUS node provides a RSP(BIND).

7. Once the DLUR receives a RSP(BIND), it issues an Attach to initiate the Receive_Encap_Msg_TP at the DLUS.

8. The DLUR follows the Attach with an encapsulated REQACTPU FID2 PIU for the DLUS node.

9. The Receive_Encap_Msg_TP at the DLUS node removes the encapsulation headers and passes the REQACTPU to the SSCP component.

10. The SSCP component looks at the information contained with the REQACTPU RU. It then determines that it will not accept the SSCP-PU session. This can be for a number of reasons. For example:

    - The DLUR-generated FQPCID is not unique (sense data X′083B 0002′).
    - The DLUS is out of necessary resources (X′0812 0000′).
    - The DLUS node does not recognize this PU and does not support dynamic PU activation (X′0806 0000′).
    - The IDBLK/IDNUM and/or CP name supplied with the REQACTPU RU are not unique (X′0806 0000′).
    - The Configuration Services XID Exit rejects the XID and the CV X′46′ which were included with the REQACTPU RU (X′0809 0000′).
    - The DLUS node does not have a network address pair to assign to the PU (X′0812 0007′).
    - The DLUR-specified network name is known but is a duplicate resource (X′0888 0016′).
    - The network ID in the Network Name control vector does not match the network ID of the target resource of the REQACTPU (X′0891 000A′).
    - The request was rejected due to constraints or policies specific to the receiving DLUS. The request should not be retried (X′080A 000F′).
    - The same signaling information subfield has been received in a FID2 Encapsulation GDS variable from both the DLUR and DLUS, but the subfields have different values, and the DLUS cannot reconcile the differences (X′0877 005A′).

    The SSCP generates a negative response to the REQACTPU with appropriate sense data.

11. The DLUS component must now send the RSP(REQACTPU) to the DLUR node. It, therefore, allocates a conwinner conversation to the Receive_Encap_Msg_TP at the DLUR node. Since no such session already exists to carry the conversation, the DLUS node initiates one. This begins with a Locate Find Cdinit being issued for the DLUR node.

12. The DLUR node provides a Locate reply.

13. The DLUS node uses information in the Locate reply to calculate a route to the DLUR and send a CPSVRMGR BIND to the DLUR.

14. The DLUR provides a BIND response.

15. Once the DLUS receives the RSP(BIND), it issues an FMH5 Attach for the Receive_Encap_Msg_TP at the DLUR.

16. The DLUS then sends the encapsulated RSP(REQACTPU) RU to the DLUR.

17. Since there are no SSCP-PU sessions and none outstanding on the conwinner CPSVRMGR session to the DLUR, the DLUS will deactivate the CP-SVR pipe. This is begun by sending an UNBIND on the conwinner session to its partner.

18. The DLUS also sends an UNBIND on the conloser session to its partner.

19. The DLUR node sends a RSP(UNBIND) to the DLUS for the conwinner session and another RSP(UNBIND) for the conloser session. The SSCP-PU session and CP-SVR pipe are now deactivated. For more details on CP-SVR pipe deactivation see 5.6, "CP-SVR Pipe Deactivation" on page 5-20.

5.6 CP-SVR Pipe Deactivation

There are several situations where a CP-SVR pipe will be deactivated, including normal deactivation and loss of session partner. In general, the DLUS will normally bring down the pipe, but either the DLUR or DLUS may deactivate the pipe if error or outage conditions have been detected. In every case, whichever end initiates the deactivation will deactivate both its conwinner and conloser sessions. In addition, when a DLUS or DLUR is requested to deactivate its conloser session, it should also deactivate its own conwinner session. Reactivation of the CP-SVR pipe must not begin until both conwinner and conloser sessions are deactivated (see 5.7, “Pipe And Session Failure Logic” on page 5-58 for more information on recovery of the CP-SVR pipe). LU 6.2 BIS protocols will be used to clear the pipe first during deactivation only when the UNBIND is issued without sense data.

5.6.1 Normal CP-SVR Pipe Deactivation

When the DLUS receives an encapsulated positive RSP(DACTPU) on the CP-SVR pipe, in addition to de-encapsulating the RU and passing it on to the SSCP, it will check to see if it can deactivate the CP-SVR pipe. The DLUS will deactivate the CP-SVR pipe if

- there are no active SSCP-PU sessions using the pipe and
- the pipe is not persistent (for more information about persistent CP-SVR pipes, see 5.8, “Persistent CP-SVR Pipes” on page 5-75) and
- one of the following is true
  - there are pending sessions but a DLUR/S capabilities mismatch was detected; or
  - there are no pending SSCP-PU sessions, e.g.,
    - REQACTPU was received from the DLUR, but no ACTPU has been sent by the SSCP yet
    - ACTPU was sent by the SSCP, but no RSP(ACTPU) has been received by the DLUS yet

If these conditions exist, the DLUS will deactivate its conwinner and conloser sessions by issuing an UNBIND on each session. Depending on the order and timing of the arrival of the UNBINDs, the DLUR will act as follows:

- ***second UNBIND arrives before first is processed*** - the DLUR will return a positive RSP(UNBIND) on each session, and the pipe will be deactivated;
- ***UNBIND(DLUS conloser) arrives after UNBIND(DLUS conwinner) is processed*** - the DLUR will return a positive RSP(UNBIND) on its conloser session and send an UNBIND on its own conwinner session. When the DLUS and DLUR receive the UNBINDs on this (DLUR conwinner) session, they will each return a positive RSP(UNBIND), and the pipe will be deactivated. The RSP(UNBIND)s will clean up the session connectors as they flow, so neither of them will actually arrive at their destination.
- ***UNBIND(DLUS conwinner) arrives after UNBIND(DLUS conloser) is processed*** - the DLUR will return a positive RSP(UNBIND) on its conwinner session and send an UNBIND on its own conloser session. When the DLUS and DLUR receive the UNBINDs on this (DLUR conloser) session, they will each return a positive RSP(UNBIND), and the pipe will be deactivated. The RSP(UNBIND)s will clean up the session connectors as they flow, so neither of them will actually arrive at their destination.

Figure 5-4 on page 5-22 through Figure 5-6 on page 5-27 illustrate these normal CP-SVR pipe deactivation scenarios:

- **Note** - In all other flows in this document, CP-SVR pipe deactivation will be represented by two UNBINDs being sent and a generic “Remaining CP-SVR pipe deactivation flows” to cover all possible UNBIND arrival cases. Each flow will be referred to this section for further details.

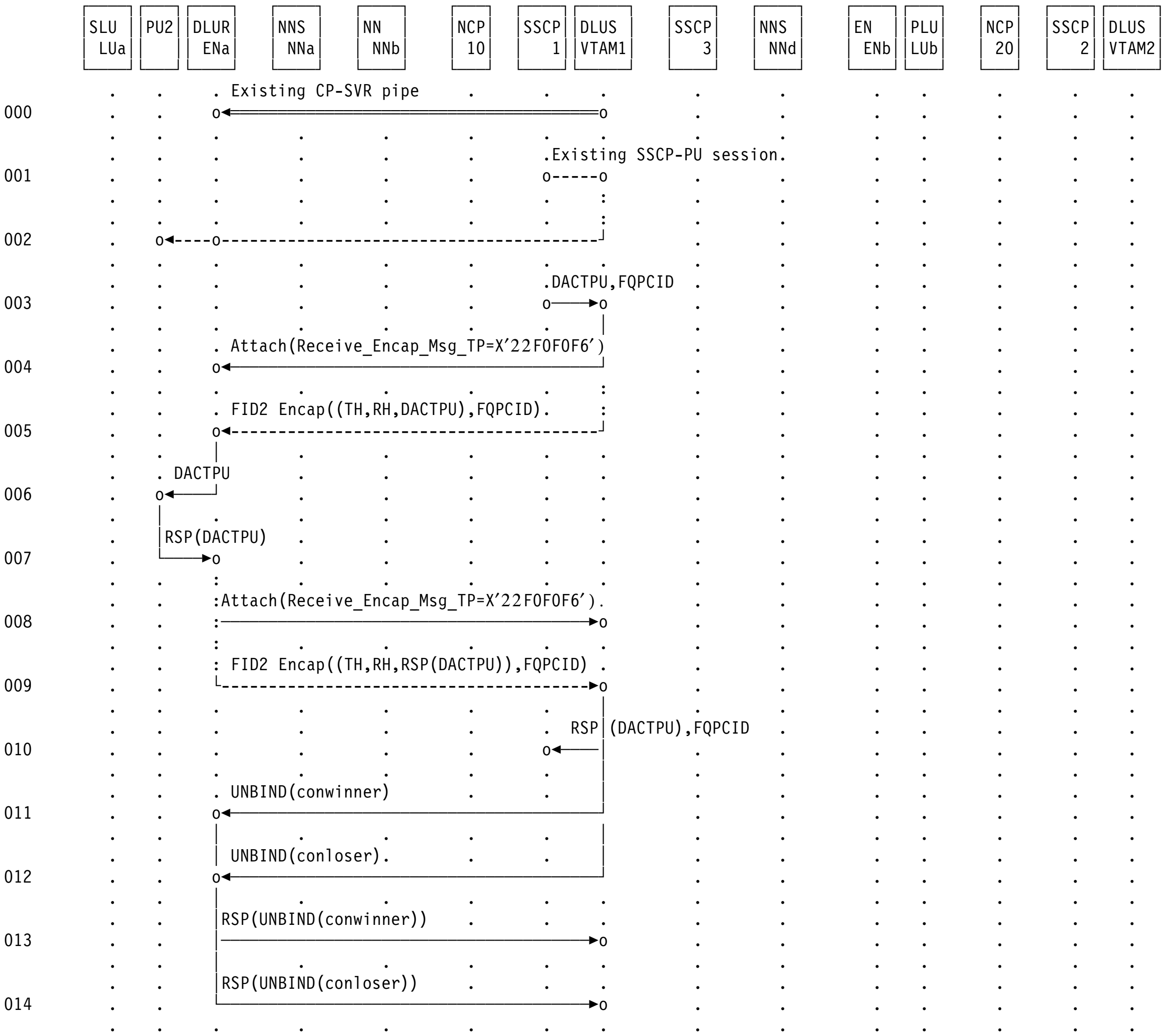


Figure 5-4. CP-SVR pipe deactivation - second UNBIND arrives before first is processed

1. There is an active session between SSCP1 and PU2. It is the only active SSCP-PU session using the CP-SVR pipe between VTAM1 and ENa.
2.
3. SSCP1 initiates normal deactivation of its session with PU2 by building a DACTPU and passing it on to VTAM1.
4. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
5. The DLUS VTAM1 encapsulates the DACTPU and forwards it to the DLUR ENa.
6. The DLUR ENa de-encapsulates the DACTPU and forwards it to PU2.
7. PU2 returns a positive RSP(DACTPU) to SSCP1 via ENa.
8. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
9. ENa encapsulates the RSP(DACTPU) and sends it to VTAM1.
10. VTAM1 de-encapsulates the RSP(DACTPU) and forwards it to SSCP1.
11. VTAM1 also determines that there are now no active or pending SSCP-PU sessions on its CP-SVR pipe to ENa, so it sends an UNBIND to ENa on VTAM1′s conwinner session to begin deactivation of the pipe.
12. VTAM1 also sends an UNBIND to ENa on VTAM1′s conloser session.
13. ENa returns a positive RSP(UNBIND) and takes down its conloser session. Since it has already received an UNBIND on its conwinner session, ENa does not send an UNBIND of its own. Upon receipt of the RSP(UNBIND), VTAM1 takes down its conwinner session.
14. ENa returns a positive RSP(UNBIND) and takes down its conwinner session. Since it has already received an UNBIND on its conloser session, ENa does not send an UNBIND of its own. Upon receipt of the RSP(UNBIND), VTAM1 takes down its conloser session.

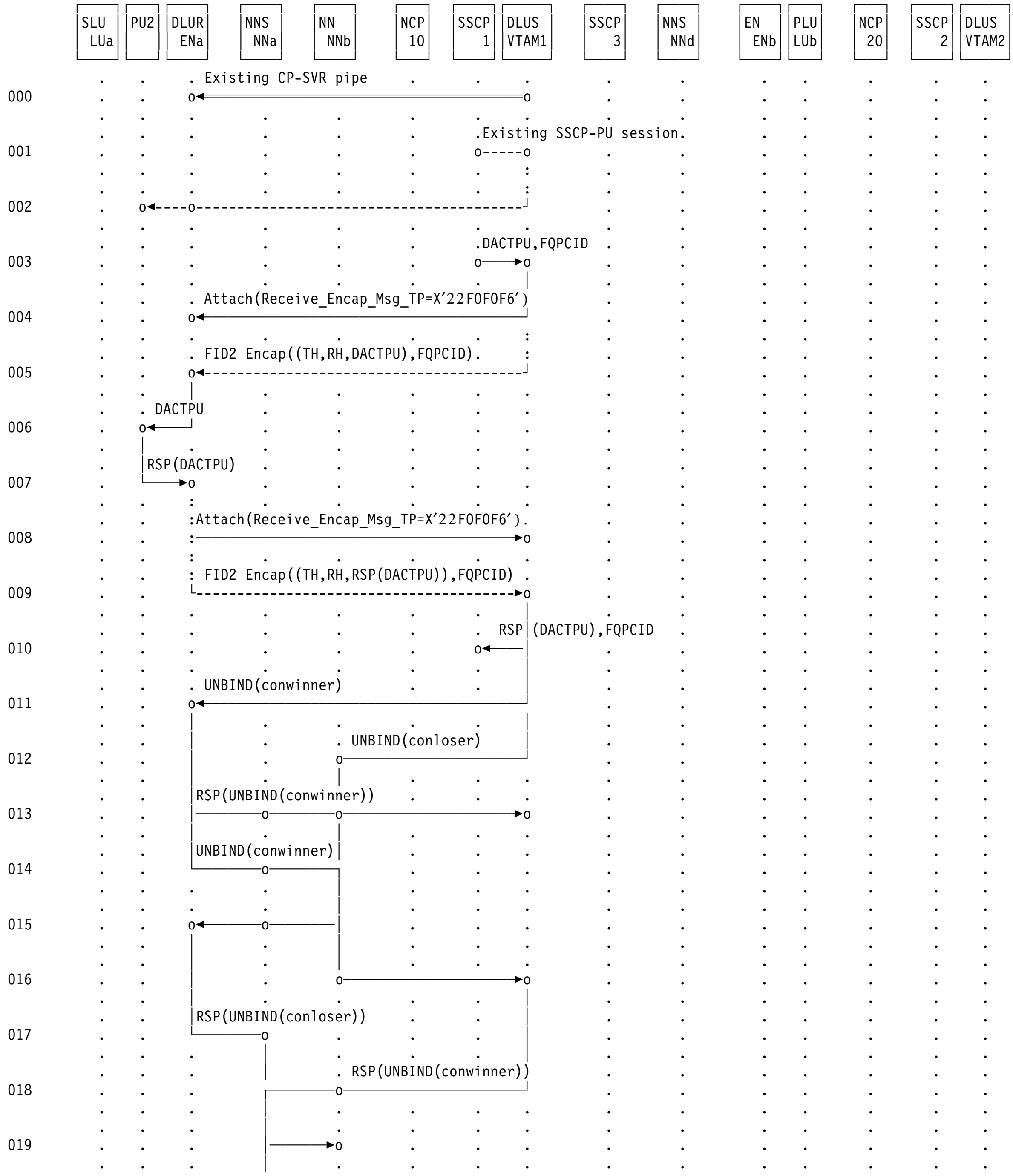


Figure 5-5. CP-SVR pipe deactivation - UNBIND(conloser) arrives after UNBIND(conwinner) is processed

```
 SLU   PU2  DLUR     NNS     NN        NCP   SSCP  DLUS      SSCP    NNS      EN    PLU    NCP   SSCP  DLUS
  LUa         ENa     NNa     NNb       10     1   VTAM1       3      NNd     ENb   LUb     20      2   VTAM2

   .     .     .       ↓       .         .     .     .         .       .        .     .      .      .     .
020   .     .     .       o       .         .     .     .         .       .        .     .      .      .     .
   .     .     .       .       .         .     .     .         .       .        .     .      .      .     .
```

1. There is an active session between SSCP1 and PU2. It is the only active SSCP-PU session using the CP-SVR pipe between VTAM1 and ENa.
2.
3. SSCP1 initiates normal deactivation of its session with PU2 by building a DACTPU and passing it on to VTAM1.
4. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
5. The DLUS VTAM1 encapsulates the DACTPU and forwards it to the DLUR ENa.
6. The DLUR ENa de-encapsulates the DACTPU and forwards it to PU2.
7. PU2 returns a positive RSP(DACTPU) to SSCP1 via ENa.
8. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
9. ENa encapsulates the RSP(DACTPU) and sends it to VTAM1.
10. VTAM1 de-encapsulates the RSP(DACTPU) and forwards it to SSCP1.
11. VTAM1 also determines that there are now no active or pending SSCP-PU sessions on its CP-SVR pipe to ENa, so it sends an UNBIND to ENa on VTAM1′s conwinner session to begin deactivation of the pipe.
12. VTAM1 also sends an UNBIND to ENa on VTAM1′s conloser session.
13. ENa returns a positive RSP(UNBIND) and takes down its conloser session. Upon receipt of the RSP(UNBIND), VTAM1 takes down its conwinner session.
14. Since it has not received an UNBIND on its conwinner session, ENa sends an UNBIND(conwinner) of its own.
15. ENa receives the UNBIND(conloser) from VTAM1.
16. VTAM1 receives the UNBIND(conwinner) from ENa.
17. ENa returns a positive RSP(UNBIND) and takes down its conwinner session.
18. VTAM1 returns a positive RSP(UNBIND) and takes down its conloser session.
19. NNb discards the RSP(UNBIND) from ENa since it tore down the session connector when it forwarded the RSP(UNBIND) from VTAM1.
20. NNa discards the RSP(UNBIND) from VTAM1 since it tore down the session connector when it forwarded the RSP(UNBIND) from ENa.

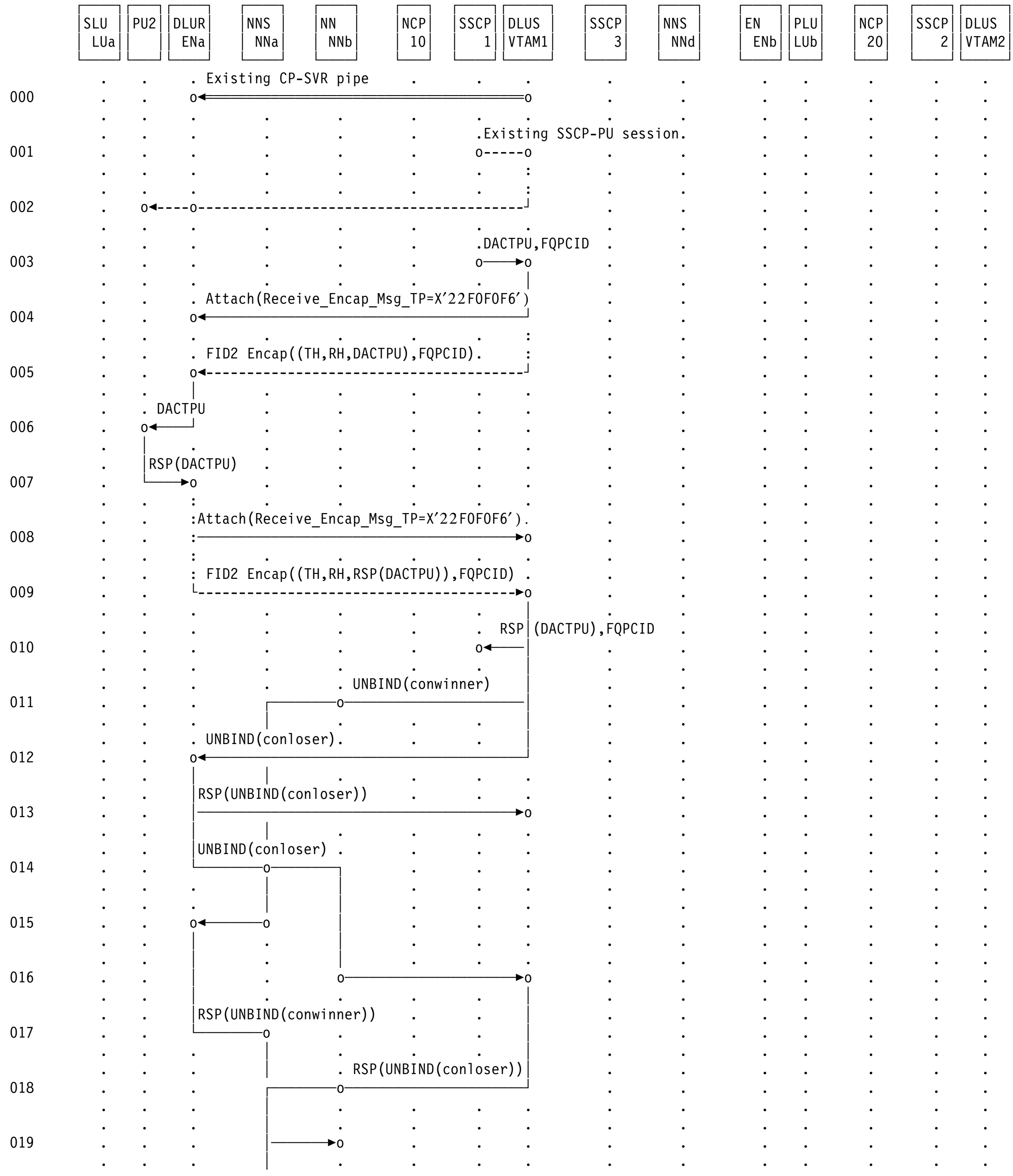


Figure 5-6. CP-SVR pipe deactivation - UNBIND(conwinner) arrives after UNBIND(conloser) is processed

```
      ┌────┐ ┌───┐ ┌────┐   ┌────┐   ┌────┐    ┌───┐  ┌────┐ ┌─────┐  ┌────┐   ┌────┐    ┌────┐ ┌───┐  ┌───┐  ┌────┐ ┌─────┐
      │SLU │ │PU2│ │DLUR│   │NNS │   │NN  │    │NCP│  │SSCP│ │DLUS │  │SSCP│   │NNS │    │EN  │ │PLU│  │NCP│  │SSCP│ │DLUS │
      │ LUa│ │   │ │ ENa│   │ NNa│   │ NNb│    │ 10│  │   1│ │VTAM1│  │   3│   │ NNd│    │ ENb│ │LUb│  │ 20│  │   2│ │VTAM2│
      └────┘ └───┘ └────┘   └────┘   └────┘    └───┘  └────┘ └─────┘  └────┘   └────┘    └────┘ └───┘  └───┘  └────┘ └─────┘
         .     .     .         ↓        .        .       .      .        .        .         .     .      .       .      .
020      .     .     .         o        .        .       .      .        .        .         .     .      .       .      .
         .     .     .         .        .        .       .      .        .        .         .     .      .       .      .
```

1. There is an active session between SSCP1 and PU2. It is the only active SSCP-PU session using the CP-SVR pipe between VTAM1 and ENa.
2.
3. SSCP1 initiates normal deactivation of its session with PU2 by building a DACTPU and passing it on to VTAM1.
4. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
5. The DLUS VTAM1 encapsulates the DACTPU and forwards it to the DLUR ENa.
6. The DLUR ENa de-encapsulates the DACTPU and forwards it to PU2.
7. PU2 returns a positive RSP(DACTPU) to SSCP1 via ENa.
8. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
9. ENa encapsulates the RSP(DACTPU) and sends it to VTAM1.
10. VTAM1 de-encapsulates the RSP(DACTPU) and forwards it to SSCP1.
11. VTAM1 also determines that there are now no active or pending SSCP-PU sessions on its CP-SVR pipe to ENa, so it sends an UNBIND to ENa on VTAM1′s conwinner session to begin deactivation of the pipe.
12. VTAM1 also sends an UNBIND to ENa on VTAM1′s conloser session.
13. ENa returns a positive RSP(UNBIND) and takes down its conwinner session. Upon receipt of the RSP(UNBIND), VTAM1 takes down its conwinner session.
14. Since it has not received an UNBIND on its conloser session, ENa sends an UNBIND(conloser) of its own.
15. ENa receives the UNBIND(conwinner) from VTAM1.
16. VTAM1 receives the UNBIND(conloser) from ENa.
17. ENa returns a positive RSP(UNBIND) and takes down its conloser session.
18. VTAM1 returns a positive RSP(UNBIND) and takes down its conwinner session.
19. NNb discards the RSP(UNBIND) from ENa since it tore down the session connector when it forwarded the RSP(UNBIND) from VTAM1.
20. NNa discards the RSP(UNBIND) from VTAM1 since it tore down the session connector when it forwarded the RSP(UNBIND) from ENa.

5.6.2 Abnormal CP-SVR Pipe Deactivation

There are other situations where the DLUS and DLUR should deactivate the CP-SVR pipe:

- If either the DLUR or DLUS should receive an UNBIND on its conloser session, it should return a positive RSP(UNBIND) on that session and, if its conwinner session is active or pending, send an UNBIND on the conwinner session.
- If an outage is detected on the CP-SVR pipe, and an UNBIND cannot be successfully routed to the session partner, then the DLUS or DLUR should deactivate and clean up both its conwinner and conloser sessions.
- Whenever a DLUR/S capabilities mismatch is detected, the CP-SVR pipe should be deactivated.

Reactivation of the CP-SVR pipe can be initiated from either the DLUS or DLUR, or both, upon completion of the deactivation of the current pipe (see 5.7, "Pipe And Session Failure Logic" on page 5-58 for more information on recovery of the CP-SVR pipe).

5.6.2.1 UNBIND-Initiated CP-SVR Pipe Deactivation

One of the DLUS requirements is to support forced deactivation of the DLUR-related sessions, i.e.,

- the CP-SVR pipe between the DLUS and the DLUR
- the SSCP-PU sessions between the SSCP in the DLUS and the PU serviced by the DLUR
- the SSCP-LU sessions between the SSCP in the DLUS and the LU serviced by the DLUR
- the LU-LU sessions involving the LU which has an SSCP-LU session with the SSCP in the DLUS

Normally, DACTLUs and DACTPUs would be sent followed by an UNBIND for the CPSVRMGR session. To speed things up, new sense data has been created for the CP-SVR pipe UNBIND which will indicate to the DLUR to deactivate some or all of the session types listed above. There are three major sense data categories:

- disruptive - all sessions (CPSVRMGR, SSCP-PU, SSCP-LU, LU-LU) will be deactivated by the DLUR for its PUs and LUs serviced by this DLUS
- nondisruptive - all sessions except active LU-LU sessions will be deactivated
- persistent - all sessions (if any) have been deactivated and now the persistent CP-SVR pipe should be deactivated (for more information about persistent CP-SVR pipes, see 5.8, "Persistent CP-SVR Pipes" on page 5-75)

5.6.2.1.1 Disruptive UNBIND-Initiated CP-SVR Pipe Deactivation: There are two disruptive session deactivation sense data, X′08A0 0007′ and X′08A0 000C′. If either sense data appears in the Extended Sense Data control vector (X′35′) in an UNBIND on the conloser CPSVRMGR session, the DLUR will:

- deactivate all SSCP-PU, SSCP-LU, and LU-LU sessions for its PUs and LUs serviced by this DLUS
- return a positive RSP(UNBIND) on the conloser session and, if its conwinner session is active or pending, send an UNBIND on the conwinner session

5.6.2.1.2 Nondisruptive UNBIND-Initiated CP-SVR Pipe Deactivation: There are four nondisruptive session deactivation sense data: X′08A0 0008′, X′08A0 0009′, X′08A0 000A′, and X′08A0 000B′. If one of these sense data appears in the CV X′35′ in an UNBIND on the conloser CPSVRMGR session, the DLUR will:

- deactivate all SSCP-PU and SSCP-LU sessions for PUs and LUs serviced by this DLUS

- deactivate all active and pending LU-LU sessions for LUs
  - serviced by this DLUS, and
  - defined with ANS=STOP
- return a positive RSP(UNBIND) on the conloser session and, if its conwinner session is active or pending, send an UNBIND on the conwinner session

The distinctions between each of these sense data are:

- X′08A0 0008′ - DLUR can immediately reinitiate CP-SVR pipe activation
- X′08A0 0009′ - DLUR can immediately reinitiate CP-SVR pipe activation / this sense data is used when a protocol violation is detected
- X′08A0 000A′ - DLUR should wait for a DLUS (need not be the same DLUS sending the UNBIND) to reinitiate CP-SVR pipe activation / DLUR-implementing products may choose to wait:
  - indefinitely, or
  - for a designated (either system- or user-defined) period of time for a CP-SVR pipe activation from a DLUS, after which the DLUR will go ahead with CP-SVR pipe activation, with either the same or a different DLUS.
- X′08A0 000B′ - DLUR can immediately reinitiate CP-SVR pipe activation, but not with the same DLUS; if no other DLUS is defined at the DLUR, then the DLUR should perform the same actions as it does for sense data X′08A0 000A′ when waiting for a DLUS to reinitiate CP-SVR pipe activation
  - At CP-SVR pipe activation, both DLURs and DLUSs will indicate support for this sense data by setting on the nondisruptive DLUS-DLUR session deactivation type X′08A0 000B′ supported indicator in the DLUR/S Capabilities (X′51′) control vector.

5.6.2.1.3 Persistent UNBIND-Initiated CP-SVR Pipe Deactivation: There is one persistent session deactivation sense data, X′08A0 000D′, sent by the DLUR in the CV X′35′ in an UNBIND on its conwinner CPSVRMGR session to deactivate a persistent CP-SVR pipe. The DLUR can deactivate a persistent pipe in this manner at any time, as long as the number of current and pending SSCP-PU sessions is zero.

5.6.2.1.4 Reactivation After UNBIND-Initiated CP-SVR Pipe Deactivation: With three exceptions, after the CP-SVR pipe deactivation has completed, both the DLUS and the DLUR can immediately activate a new CP-SVR pipe, with the same or a different DLUR/S partner. The exceptions are:

- upon receipt of sense data X′08A0 000A′, the DLUR must wait for a DLUS to activate the pipe; it is a product option as to whether the DLUR waits indefinitely or for a finite (system- or user-defined) period of time before activating a new CP-SVR pipe
- upon receipt of sense data X′08A0 000B′, the DLUR can immediately reinitiate CP-SVR pipe activation, but not with the same DLUS; if no other DLUS is defined at the DLUR, then the DLUR should perform the same actions as it does for sense data X′08A0 000A′ when waiting for a DLUS to reinitiate CP-SVR pipe activation
- upon receipt of sense data X′08A0 000C′, the DLUR can immediately reinitiate persistent CP-SVR pipe activation, but not with the same DLUS; if no other DLUS is defined at the DLUR, then the DLUR can perform the same actions as it does for sense data X′08A0 000A′ when waiting for a DLUS to reinitiate CP-SVR pipe activation or it can attempt to activate a nonpersistent CP-SVR pipe with the same DLUS if it has a PU needing service

5.6.2.2 DLUR/S Capabilities Mismatches

A CP-SVR pipe should only be established between a DLUS and a DLUR with compatible capabilities. Exchanging the DLUR/S Capabilities control vector (CV X′51′) assists a node in determining whether to allow a CP-SVR pipe to be activated between itself and a partner node.

- To ensure detection of a capabilities mismatch, the CV X′51′ should be examined before the RU included in the FID2 Encapsulation GDS variable is processed.

Whenever a mismatch in DLUR/S capabilities is detected, the CP-SVR pipe should be deactivated. Whichever node detects the mismatch will initiate the pipe deactivation.

5.6.2.2.1 DLUS-Detected Capabilities Mismatch (DLUS-DLUR Pipe): Figure 5-7 on page 5-33 through Figure 5-8 on page 5-35 illustrate examples of DLUR/S capabilities mismatches detected by the DLUS during activation of a CP-SVR pipe between a DLUS and a DLUR.

```
      ┌───┐ ┌───┐ ┌────┐   ┌────┐   ┌────┐   ┌───┐  ┌────┐┌────┐   ┌────┐   ┌────┐    ┌────┐┌───┐
      │SLU│ │PU2│ │DLUR│   │NNS │   │NN  │   │NCP│  │SSCP││DLUS│   │NN  │   │NNS │    │EN  ││PLU│
      │   │ │   │ │ ENa│   │ NNa│   │ NNb│   │   │  │    ││VTAM│   │ NNc│   │ NNd│    │ ENb││   │
      └───┘ └───┘ └────┘   └────┘   └────┘   └───┘  └────┘└────┘   └────┘   └────┘    └────┘└───┘
        .    .XID or external operator command or internal activation signal   .          .    .
000     .    o─────┐     .        .        .       .     .          .        .          .    .
        .    .     │     .        .        .       .     .          .        .          .    .
        .    .     ▼Locate(Rq,Find(VTAM),Cdinit(mode=CPSVRMGR))     .        .          .    .
001     .    .     o────────►o    .        .       .     .          .        .          .    .
        .    .     .         │    .        .       .     .          .        .          .    .
        .    .     .         │Locate(Rq,Find(VTAM))    .     .      .        .          .    .
002     .    .     .         └────────o───────────────────►o        .        .          .    .
        .    .     .         .        .        .       .   │        .        .          .    .
        .    .     .         . Locate(Rp,Found(VTAM))  .   │        .        .          .    .
003     .    .     .         o◄────────────────────────────┘        .        .          .    .
        .    .     .         │        .        .       .     .      .        .          .    .
        .    .     . Locate(Rp,Found(VTAM),Cdinit(RSCV))     .      .        .          .    .
004     .    .     o◄────────┘        .        .       .     .      .        .          .    .
        .    .     │         .        .        .       .     .      .        .          .    .
        .    .     │BIND(mode=CPSVRMGR)        .       .     .      .        .          .    .
005     .    .     └───────────────────────────────────────►o       .        .          .    .
        .    .     .         .        .        .       .   │        .        .          .    .
        .    .     . RSP(BIND)        .        .       .   │        .        .          .    .
006     .    .     o◄──────────────────────────────────────┘        .        .          .    .
        .    .     :         .        .        .       .     .      .        .          .    .
        .    .     :Attach(Receive_Encap_Msg_TP=X'22F0F0F6').       .        .          .    .
007     .    .     :──────────────────────────────────────►o        .        .          .    .
        .    .     :         .        .        .       .     .      .        .          .    .
        .    .     : FID2 Encap((TH,RH,REQACTPU),XID,PU TG,DL CAP,FQPCID)    .          .    .
008     .    .     └- - - - - - - - - - - - - - - - - - - ►X        .        .          .    .
        .    .     .         .        .        .       .   │        .        .          .    .
        .    .     . UNBIND  .        .        .       .   │        .        .          .    .
009     .    .     o◄──────────────────────────────────────┘        .        .          .    .
        .    .     ║         .        .        .       .     .      .        .          .    .
        .    .     ║Remaining CP-SVR pipe deactivation flows.       .        .          .    .
010     .    .     ╚══════════════════════════════════════►o        .        .          .    .
        .    .     .         .        .        .       .     .      .        .          .    .
```

Figure 5-7. DLUR-initiated, DLUS-rejected CP-SVR pipe activation

1. Following a stimulus (flow 0), the DLUR node desires to activate a PU. To do this, it needs to request an ACTPU from some DLUS node. It uses predefined information to determine the DLUS node and allocates a conversation to the Receive_Encap_Msg_TP at that DLUS node. Since no conwinner session to that DLUS already exists, the DLUR node uses normal APPN flows to initiate such a session. This begins with a Locate Find Cdinit for the DLUS node.
2. The NNS(DLUR) issues a broadcast or directed search for the DLUS.
3. The DLUS provides a locate reply.
4. The NNS(DLUR) uses information in the Locate reply to calculate a route to the DLUS and appends this to the Locate reply in the form of an RSCV.
5. The DLUR node then issues a conwinner BIND to the DLUS node.
6. The DLUS node provides a RSP(BIND).
7. Once the DLUR receives a RSP(BIND), it issues an Attach to initiate the Receive_Encap_Msg_TP at the DLUS.
8. The DLUR follows the Attach with an encapsulated REQACTPU FID2 PIU for the DLUS node.
9. The Receive_Encap_Msg_TP at the DLUS node removes the encapsulation headers. The DLUS component looks at the DLUR′s capabilities and determines that they are incompatible with those of the DLUS. Since a DLUR/S capabilities mismatch was detected, the DLUS will deactivate the CP-SVR pipe. This is begun by sending an UNBIND on the conloser session to its partner, including sense data X′088E 0008′ in the CV X′35′.
10. The DLUR node sends a RSP(UNBIND) to the DLUS for its conwinner session. The CP-SVR pipe is now deactivated. For more details on CP-SVR pipe deactivation see 5.6, “CP-SVR Pipe Deactivation” on page 5-20.

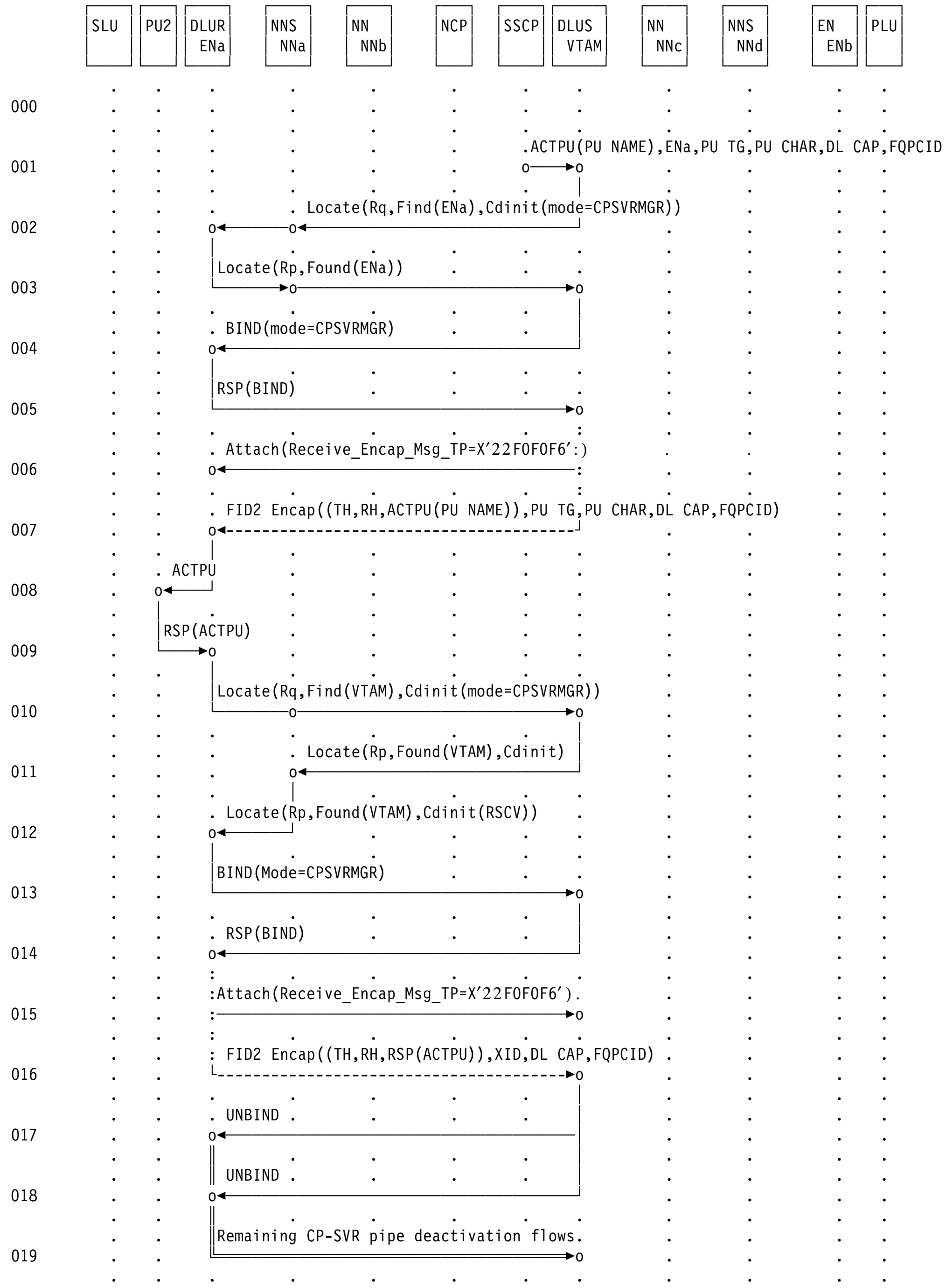


Figure 5-8. DLUS-initiated, DLUS-rejected CP-SVR pipe activation

1. When a DLUS node wishes to activate a PU, it uses its internal information (predefined or dynamically generated) to send an ACTPU signal to the DLUS component. This includes information about the PU, its connection to the DLUR, the DLUS′s capabilities, and an FQPCID.
2. The DLUS component uses this same information to determine the DLUR node associated with this PU (see 5.4.1, "New PU Definition Parameters" on page 5-6 for more details). It then allocates a conversation to the Receive_Encap_Msg_TP at the DLUR node. Since no session already exists to carry this conversation, the DLUS node initiates one starting with a Locate Find Cdinit for the CP_LU of the DLUR node = ENa.
3. The DLUR node provides a Locate reply.
4. The DLUS node uses the information in the Locate reply to calculate a route to the DLUR node and send a BIND using the CPSVRMGR mode.
5. The DLUR node provides a BIND response.
6. Once the BIND response has been received, the DLUS node sends an FMH5 to attach the Receive_Encap_Msg_TP at the DLUR node.
7. The DLUS now follows the attach with an encapsulated ACTPU FID2 PIU. This FID2 Encapsulation GDS variable carries with it an FQPCID that has been generated by the DLUS node to be used as an SSCP-PU session identifier for the DLUR and DLUS nodes, as well as the PU TG and DL CAP control vectors.
8. When the DLUR node receives the encapsulated message it removes the headers, validates the capabilities, uses information with the ACTPU (such as the PU TG and PU CHAR control vectors) to determine the receiving PU, removes the PU name (storing it for network management), and sends the ACTPU to this node.
9. The PU responds to the ACTPU with an ACTPU response message to the DLUR.
10. The DLUR must now send the RSP(ACTPU) to the DLUS. In order to do this, it needs to allocate a conwinner conversation between its Send_Encap_Msg_TP and the Receive_Encap_Msg_TP at the DLUS node. Since no DLUR conwinner session already exists, the DLUR node uses the normal APPN flows to create such a session beginning with a Locate Find Cdinit being issued to the DLUS CP_LU=VTAM.
11. The DLUS node responds to the Locate with a Locate reply.
12. The NNS(DLUR) uses the information in the Locate reply to calculate a route to the DLUS node and sends this to the DLUR in the form of an RSCV attached to the Locate reply.
13. The DLUR node now issues a CPSVRMGR BIND to the DLUS node.
14. The DLUS node provides a BIND response.
15. After receiving the RSP(BIND), the DLUR node now sends an Attach for the Receive_Encap_Msg_TP at the DLUS.
16. Following the Attach, the DLUR node can send the encapsulated RSP(ACTPU) to the DLUS. It also includes the XID control vector as well as a DL CAP control vector identifying the DLUR′s capabilities.
17. The DLUS component of the DLUS node removes the encapsulation headers and validates the capabilities. In this case, the DLUS examines the DLUR′s capabilities and finds a mismatch with its own. Therefore, the DLUS will deactivate the CP-SVR pipe. This is begun by sending an UNBIND, including sense data X′088E 0008′ in CV X′35′, on the conwinner session to its partner.
18. The DLUS also sends an UNBIND, including sense data X′088E 0008′ in the CV X′35′, on the conloser session to its partner.

19. The DLUR node sends a RSP(UNBIND) to the DLUS for the conwinner session and another RSP(UNBIND) for the conloser session. The SSCP-PU session and CP-SVR pipe are now deactivated. For more details on CP-SVR pipe deactivation see 5.6, “CP-SVR Pipe Deactivation” on page 5-20.

5.6.2.2.2 DLUR-Detected Capabilities Mismatch (DLUS-DLUR Pipe): Figure 5-9 on page 5-38 through Figure 5-10 on page 5-42 illustrate examples of DLUR/S capabilities mismatches detected by the DLUR during activation of a CP-SVR pipe between a DLUS and a DLUR.

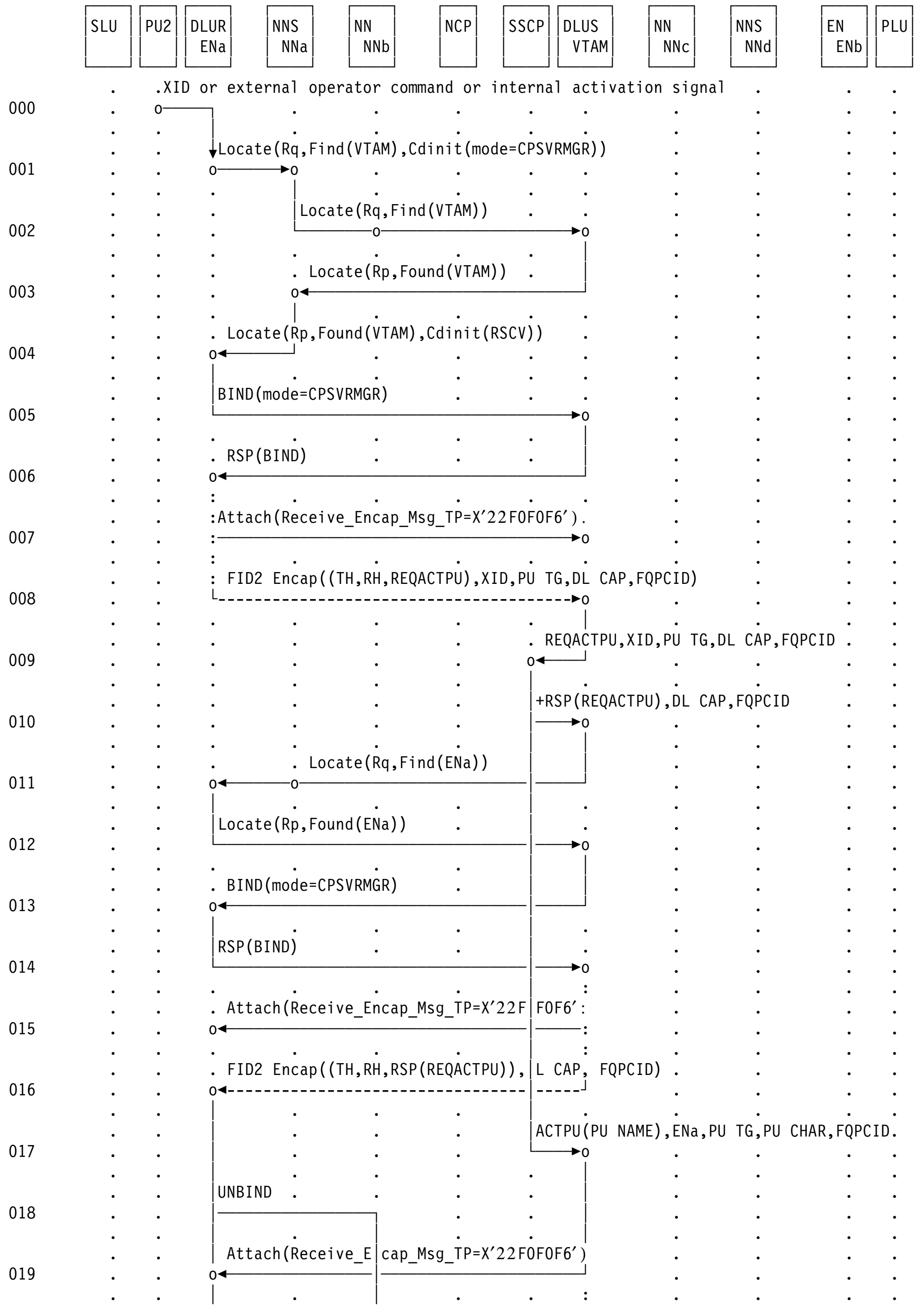


Figure 5-9. DLUR-initiated, DLUR-rejected CP-SVR pipe activation

```
          ┌─────┐ ┌───┐ ┌────┐    ┌────┐    ┌────┐     ┌───┐   ┌────┐┌────┐    ┌────┐    ┌────┐     ┌────┐┌───┐
          │SLU  │ │PU2│ │DLUR│    │NNS │    │NN  │     │NCP│   │SSCP││DLUS│    │NN  │    │NNS │     │EN  ││PLU│
          │     │ │   │ │ ENa│    │ NNa│    │ NNb│     │   │   │    ││VTAM│    │ NNc│    │ NNd│     │ ENb││   │
          └─────┘ └───┘ └────┘    └────┘    └────┘     └───┘   └────┘└────┘    └────┘    └────┘     └────┘└───┘
             .     .    │ FID2 Encap((TH,R│,ACTPU(PU NAME)),PU TG,PU CHAR,FQPCID)   .          .     .
020          .     .    o◄----------------│-----------------------┘          .          .          .     .
             .     .    │         .       │         .       .     .          .          .          .     .
             .     .    │         .       │         .       .     .          .          .          .     .
021          .     .    │         .       └───────────────────────►o         .          .          .     .
             .     .    │         .       .         .       .     .          .          .          .     .
             .     .    │UNBIND   .       .         .       .     .          .          .          .     .
022          .     .    └─────────────────────────────────────────►o         .          .          .     .
             .     .    .         .       .         .       .     ║          .          .          .     .
             .     .    . Remaining CP-SVR pipe deactivation flows           .          .          .     .
023          .     .    o◄════════════════════════════════════════╝          .          .          .     .
             .     .    .         .       .         .       .                .          .          .     .
```

1. Upon receipt of an indication that a DLUR-supported PU requires activation (represented in flow 0), the DLUR node determines the DLUS node to be contacted on behalf of this PU (see 5.3, “DLUR-Initiated CP-SVR Pipe Activation” on page 5-2). Once the DLUS node has been determined, the DLUR node allocates a conwinner conversation between its Send_Encap_Msg_TP and the Receive_Encap_Msg_TP (X′22F0F0F6′) at the DLUS node. Since no session using the CPSVRMGR mode between this DLUR and the DLUS already exists, the DLUR node initiates the normal APPN flow sequence to bring up such a session. This begins with the DLUR node sending a Locate Find Cdinit request to its network node server. The LU target of the session is the CP name of the DLUS node=VTAM in this case.
2. The NNS(DLUR) issues either a broadcast or directed search for the CP_LU of the DLUS=VTAM.
3. The DLUS node responds to the search request with a Locate Found reply.
4. The NNS(DLUR) uses the information in the Locate reply to calculate a route to the DLUS node and passes this back to the DLUR in the form of an RSCV.
5. The DLUR node issues a CPSVRMGR BIND to the DLUS node.
6. The DLUS node provides a BIND response.
7. Following the RSP(BIND), the DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
8. Once the Attach has been sent, the DLUR must send a REQACTPU to the DLUS node. This will indicate to the DLUS node that an ACTPU is requested on behalf of some local or downstream PU. Before the REQACTPU is sent, however, the DLUR node must generate an FQPCID for this PU. This CV X′60′ will be used to uniquely identify the SSCP-PU sessions when more than one exists on a given CP-SVR pipe. A FID2 Encapsulation GDS variable (X′1500′) is built, including within it the FID2 PIU, XID image, PU TG CV, DL CAP CV, and FQPCID. The GDS variable is then sent on the conwinner CPSVRMGR session to the DLUS node.
9. When the DLUS component of the DLUS node receives the FID2 Encapsulation GDS variable, it removes the encapsulation headers and passes the information within the variable to the SSCP component of the DLUS node.
10. At this point, the DLUS node makes a determination whether it accepts the SSCP-PU session. In this case, the session is accepted and a +RSP to the REQACTPU is generated and sent back to the DLUS component of the DLUS node. Also included is another DL CAP control vector, this one identifying the DLUS′s capabilities.
11. The DLUS must now send the +RSP to the REQACTPU to the DLUR node. To do this, the DLUS Send_Encap_Msg_TP allocates a conwinner conversation to the Receive_Encap_Msg_TP at the DLUR node. This causes the normal APPN session activation flows to occur between the DLUS and DLUR nodes. This begins with the DLUS node initiating a Locate Find Cdinit for the CP name of the DLUR = ENa in this case.
12. The DLUR responds to the Locate request with a Locate Found reply.
13. The DLUS node uses the information contained in the Locate reply to calculate a route to the DLUR node. It then issues a BIND to the DLUR node to establish its conwinner session with the DLUR node.
14. The DLUR node provides a BIND response.
15. The DLUS node then sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
16. The DLUS node can now send its RSP(REQACTPU) to the DLUR. The DLUS encapsulates in the FID2 Encapsulation GDS variable the FID2 PIU, the DL CAP control vector, and the FQPCID (to iden-

tify the related request to the DLUR). In this case, the DLUR examines the DLUS′s capabilities and finds a mismatch with its own.

17. Meanwhile, the DLUS SSCP has issued an ACTPU in response to the REQACTPU received from the DLUR.
18. Since a DLUR/S capabilities mismatch was detected, the DLUR will deactivate the CP-SVR pipe. This is begun by sending an UNBIND on the conwinner session to its partner, including sense data X′088E 0009′ in CV X′35′.
19. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
20. The DLUS component of the DLUS node then encapsulates the ACTPU FID2 PIU along with the PU TG control vector, PU CHAR control vector (used to define such PU characteristics as ANS=STOP|CONT), and the FQPCID of this SSCP-PU relationship with a FID2 Encapsulation GDS variable and sends the flow on its conwinner CPSVRMGR session to the DLUR.
21. The Receive_Encap_Msg_TP at the DLUR receives the encapsulated ACTPU message and removes the encapsulation headers. It uses the FQPCID information and PU TG control vector to identify the PU and its location. In this case, since it is in the process of deactivating the CP-SVR pipe due to a capabilities mismatch, it will discard the ACTPU.
22. The DLUR also sends an UNBIND on the conloser session to its partner, including sense data X′088E 0009′ in CV X′35′.
23. The DLUS node sends a RSP(UNBIND) to the DLUR for the conwinner session and another RSP(UNBIND) for the conloser session. The SSCP-PU session and CP-SVR pipe are now deactivated. For more details on CP-SVR pipe deactivation see 5.6, “CP-SVR Pipe Deactivation” on page 5-20.

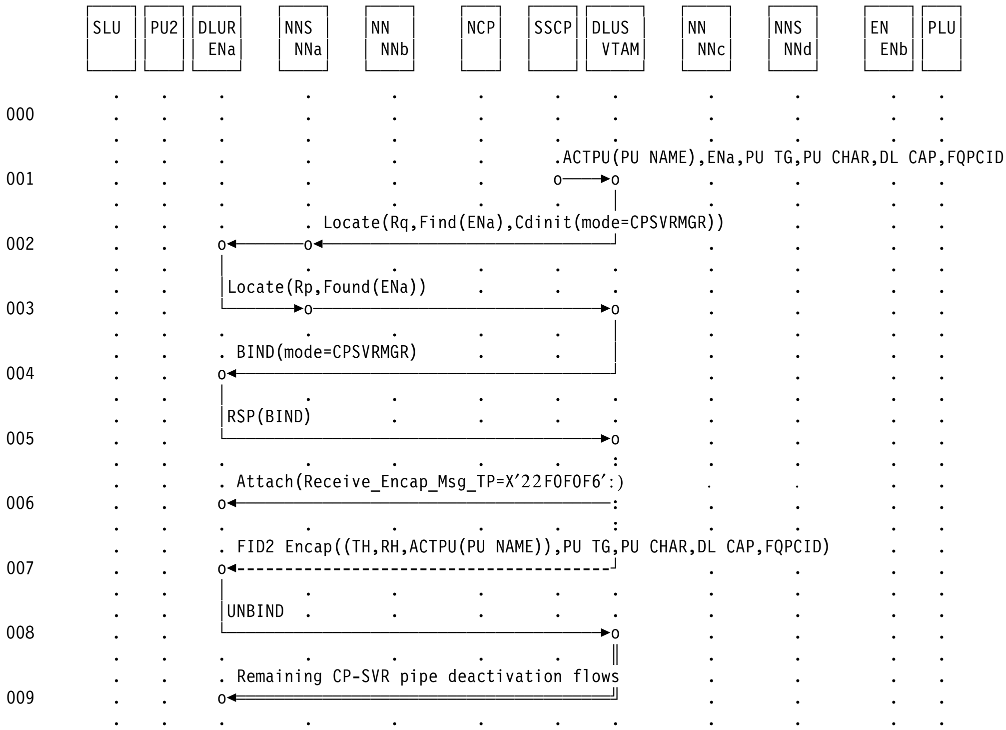


Figure 5-10. DLUS-initiated, DLUR-rejected CP-SVR pipe activation

1. When a DLUS node wishes to activate a PU, the DLUS uses its own internal information (predefined or dynamically generated) to send an ACTPU signal to the DLUS component. This includes information about the PU, its connection to the DLUR, the DLUS′s capabilities, and an FQPCID.

2. The DLUS component uses this same information to determine the DLUR node associated with this PU (see 5.4.1, "New PU Definition Parameters" on page 5-6 for more details). It then allocates a conversation to the Receive_Encap_Msg_TP at the DLUR node. Since no session already exists to carry this conversation, the DLUS node initiates one starting with a Locate Find Cdinit for the CP_LU of the DLUR node = ENa.

3. The DLUR node provides a Locate reply.

4. The DLUS node uses the information in the Locate reply to calculate a route to the DLUR node and send a BIND using the CPSVRMGR mode.

5. The DLUR node provides a BIND response.

6. Once the BIND response has been received, the DLUS node sends an FMH5 to attach the Receive_Encap_Msg_TP at the DLUR node.

7. The DLUS now follows the Attach with an encapsulated ACTPU FID2 PIU. This FID2 Encapsulation GDS variable carries with it an FQPCID that has been generated by the DLUS node to be used as an SSCP-PU session identifier for the DLUR and DLUS nodes, as well as the PU TG and DL CAP control vectors.

8. When the DLUR node receives the encapsulated message, it removes the headers and validates the capabilities. In this case, it examines the DLUS′s capabilities and finds a mismatch with its own. The DLUR must deactivate the CP-SVR pipe. This is begun by sending an UNBIND on the conloser session to its partner, including sense data X′088E 0009′ in CV X′35′.

9. The DLUS node sends a RSP(UNBIND) to the DLUS for its conwinner session. The CP-SVR pipe is now deactivated. For more details on CP-SVR pipe deactivation see 5.6, "CP-SVR Pipe Deactivation" on page 5-20.

5.6.2.2.3 DLUS-Detected Capabilities Mismatch (DLUS-DLUS Pipe): Figure 5-11 on page 5-44 illustrates an example of a DLUR/S capabilities mismatch detected by the DLUS during activation of a CP-SVR pipe between a DLUS and a DLUS.

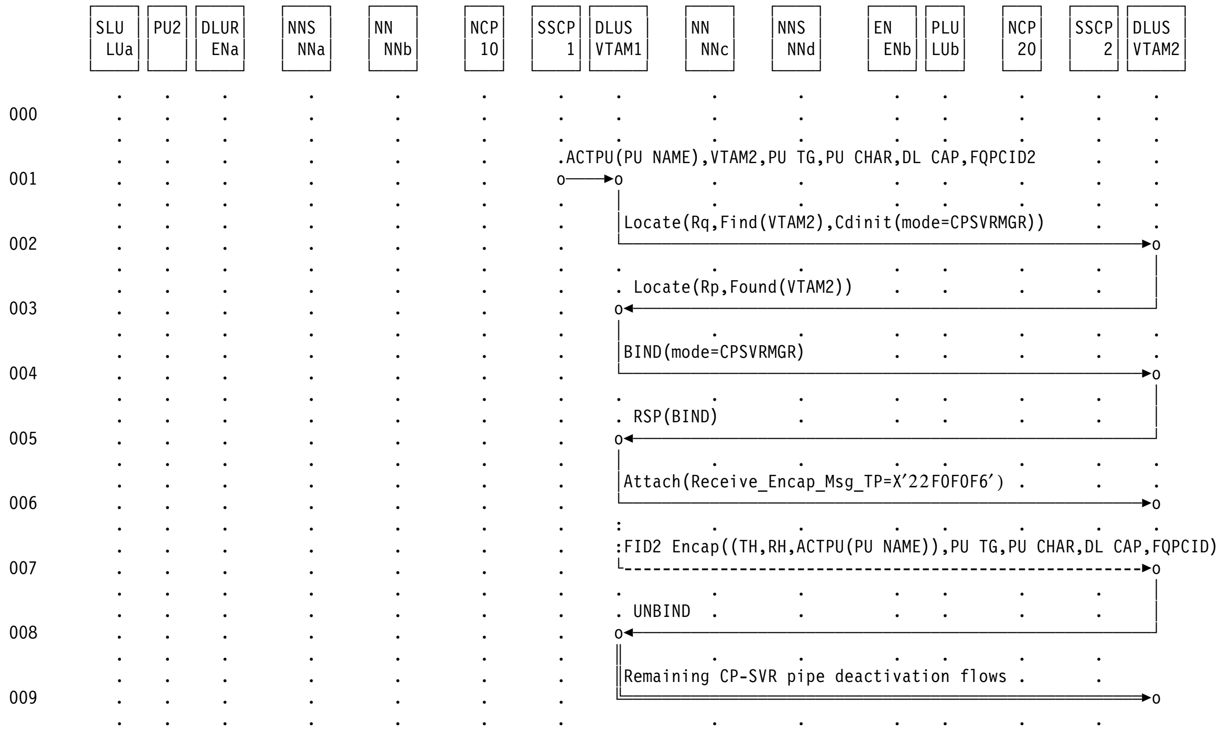


Figure 5-11. DLUS-initiated, DLUS-rejected CP-SVR pipe activation (DLUS-DLUS pipe)

1. When DLUS node VTAM1 wishes to activate a PU, it uses its internal information (predefined or dynamically generated) to send an ACTPU signal to the DLUS component. This includes information about the PU, its connection to the DLUR, the DLUS′s capabilities, and an FQPCID.

2. The DLUS component uses this same information to determine the DLUR node associated with this PU (see 5.4.1, "New PU Definition Parameters" on page 5-6 for more details). In this case the DLUR node was erroneously predefined as VTAM2. VTAM1 then allocates a conversation to the Receive_Encap_Msg_TP at VTAM2. Since no session already exists to carry this conversation, VTAM1 initiates one starting with a Locate Find Cdinit for the CP_LU of the DLUR node = VTAM2.

3. VTAM2 provides a Locate reply.

4. VTAM1 uses the information in the Locate reply to calculate a route to VTAM2 and send a BIND using the CPSVRMGR mode.

5. VTAM2 provides a BIND response.

6. Once the BIND response has been received, VTAM1 sends an FMH5 to attach the Receive_Encap_Msg_TP at VTAM2.

7. VTAM1 now follows the Attach with an encapsulated ACTPU FID2 PIU. This FID2 Encapsulation GDS variable carries with it an FQPCID that has been generated by VTAM1 to be used as an SSCP-PU session identifier for the DLUR and DLUS nodes, as well as the PU TG, PU CHAR, and DL CAP control vectors.

8. When VTAM2 receives the encapsulated message, it removes the headers and validates the capabilities. It recognizes a DLUR/S capabilities mismatch and deactivates the CP-SVR pipe. This is begun by sending an UNBIND on the conloser session to its partner, including sense data X′088E 0008′ in the CV X′35′.

9. VTAM1 sends a RSP(UNBIND) to VTAM2 for its conwinner session. The CP-SVR pipe is now deactivated. For more details on CP-SVR pipe deactivation see 5.6, "CP-SVR Pipe Deactivation" on page 5-20.

5.6.2.2.4 DLUR-Detected Capabilities Mismatch (DLUR-DLUR Pipe): Figure 5-12 on page 5-46 illustrates an example of a DLUR/S capabilities mismatch detected by the DLUR during activation of a CP-SVR pipe between a DLUR and a DLUR.

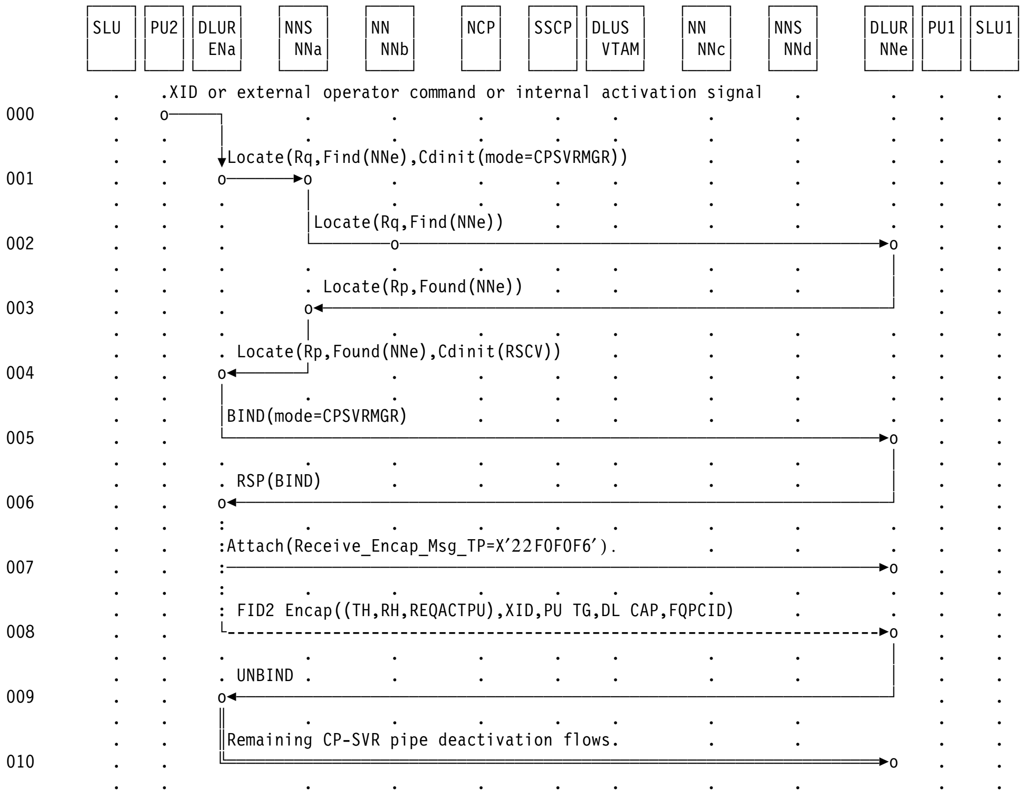


Figure 5-12. DLUR-initiated, DLUR-rejected CP-SVR pipe activation (DLUR-DLUR pipe)

1. Upon receipt of some indication that a DLUR-supported PU requires activation (represented in flow 0), the DLUR node determines the DLUS node to be contacted on behalf of this PU (see 5.3, “DLUR-Initiated CP-SVR Pipe Activation” on page 5-2). In this case, at ENa the DLUS name was erroneously predefined as NNe. ENa allocates a conwinner conversation between its Send_Encap_Msg_TP and the Receive_Encap_Msg_TP (X′22F0F0F6′) at NNe. Since no session using the CPSVRMGR mode between ENa and NNe already exists, ENa initiates the normal APPN flow sequence to bring up such a session. This begins with ENa sending a Locate Find Cdinit request to its network node server. The LU target of the session is the CP name of the DLUS node=NNe in this case.
2. The NNS(DLUR) issues either a broadcast or directed search for the CP_LU of the DLUS=NNe.
3. NNe responds to the search request with a Locate Found reply.
4. The NNS(DLUR) uses the information in the Locate reply to calculate a route to NNe and passes this back to ENa in the form of an RSCV.
5. ENa issues a CPSVRMGR BIND to NNe.
6. NNe provides a BIND response.
7. Following the RSP(BIND), ENa sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at NNe.
8. Once the Attach has been sent, ENa must send a REQACTPU to NNe. This will indicate to NNe that an ACTPU is requested on behalf of some local or downstream PU. Before the REQACTPU is sent, however, ENa must generate an FQPCID for this PU. This CV X′60′. will be used to uniquely identify the SSCP-PU sessions when more than one exists on a given CP-SVR pipe. A FID2 Encapsulation GDS variable (X′1500′) is built, including within it the FID2 PIU, XID image, PU TG CV, DL CAP CV, and FQPCID. The GDS variable is then sent on the conwinner CPSVRMGR session to NNe.
9. When the DLUR component of NNe receives the FID2 Encapsulation GDS variable, it removes the encapsulation headers and examines the information within the variable. At this point, NNe finds a DLUR/S capabilities mismatch (DLUR-DLUR pipe) and deactivates the CP-SVR pipe. This is begun by sending an UNBIND on the conloser session to its partner ENa, including sense data X′088E 0009′ in CV X′35′.
10. ENa sends a RSP(UNBIND) to the other DLUR for its conwinner session. The CP-SVR pipe is now deactivated. For more details on CP-SVR pipe deactivation see 5.6, “CP-SVR Pipe Deactivation” on page 5-20.

5.6.3 Abnormal SSCP-PU/SSCP-LU Session Deactivation (DLUR-Initiated)

(For DLUS-initiated SSCP-PU and SSCP-LU session deactivation, see 6.4, “SSCP-PU/SSCP-LU Session Deactivation (DLUS-Initiated)” on page 6-33)

A DLUR with a downstream PU (DSPU) has a mechanism (REQACTPU) to communicate to its DLUS a request to start an SSCP-PU session to that downstream PU. There must be a similar mechanism available to the DLUR to signal to the SSCP that the DLUR has lost its connection with the PU, so that the SSCP can terminate the SSCP-PU session (and SSCP-LU session(s) if any).

This mechanism shall be a new RU, REQDACTPU, to be sent on the SSCP-PU session by the DLUR to request to the SSCP that it take down the SSCP-PU session as well as any associated SSCP-LU sessions. Since the DLUR will no longer have a connection to the downstream device, it will have to respond to the DACTPU sent by the SSCP in response to the REQDACTPU. Once the SSCP-PU session has been deactivated, the DLUS can decide, using normal deactivation rules, whether or not to deactivate the CP-SVR pipe. There are three different deactivation scenarios, each identified with different cause value in the REQDACTPU RU:

- Cause X′00′: downstream PU outage
- Cause X′01′: receipt of REQDISCONT(normal) from downstream PU
- Cause X′02′: receipt of REQDISCONT(immediate) from downstream PU

These scenarios assume a positive response to REQDACTPU - additional processing is included at 5.6.3.3, “-RSP(REQDACTPU) DLUR Processing” on page 5-56 to handle the situation where the DLUR receives a negative response to REQDACTPU.

5.6.3.1 Downstream PU Outage

In this scenario the DLUR BF detects a loss of connectivity with the downstream PU (there are several possible failures, e.g., polling timeouts and link failures to name two). This situation corresponds in the subarea BF to an INOPERATIVE condition.

Optionally, the DLUR can try to recover the downstream connection and, failing that, send the REQDACTPU or it can send the REQDACTPU right away.

Figure 5-13 illustrates a downstream PU outage and the resulting SSCP-PU session and CP-SVR pipe deactivation:

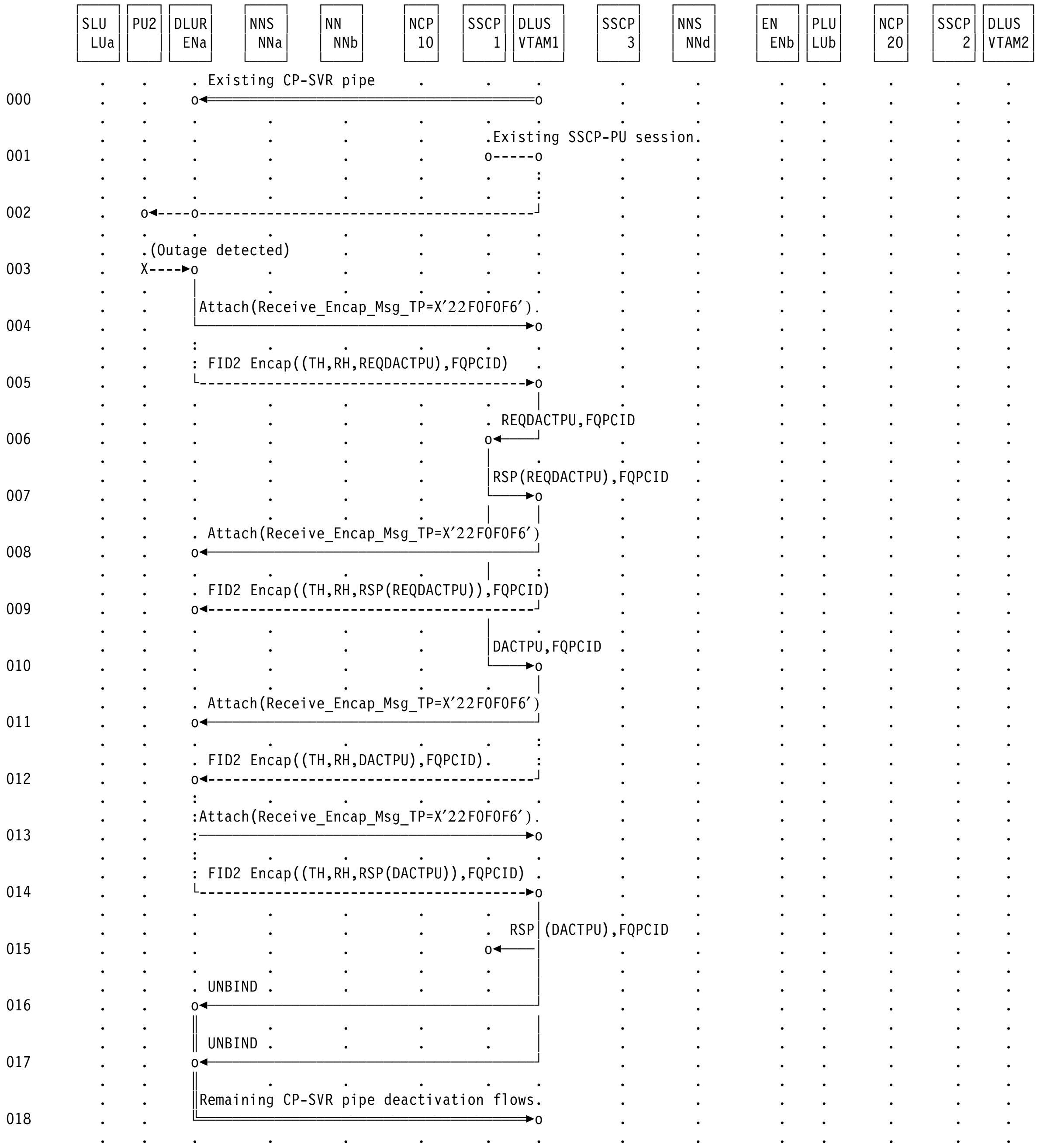


Figure 5-13. Downstream PU outage

1. There is an active session between SSCP1 and PU2. It is the only active SSCP-PU session using the CP-SVR pipe between VTAM1 and ENa.
2.
3. ENa detects an outage on its connection to PU2.
   - **Note** - If there had been an active LU-LU session involving LUa, ENa would build a Format 3 SESSEND RU, encapsulate it, and forward it to VTAM1 on the CP-SVR pipe. VTAM1 would then de-encapsulate the SESSEND and forward it to SSCP1.
4. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
5. To signal to SSCP1 that its SSCP-PU session with PU2 is broken, ENa builds a REQDACTPU with a cause value of X′00′, encapsulates it, and forwards it to VTAM1 on its CP-SVR pipe.
6. VTAM1 de-encapsulates the REQDACTPU and forwards it to SSCP1.
7. SSCP1 responds to the REQDACTPU positively, passing the RSP(REQDACTPU) on to VTAM1.
8. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
9. VTAM1 encapsulates the RSP(REQDACTPU) and forwards it to ENa.
10. SSCP1 builds a DACTPU and passes it on to VTAM1.
11. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
12. VTAM1 encapsulates the DACTPU and forwards it to ENa.
13. ENa de-encapsulates the DACTPU, and since it no longer has a connection to PU2, converts the DACTPU into a positive RSP(DACTPU). The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
14. ENa then encapsulates the RSP(DACTPU) and sends it to VTAM1.
15. VTAM1 de-encapsulates the RSP(DACTPU) and forwards it to SSCP1.
16. VTAM1 also determines that there are now no active or pending SSCP-PU sessions on its CP-SVR pipe to ENa, so it sends an UNBIND to ENa on VTAM1′s conwinner session to begin deactivation of the pipe.
17. VTAM1 also sends an UNBIND on its conloser session to ENa.
18. The DLUR node sends a RSP(UNBIND) to the DLUS for the conwinner session and another RSP(UNBIND) for the conloser session. For more details on CP-SVR pipe deactivation see 5.6, “CP-SVR Pipe Deactivation” on page 5-20.

5.6.3.2 Receipt Of REQDISCONT From Downstream PU

In these scenarios the DLUR BF receives a REQDISCONT RU from the downstream PU. In this situation the DLUR will:

- convert the REQDISCONT to a REQDACTPU, indicating a cause value of X′01′ if the REQDISCONT type is normal and X′02′ if the type is immediate
- encapsulate the REQDACTPU and send it to the DLUS on the CP-SVR pipe
- allow the DLUS to deactivate the PU gracefully if type = normal or forced if type = immediate (see 6.4, “SSCP-PU/SSCP-LU Session Deactivation (DLUS-Initiated)” on page 6-33 for more information on normal and forced SSCP-PU and SSCP-LU session deactivation), and then
- deactivate the DLC from the BF to the DSPU (emulating the receipt of a DISCONTACT)

Figure 5-14 illustrates receipt of a REQDISCONT(normal) and the resulting SSCP-PU/LU session and CP-SVR pipe deactivation:

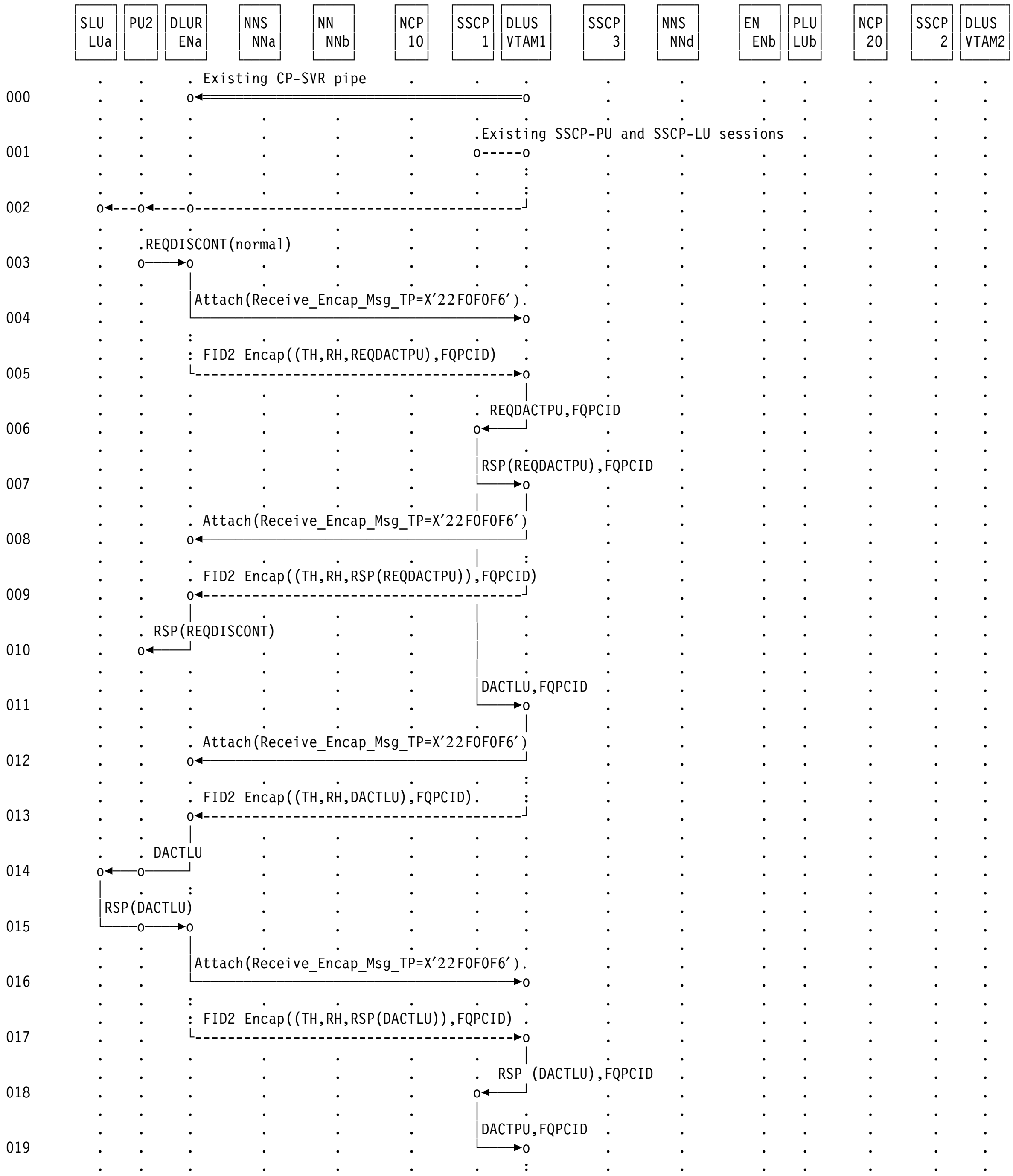


Figure 5-14. Receipt of REQDISCONT(normal) from downstream PU

```
           SLU  PU2  DLUR     NNS      NN       NCP     SSCP  DLUS      SSCP     NNS       EN   PLU    NCP     SSCP  DLUS
           LUa       ENa      NNa      NNb      10      1     VTAM1     3        NNd       ENb  LUb    20      2     VTAM2
           .    .     . Attach(Receive_Encap_Msg_TP=X'22F0F0F6':         .        .         .    .       .       .     .
020        .    .     o◄───────────────────────────────────────:         .        .         .    .       .       .     .
           .    .     .        .        .        .       .     :         .        .         .    .       .       .     .
           .    .     . FID2 Encap((TH,RH,DACTPU),FQPCID).     :         .        .         .    .       .       .     .
021        .    .     o◄---------------------------------------┘         .        .         .    .       .       .     .
           .    .     │        .        .        .       .     .         .        .         .    .       .       .     .
           .    . DACTPU       .        .        .       .     .         .        .         .    .       .       .     .
022        .    o◄────┘        .        .        .       .     .         .        .         .    .       .       .     .
           .    │     .        .        .        .       .     .         .        .         .    .       .       .     .
           .    │RSP(DACTPU)   .        .        .       .     .         .        .         .    .       .       .     .
023        .    └────►o        .        .        .       .     .         .        .         .    .       .       .     .
           .    .     ║        .        .        .       .     .         .        .         .    .       .       .     .
           .    .     ║Attach(Receive_Encap_Msg_TP=X'22F0F0F6').         .        .         .    .       .       .     .
024        .    .     ║───────────────────────────────────────►o         .        .         .    .       .       .     .
           .    .     ║        .        .        .       .     .         .        .         .    .       .       .     .
           .    .     ║FID2 Encap((TH,RH,RSP(DACTPU)),FQPCID)  .         .        .         .    .       .       .     .
025        .    .     ║---------------------------------------►o         .        .         .    .       .       .     .
           .    .     ║        .        .        .       .     │         .        .         .    .       .       .     .
           .    . DLC deactivation      .        .       .     │         .        .         .    .       .       .     .
026        .    o◄════╝        .        .        .       .     │         .        .         .    .       .       .     .
           .    .     .        .        .        .       .     │         .        .         .    .       .       .     .
           .    .     .        .        .        .       . RSP(│ACTPU),FQPCID     .         .    .       .       .     .
027        .    .     .        .        .        .       o◄────│         .        .         .    .       .       .     .
           .    .     .        .        .        .       .     │         .        .         .    .       .       .     .
           .    .     . UNBIND .        .        .       .     │         .        .         .    .       .       .     .
028        .    .     o◄───────────────────────────────────────│         .        .         .    .       .       .     .
           .    .     ║        .        .        .       .     │         .        .         .    .       .       .     .
           .    .     ║ UNBIND .        .        .       .     │         .        .         .    .       .       .     .
029        .    .     o◄───────────────────────────────────────┘         .        .         .    .       .       .     .
           .    .     ║        .        .        .       .     .         .        .         .    .       .       .     .
           .    .     ║Remaining CP-SVR pipe deactivation flows.         .        .         .    .       .       .     .
030        .    .     ╚═══════════════════════════════════════►o         .        .         .    .       .       .     .
           .    .     .        .        .        .       .     .         .        .         .    .       .       .     .
```

1. There are active sessions between SSCP1 and PU2 and SSCP1 and LUa. These are the only active SSCP-PU and SSCP-LU sessions using the CP-SVR pipe between VTAM1 and ENa.
2.
3. ENa receives a REQDISCONT from PU2.
4. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
5. To signal to SSCP1 that its SSCP-PU session with PU2 should be deactivated, ENa builds a REQDACTPU with a cause value of X′01′, encapsulates it, and forwards it to VTAM1 on its CP-SVR pipe.
6. VTAM1 de-encapsulates the REQDACTPU and forwards it to SSCP1.
7. SSCP1 responds to the REQDACTPU positively, passing the RSP(REQDACTPU) on to VTAM1.
8. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
9. VTAM1 encapsulates the RSP(REQDACTPU) and forwards it to ENa.
10. ENa then converts the REQDISCONT into a positive RSP(REQDISCONT) and forwards it to PU2.
11. SSCP1 builds a DACTLU and passes it on to VTAM1.
12. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
13. VTAM1 encapsulates the DACTLU and forwards it to ENa.
14. ENa de-encapsulates the DACTLU and forwards it to PU2 and then to LUa.
15. LUa deactivates the SSCP-LU session, converts the DACTLU into a positive RSP(DACTLU), and forwards it to PU2 and then to ENa.
16. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
17. ENa then encapsulates the RSP(DACTLU) and sends it to VTAM1.
18. VTAM1 de-encapsulates the RSP(DACTLU) and forwards it to SSCP1.
19. SSCP1 builds a DACTPU and passes it on to VTAM1.
20. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
21. VTAM1 encapsulates the DACTPU and forwards it to ENa.
22. ENa de-encapsulates the DACTPU and forwards it to PU2.
23. PU2 deactivates the SSCP-PU session, converts the DACTPU into a positive RSP(DACTPU), and forwards it to ENa.
24. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
25. ENa then encapsulates the RSP(DACTPU) and sends it to VTAM1.
26. ENa initiates DLC deactivation to PU2.
27. VTAM1 de-encapsulates the RSP(DACTPU) and forwards it to SSCP1.
28. VTAM1 also determines that there are now no active or pending SSCP-PU sessions on its CP-SVR pipe to ENa, so it sends an UNBIND to ENa on VTAM1′s conwinner session to begin deactivation of the pipe.
29. VTAM1 also sends an UNBIND on its conloser session to ENa.
30. The DLUR node sends a RSP(UNBIND) to the DLUS for the conwinner session and another RSP(UNBIND) for the conloser session. For more details on CP-SVR pipe deactivation see 5.6, “CP-SVR Pipe Deactivation” on page 5-20.

Figure 5-15 illustrates receipt of a REQDISCONT(immediate) and the resulting SSCP-PU/LU session and CP-SVR pipe deactivation:

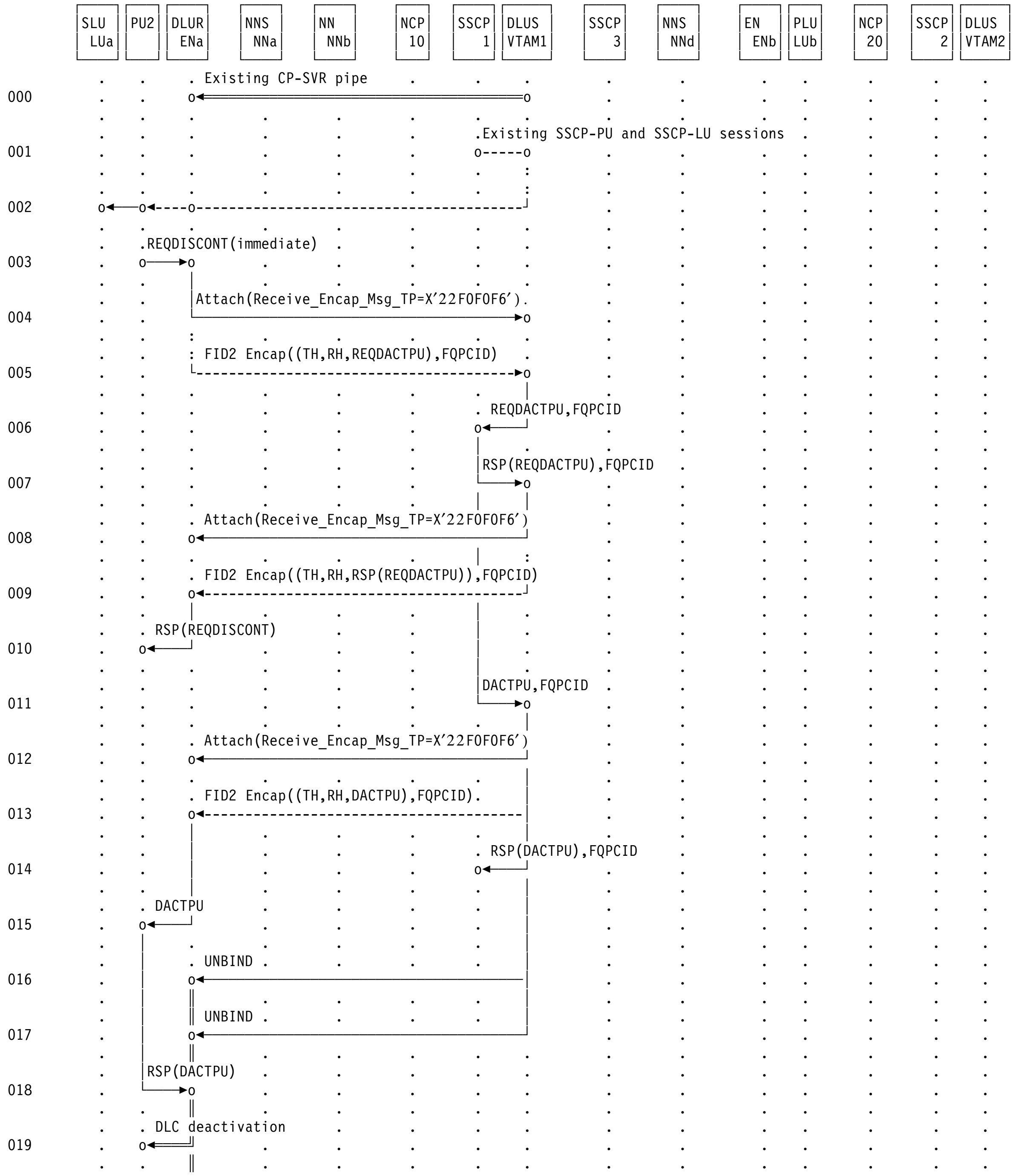


Figure 5-15. Receipt of REQDISCONT(immediate) from downstream PU

```
          ┌────┐ ┌───┐ ┌────┐    ┌────┐    ┌────┐     ┌───┐   ┌────┐┌─────┐   ┌────┐    ┌────┐     ┌────┐┌───┐    ┌───┐   ┌────┐┌─────┐
          │SLU │ │PU2│ │DLUR│    │NNS │    │NN  │     │NCP│   │SSCP││DLUS │   │SSCP│    │NNS │     │EN  ││PLU│    │NCP│   │SSCP││DLUS │
          │ LUa│ │   │ │ ENa│    │ NNa│    │ NNb│     │ 10│   │   1││VTAM1│   │   3│    │ NNd│     │ ENb││LUb│    │ 20│   │   2││VTAM2│
          └────┘ └───┘ └────┘    └────┘    └────┘     └───┘   └────┘└─────┘   └────┘    └────┘     └────┘└───┘    └───┘   └────┘└─────┘
             .     .   ║Remaining CP-SVR pipe deactivation flows.   .         .           .    .        .       .     .
020          .     .   ╚═══════════════════════════════════════►o   .         .           .    .        .       .     .
             .     .     .       .        .         .       .   .   .         .           .    .        .       .     .
```

1. There are active sessions between SSCP1 and PU2 and SSCP1 and LUa. These are the only active SSCP-PU and SSCP-LU sessions using the CP-SVR pipe between VTAM1 and ENa.
2.
3. ENa receives a REQDISCONT from PU2.
4. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
5. To signal to SSCP1 that its SSCP-PU session with PU2 should be deactivated, ENa builds a REQDACTPU with a cause value of X′02′, encapsulates it, and forwards it to VTAM1 on its CP-SVR pipe.
6. VTAM1 de-encapsulates the REQDACTPU and forwards it to SSCP1.
7. SSCP1 responds to the REQDACTPU positively, passing the RSP(REQDACTPU) on to VTAM1.
8. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
9. VTAM1 encapsulates the RSP(REQDACTPU) and forwards it to ENa.
10. ENa then converts the REQDISCONT into a positive RSP(REQDISCONT) and forwards it to PU2.
11. SSCP1 builds a DACTPU and passes it on to VTAM1.
12. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
13. VTAM1 encapsulates the DACTPU and forwards it to ENa.
14. VTAM1 builds a RSP(DACTPU) and forwards it to SSCP1, not waiting for the response to return from the PU.
15. ENa de-encapsulates the DACTPU and forwards it to PU2.
16. VTAM1 determines that there are now no active or pending SSCP-PU sessions on its CP-SVR pipe to ENa, so it sends an UNBIND to ENa on VTAM1′s conwinner session to begin deactivation of the pipe.
17. VTAM1 also sends an UNBIND on its conloser session to ENa.
18. PU2 deactivates the SSCP-PU and associated SSCP-LU session. It converts the DACTPU into a positive RSP(DACTPU) and returns it to ENa.
19. ENa initiates DLC deactivation to PU2, and discards the RSP(DACTPU) since the CP-SVR pipe is in the process of being deactivated.
20. The DLUR node sends a RSP(UNBIND) to the DLUS for the conwinner session and another RSP(UNBIND) for the conloser session. For more details on CP-SVR pipe deactivation see 5.6, “CP-SVR Pipe Deactivation” on page 5-20.

5.6.3.3 -RSP(REQDACTPU) DLUR Processing

When the DLUR receives a negative response to REQDACTPU, regardless of the cause type, it will reset the PU and its associated LUs. This may include:

- resetting the LU-LU sessions, creating and sending SESSENDs
- resetting the SSCP-LU sessions
- resetting the SSCP-PU session
- deactivating the DLC if there was no outage

All of the -RSP(REQDACTPU) cases will be handled as the +RSP(REQDACTPU) downstream PU outage case (see 5.6.3.1, “Downstream PU Outage” on page 5-48), i.e., no DACTLUs or DACTPUs will be built and sent to the downstream devices.

5.7 Pipe And Session Failure Logic

As mentioned above, CP-SVR pipes will actually consist of two CPSVRMGR sessions between the DLUR and DLUS CPs. The SSCP-PU and SSCP-LU sessions will be encapsulated on the two CPSVRMGR sessions such that flows from the SSCP to the PU/LU will flow on the conwinner session from DLUS to DLUR node, and flows from the PU/LU to the SSCP will flow on the conwinner session from DLUR to DLUS node. Moreover, each node (DLUR and DLUS) will compute the session route independently. It is, therefore, possible that the two CPSVRMGR sessions will not traverse the same physical path. Consequently, one direction of the SSCP-PU and SSCP-LU sessions may fail and not the other. Even if the same physical path is used, it is still possible for one session to fail and not the other. This will necessitate special handling on the part of DLUS and DLUR nodes.

5.7.1 CPSVRMGR Session Recovery

If a CPSVRMGR session fails, it is the responsibility of both the DLUS and the DLUR to bring down its conwinner and conloser sessions with nonretryable sense data. At this point, either or both nodes may attempt to reinstate the CP-SVR pipe and recover the SSCP-PU/LU sessions.

The DLUR node may chose to attempt to reactivate its dependent LUs by activating a CP-SVR pipe (and subsequent SSCP-PU and SSCP-LU sessions) with the same or some other DLUS node. The DLUR node may only attempt this after both the conwinner and conloser sessions have been successfully deactivated. Otherwise, the DLUR node may activate a conwinner session with one DLUS node while a conloser session exists with some other DLUS node. Alternative DLUS nodes may be either dynamically discovered (as described in 5.3.2.2, “Dynamic DLUS Node Determination” on page 5-5) or predefined (as described in 5.3.2.1, “Predefined DLUS Node Determination” on page 5-4).

As previously described in 5.6.2.1, “UNBIND-Initiated CP-SVR Pipe Deactivation” on page 5-30, the DLUR may reactivate immediately after completion of the CP-SVR pipe deactivation, with one exception (receipt of X′08A0 000A′ sense data with UNBIND requires the DLUR to wait for a DLUS to activate the CP-SVR pipe; DLUR can wait indefinitely or set a timeout, after which the DLUR can activate a CP-SVR pipe with the same or a different DLUS).

It is also possible that the DLUS node may try to re-activate the dependent LUs by re-establishing a CP-SVR pipe to the DLUR node as described in 5.4, “DLUS-Initiated CP-SVR Pipe Activation” on page 5-6.

5.7.2 SSCP-PU And SSCP-LU Session Recovery

When both of the CPSVRMGR sessions are lost, the DLUS and DLUR nodes must recognize this situation. Ongoing LU-LU sessions may remain active but will not be able to obtain SSCP services when required. This is functionally equivalent to existing subarea operation. The SSCP-PU and SSCP-LU sessions can be recovered through SSCP takeover (i.e., DLUS takeover) flows which reset the SSCP-PU and SSCP-LU sessions as described in 10.2.2, “SSCP Takeover and Giveback Flows” on page 10-11. When the DLUR attempts to reactivate the SSCP-PU and SSCP-LU sessions, it can choose the same or another DLUS node. If the reason for session deactivation was the DLUR receiving a DACTPU with a “giveback” reason code, the DLUR should try to get in session with a new DLUS.

5.7.3 Pipe And Session Activation/Recovery Race Conditions

5.7.3.1 Multi-DLUS Race Conditions

Race conditions can exist where more than one DLUS may be involved in CP-SVR pipe activation for the same dependent LU. For example:

- ***initial CP-SVR pipe activation*** - a pipe is being activated and an activation request for a second pipe is received
- ***CP-SVR pipe override*** - a pipe to a backup DLUS is active and the predefined DLUS requests activation of a pipe
- ***CP-SVR pipe recovery*** - a pipe to a backup DLUS is being activated and the predefined DLUS requests activation of a pipe

In these situations, the DLUR node is responsible for deciding with which DLUS it will establish the CP-SVR pipe for a given PU. In doing so, the DLUR will use the following rules:

- If the DLUR has a choice between an active pipe and a request to activate a pipe, it should choose the DLUS with the active pipe.
- If the DLUR has a choice between two requests for activation, the implementing product can decide on its DLUS selection criteria:
  - "known" (predefined) vs. "determined" (found) - select "known"
  - "determined" vs. "determined" - product option to choose one

Once the race condition is resolved, the DLUR node is responsible for rejecting the CP-SVR pipe with one of the two DLUS nodes. This is done by sending -RSP to one of the ACTPU messages. The DLUS receiving the -RSP(ACTPU) will then decide whether to take down the CP-SVR pipe using the same criteria as normal CP-SVR pipe deactivation, i.e., if there are no other active or pending SSCP-PU sessions using the CP-SVR pipe, the DLUS will UNBIND its CPSVRMGR conwinner and conloser sessions.

Figure 5-16 on page 5-60 illustrates an unsuccessful attempt at overriding an existing CP-SVR pipe:

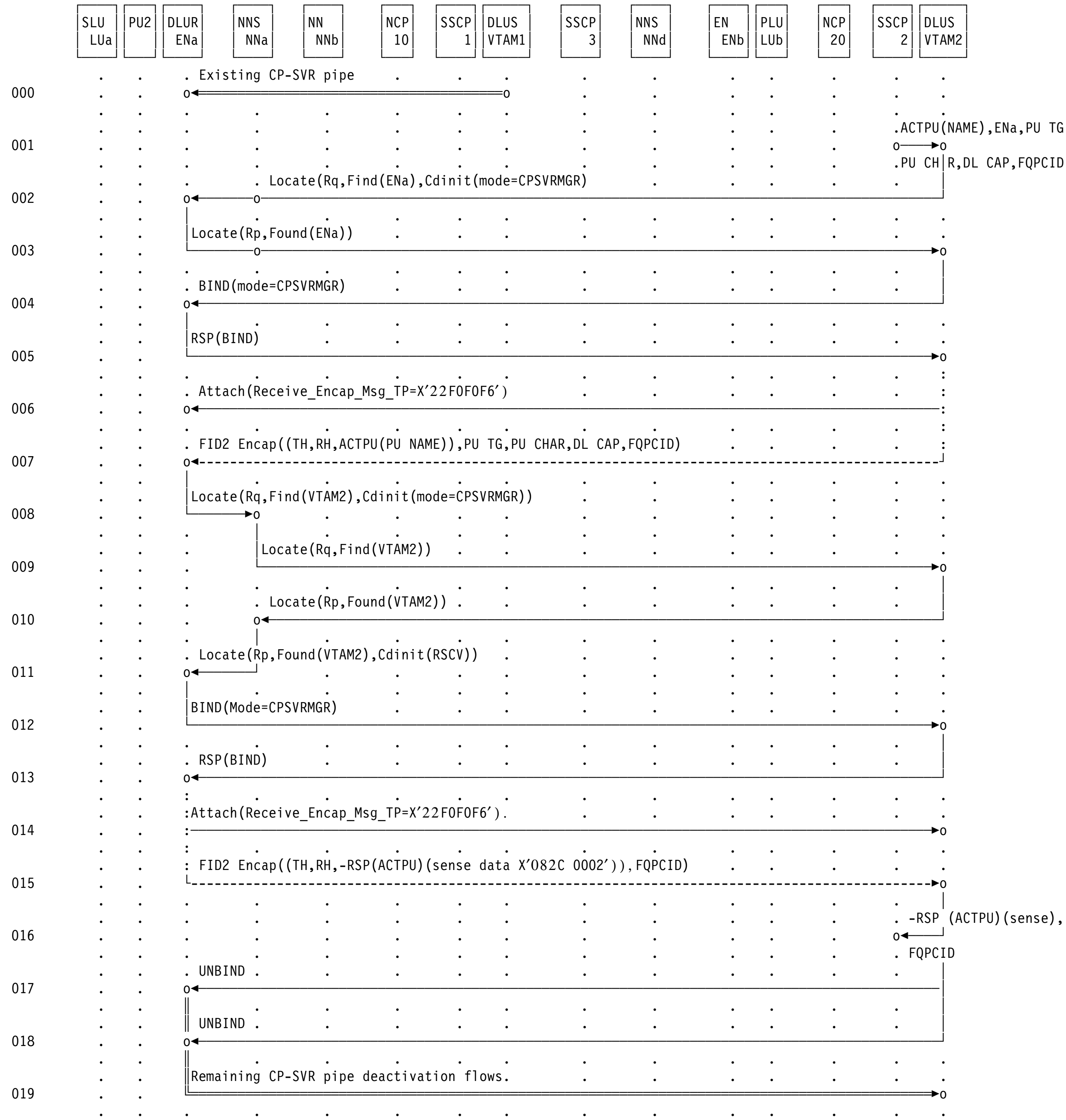


Figure 5-16. CP-SVR pipe override (rejected)

1. There is an active CP-SVR pipe between VTAM1 and ENa, where VTAM1 is a “known” DLUS. SSCP1 and PU2 have an active SSCP-PU session. SSCP2 builds an ACTPU to start an SSCP-PU session with PU2 and forwards the RU to its DLUS, VTAM2.
2. VTAM2 locates the DLUR ENa.
3.
4. VTAM2 activates its CPSVRMGR conwinner session with ENa.
5.
6.
7. VTAM2 encapsulates the ACTPU and sends it to ENa. This ACTPU is from SSCP2 via VTAM2, which to ENa is a “determined” DLUS. Since there is already an active SSCP-PU session for PU2 (doesn′t matter whether it is with a “known” or “determined” DLUS), the new ACTPU will be rejected with sense data X′082C 0002′.
8. ENa locates the DLUS VTAM2.
9.
10.
11.
12. ENa activates its CPSVRMGR conwinner session with VTAM2.
13.
14.
15. ENa sends a negative RSP(ACTPU) to terminate the pending SSCP-PU session.
16.
17. VTAM2 also determines that there are now no active or pending SSCP-PU sessions on its CP-SVR pipe to ENa, so it sends an UNBIND to ENa on VTAM2′s conwinner session to begin deactivation of the pipe.
18. VTAM2 also sends an UNBIND on its conloser session to ENa.
19. The DLUR node sends a RSP(UNBIND) to the DLUS for the conwinner session and another RSP(UNBIND) for the conloser session. For more details on CP-SVR pipe deactivation see 5.6, “CP-SVR Pipe Deactivation” on page 5-20.

5.7.3.2 Single DLUS Race Conditions

Race conditions can exist between a DLUR and one DLUS. For example:

- ***CP-SVR pipe deactivation/SSCP-PU session activation race*** - a DLUR has just sent a REQACTPU to the DLUS, and it receives an UNBIND for the CP-SVR pipe with that DLUS
  - the DLUR has two options, equivalent to the Auto Network Shutdown function (user preference could optionally be predefined per PU):
    - the DLUR could deactivate the CP-SVR pipe and subsequently redrive reactivation of that pipe for the pending SSCP-PU session, retransmitting the REQACTPU on the new pipe (similar to ANS=CONT)
    - the DLUR could deactivate the CP-SVR pipe and the connection to the PU (similar to ANS=STOP)
- ***"surprise" CP-SVR pipe activation*** - a pipe is active and an activation request for the same pipe is received from the same DLUS
  - the activation request should come from a DLUS attempting to recover a CP-SVR pipe it has determined has gone down; since the DLUR has not yet determined that the pipe is down, the new activation request should be rejected (no available conwinner or conloser sessions to that DLUS, specifying sense code X′0805 0000′ in the -RSP(BIND), or in an UNBIND if BIND has already been processed due to implementation variations) - eventually the DLUR will determine that the pipe needs to be deactivated (for instance, receipt of UNBIND from DLUS) and will complete the pipe deactivation and then drive pipe reactivation
- ***SSCP-PU session deactivation race (DACTPU/REQDACTPU)*** - a node (either requester or server) is processing or has sent an SSCP-PU session deactivation request (DACTPU from the DLUS; REQDACTPU from the DLUR), and it receives an SSCP-PU session deactivation request for the same PU from the other node
  - DLUR processing
    - if the requester is in an awaiting RSP(REQDACTPU) state and it receives a DACTPU from the DLUS for that PU, it should accept and process that DACTPU request
    - if while processing the DACTPU, the +RSP(REQDACTPU) should arrive, the requester should accept the response and continue DACTPU processing

Figure 5-17 on page 5-63 through Figure 5-20 on page 5-72 illustrate examples of these race conditions:

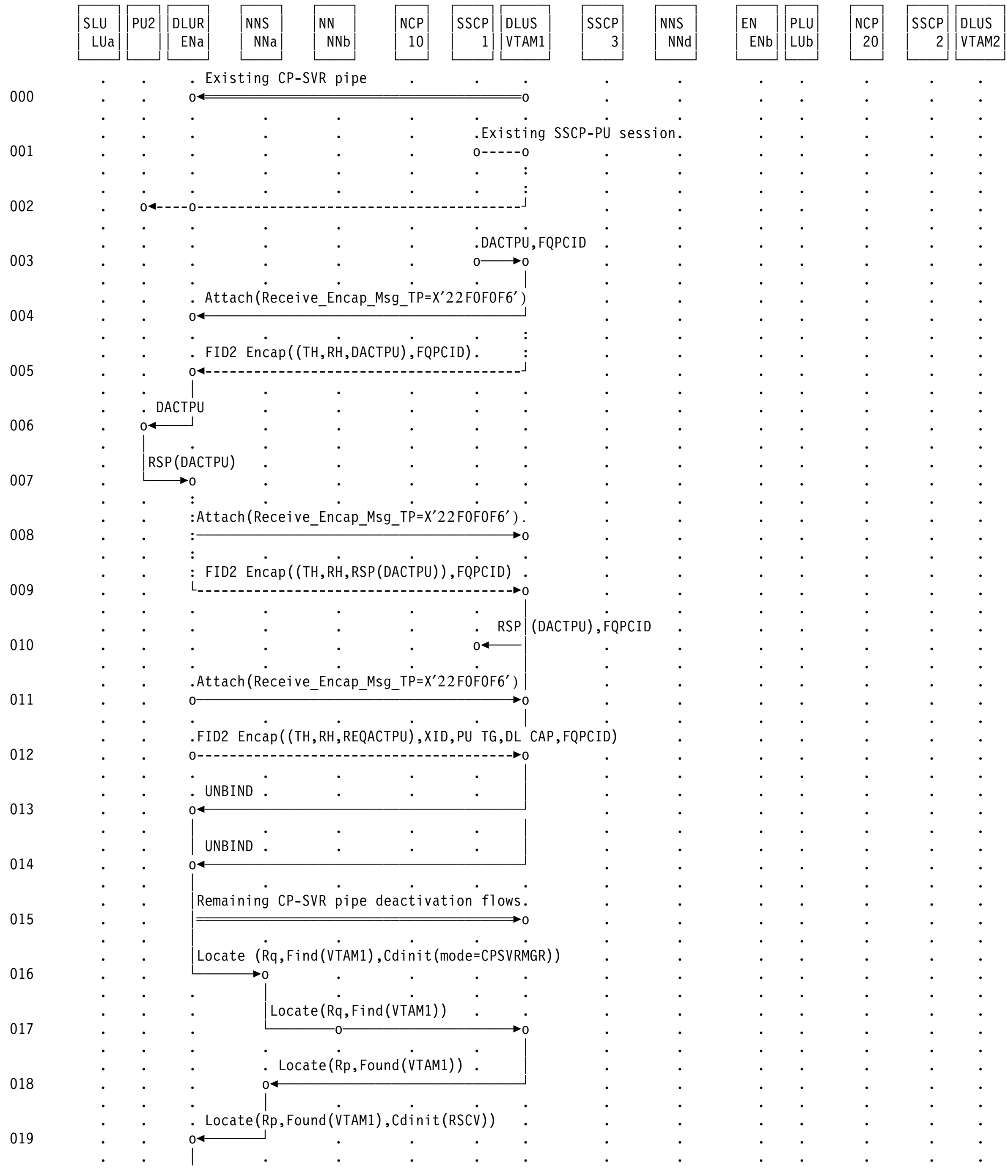
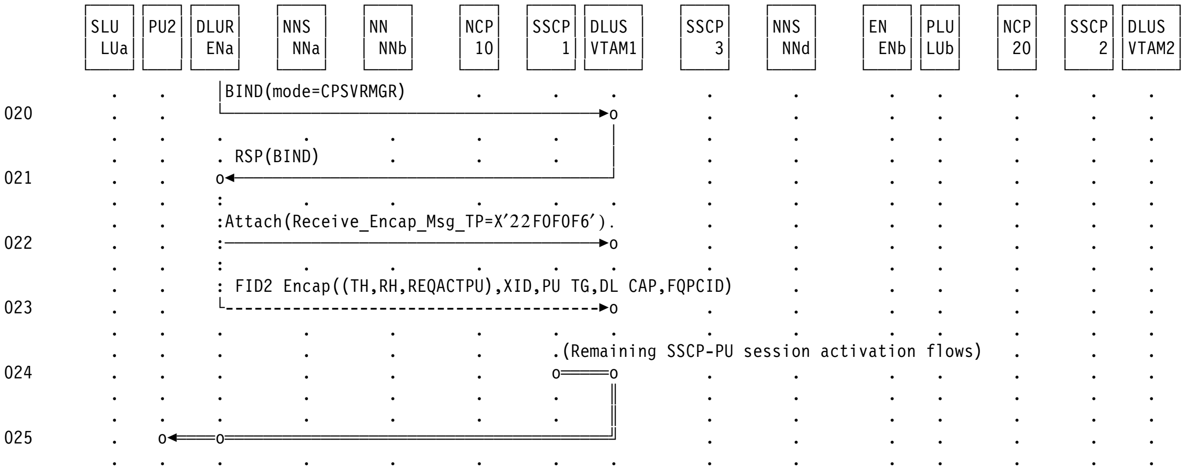


Figure 5-17. CP-SVR pipe deactivation/SSCP-PU session activation race - redrive pipe activation



SLU LUa
PU2
DLUR ENa
NNS NNa
NN NNb
NCP 10
SSCP 1
DLUS VTAM1
SSCP 3
NNS NNd
EN ENb
PLU LUb
NCP 20
SSCP 2
DLUS VTAM2
020
BIND(mode=CPSVRMGR)
021
RSP(BIND)
022
Attach(Receive_Encap_Msg_TP=X'22F0F0F6')
023
FID2 Encap((TH,RH,REQACTPU),XID,PU TG,DL CAP,FQPCID)
024
(Remaining SSCP-PU session activation flows)
025

1. There is an active CP-SVR pipe between VTAM1 and ENa, where VTAM1 is a "known" DLUS. SSCP1 and PU2 have an active SSCP-PU session. It is the only active SSCP-PU session using the CP-SVR pipe between VTAM1 and ENa. It has been indicated that PU2 session with SSCP1 should be restarted in case of pipe failure.
2.
3. SSCP1 initiates normal deactivation of its session with PU2 by building a DACTPU and passing it on to VTAM1.
4. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
5. The DLUS VTAM1 encapsulates the DACTPU and forwards it to the DLUR ENa.
6. The DLUR ENa de-encapsulates the DACTPU and forwards it to PU2.
7. PU2 returns a positive RSP(DACTPU) to SSCP1 via ENa.
8. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
9. ENa encapsulates the RSP(DACTPU) and sends it to VTAM1.
10. VTAM1 de-encapsulates the RSP(DACTPU) and forwards it to SSCP1.
11. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
12. ENa encapsulates and sends to VTAM1 a REQACTPU.
13. Meanwhile VTAM1 determines that there are now no active or pending SSCP-PU sessions on its CP-SVR pipe to ENa, so it sends an UNBIND to ENa on VTAM1′s conwinner session to begin deactivation of the pipe.
14. VTAM1 also sends an UNBIND on its conloser session to ENa.
15. The DLUR node sends a RSP(UNBIND) to the DLUS for the conwinner session and another RSP(UNBIND) for the conloser session. For more details on CP-SVR pipe deactivation see 5.6, "CP-SVR Pipe Deactivation" on page 5-20.
16. ENa determines that there was a pending SSCP-PU session on the pipe, and it restarts activation of the CP-SVR pipe to VTAM1.
17.
18.
19.
20.
21.
22. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
23. ENa encapsulates and sends to VTAM1 the REQACTPU sent originally on the previous pipe.
24. SSCP-PU session reactivation proceeds normally.
25.

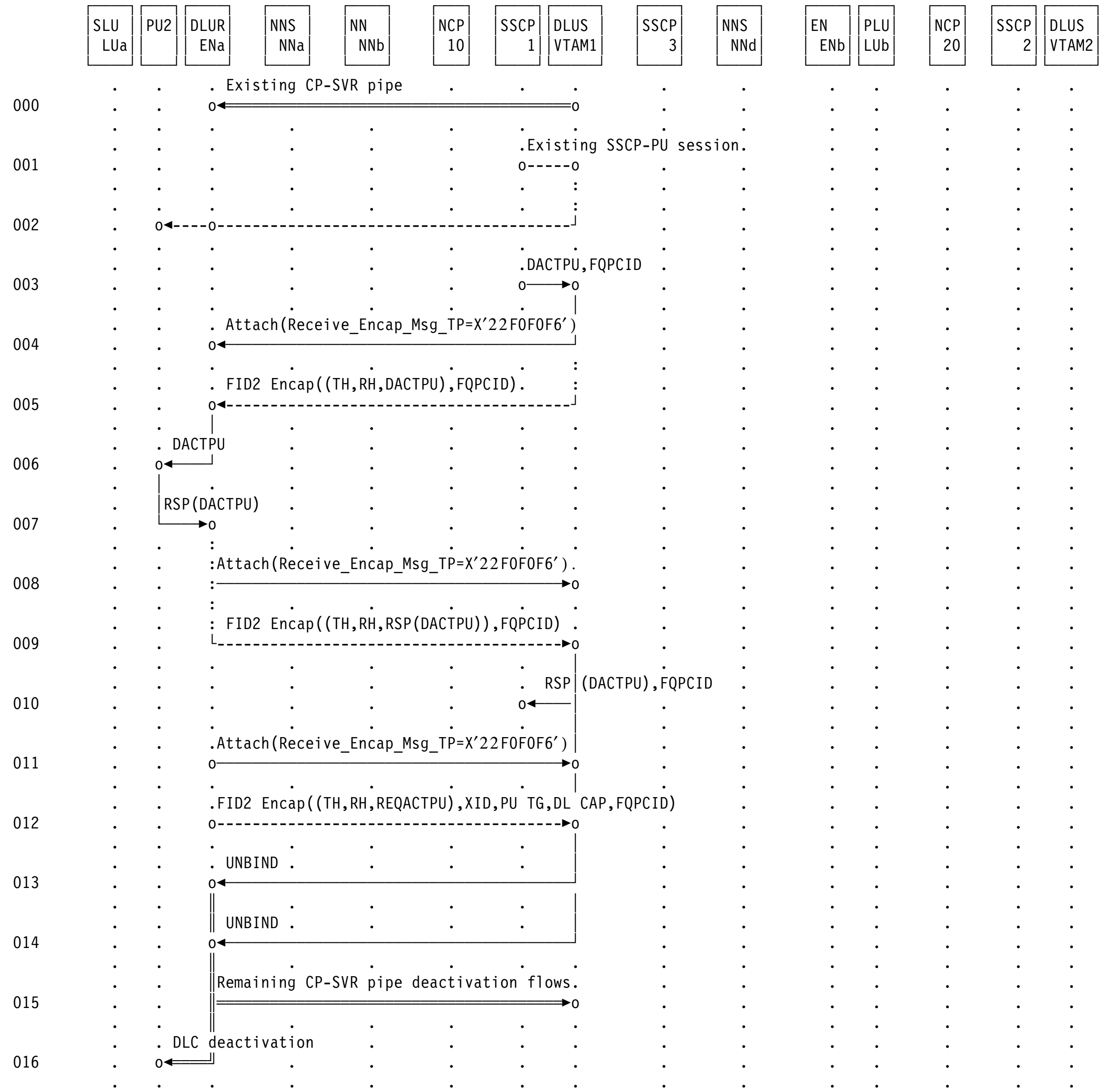


Figure 5-18. CP-SVR pipe deactivation/SSCP-PU session activation race - do not redrive pipe activation

1. There is an active CP-SVR pipe between VTAM1 and ENa, where VTAM1 is a "known" DLUS. SSCP1 and PU2 have an active SSCP-PU session. It is the only active SSCP-PU session using the CP-SVR pipe between VTAM1 and ENa. It has been indicated that PU2 session with SSCP1 should not be restarted in case of pipe failure.
2.
3. SSCP1 initiates normal deactivation of its session with PU2 by building a DACTPU and passing it on to VTAM1.
4. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
5. The DLUS VTAM1 encapsulates the DACTPU and forwards it to the DLUR ENa.
6. The DLUR ENa de-encapsulates the DACTPU and forwards it to PU2.
7. PU2 returns a positive RSP(DACTPU) to SSCP1 via ENa.
8. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
9. ENa encapsulates the RSP(DACTPU) and sends it to VTAM1.
10. VTAM1 de-encapsulates the RSP(DACTPU) and forwards it to SSCP1.
11. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
12. ENa encapsulates and sends to VTAM1 a REQACTPU.
13. Meanwhile VTAM1 determines that there are now no active or pending SSCP-PU sessions on its CP-SVR pipe to ENa, so it sends an UNBIND to ENa on VTAM1′s conwinner session to begin deactivation of the pipe.
14. VTAM1 also sends an UNBIND on its conloser session to ENa.
15. The DLUR node sends a RSP(UNBIND) to the DLUS for the conwinner session and another RSP(UNBIND) for the conloser session. For more details on CP-SVR pipe deactivation see 5.6, "CP-SVR Pipe Deactivation" on page 5-20.
16. ENa determines that there was a pending SSCP-PU session on the pipe, but it does not attempt to redrive the activation request. Instead it terminates the downstream PU connection.

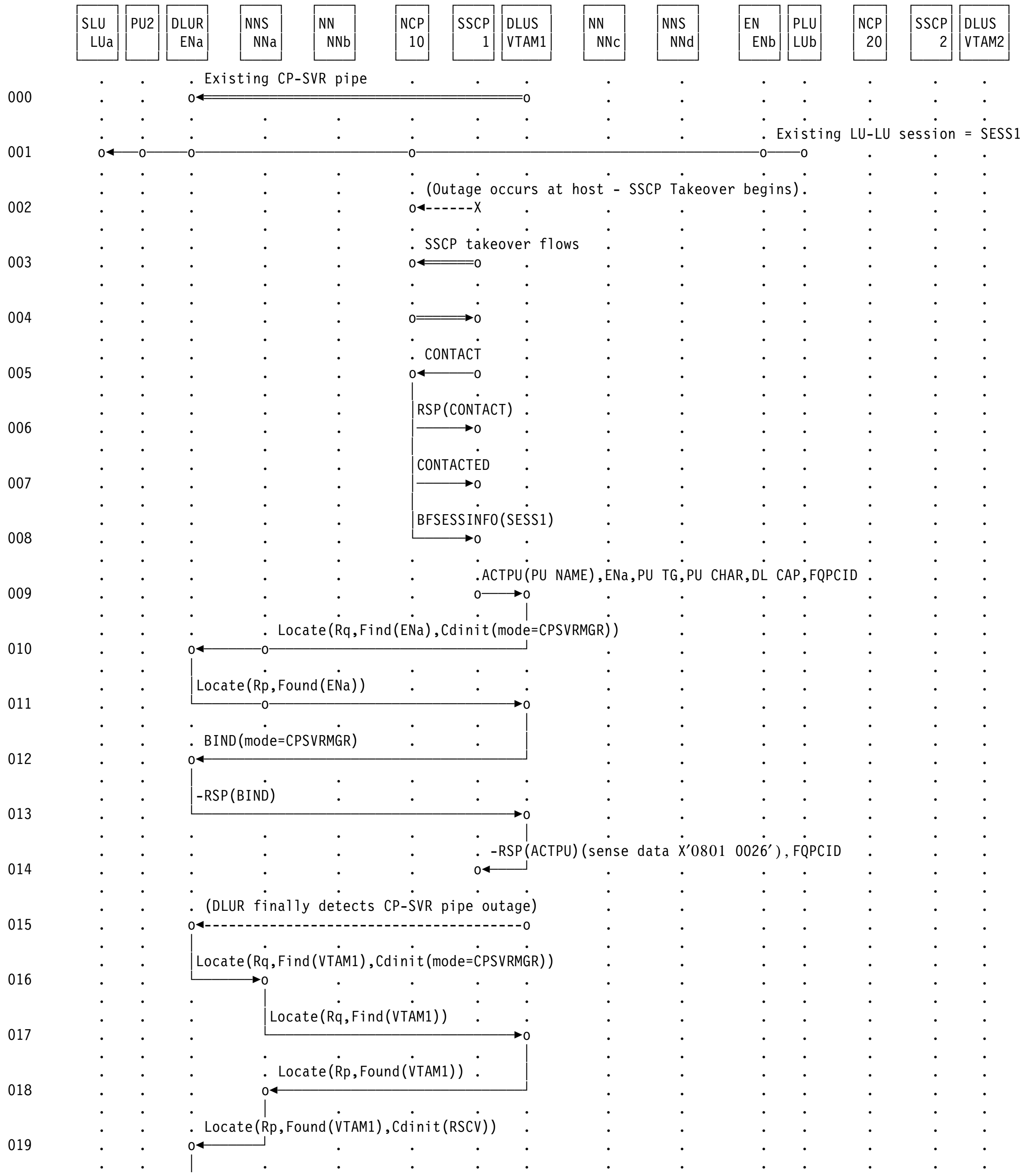
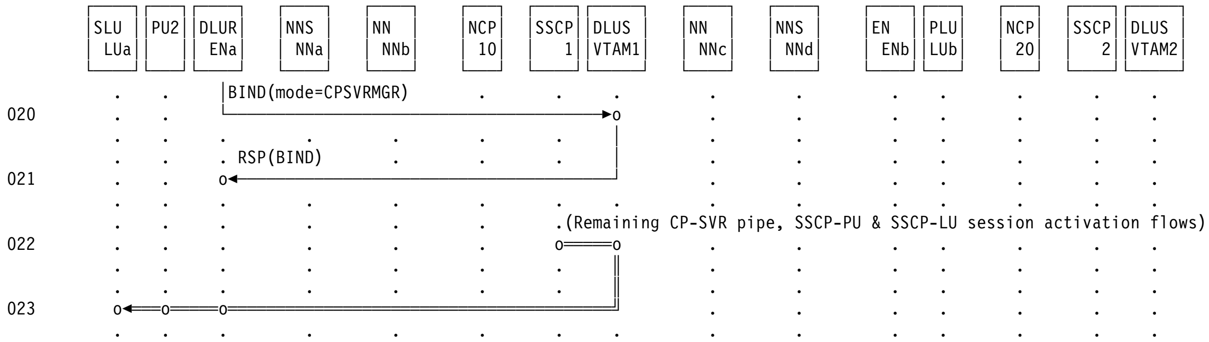


Figure 5-19. "Surprise" CP-SVR pipe activation


SLU LUa
PU2
DLUR ENa
NNS NNa
NN NNb
NCP 10
SSCP 1
DLUS VTAM1
NN NNc
NNS NNd
EN ENb
PLU LUb
NCP 20
SSCP 2
DLUS VTAM2
BIND(mode=CPSVRMGR)
020
RSP(BIND)
021
(Remaining CP-SVR pipe, SSCP-PU & SSCP-LU session activation flows)
022
023

1. The current LU-LU session between LUa and LUb routes through PU2, ENa, NNa, NNb, NCP10, NNc, NNd, and ENb.
2. The SSCP-NCP session between SSCP1 and NCP10 ends due to a host outage.
3. SSCP1 reacquires ownership of NCP10 and its attached resources.
4.
5.
6.
7.
8. When SSCP1 has acquired ownership of the link station through which the LU-LU session is routed, NCP10 builds and sends to SSCP1 a BFSESSINFO. This PIU indicates that a session exists between LUa (which it believes to be an independent LU) and LUb.
9. SSCP1 issues ACTPU to PU2 to restart its SSCP-PU session. SSCP1 passes the ACTPU to its DLUS component, VTAM1.
10. VTAM1 determines that ENa is the DLUR associated with PU2. VTAM1 then allocates a conwinner conversation between its Send_Encap_Msg_TP and the Receive_Encap_Msg_TP at the DLUR node. Since no session already exists to carry this conversation, VTAM1 starts one by sending out a Locate Find Cdinit for the CP_LU of the DLUR, mode=CPSVRMGR.
11. The DLUR ENa returns a Locate reply.
12. The DLUS VTAM1 uses the information in the Locate reply to calculate a route to the DLUR node and then builds and sends a BIND using the CPSVRMGR mode.
13. Since ENa has not yet received indication of the loss of the current CP-SVR pipe with VTAM1, it has no available conloser session to provide for the new CP-SVR pipe activation request. Therefore, ENa sends back a negative response to the BIND indicating there is no available conloser session, specifying sense data
    X′0805 0000′ in the -RSP(BIND)
    (or in an UNBIND if BIND has already been processed due to implementation variations).
14. Upon receipt of the negative response to the BIND, VTAM1 converts the ACTPU request into a negative response (sense data X′0801 0026′) and returns it to SSCP1.
15. ENa finally receives indication of the failure of the CP-SVR pipe with VTAM1. Having an attached LU with an active LU-LU session, ENa will attempt to activate a CP-SVR pipe with a DLUS.
16. The DLUR node determines the the DLUS node to be contacted on behalf of PU2. Once the DLUS node has been determined, the DLUR node Allocates a conwinner conversation between its Send_Encap_Msg_TP and the Receive_Encap_Msg_TP (X′22F0F0F6′) at the DLUS node. Since no session using the CPSVRMGR mode between this DLUR and the DLUS already exists, the DLUR node initiates the normal APPN flow sequence to bring up such a session. This begins with the DLUR node sending a Locate Find Cdinit request to its network node server. The LU target of the session is the CP name of the DLUS node=VTAM1 in this case.
17. The NNS(ENa) issues either a broadcast or directed search for the CP_LU of the DLUS=VTAM1.
18. The DLUS node responds to the search request with a Locate Found reply.
19. The NNS(ENa) uses the information in the Locate reply to calculate a route to the DLUS node and passes this back to the DLUR in the form of an RSCV.
20. The DLUR node ENa issues a CPSVRMGR BIND request to the DLUS node VTAM1.
21. The DLUS node VTAM1 returns a positive response to the BIND.

22. The activation of the CP-SVR pipe and the SSCP-PU and SSCP-LU sessions proceeds normally.

23.

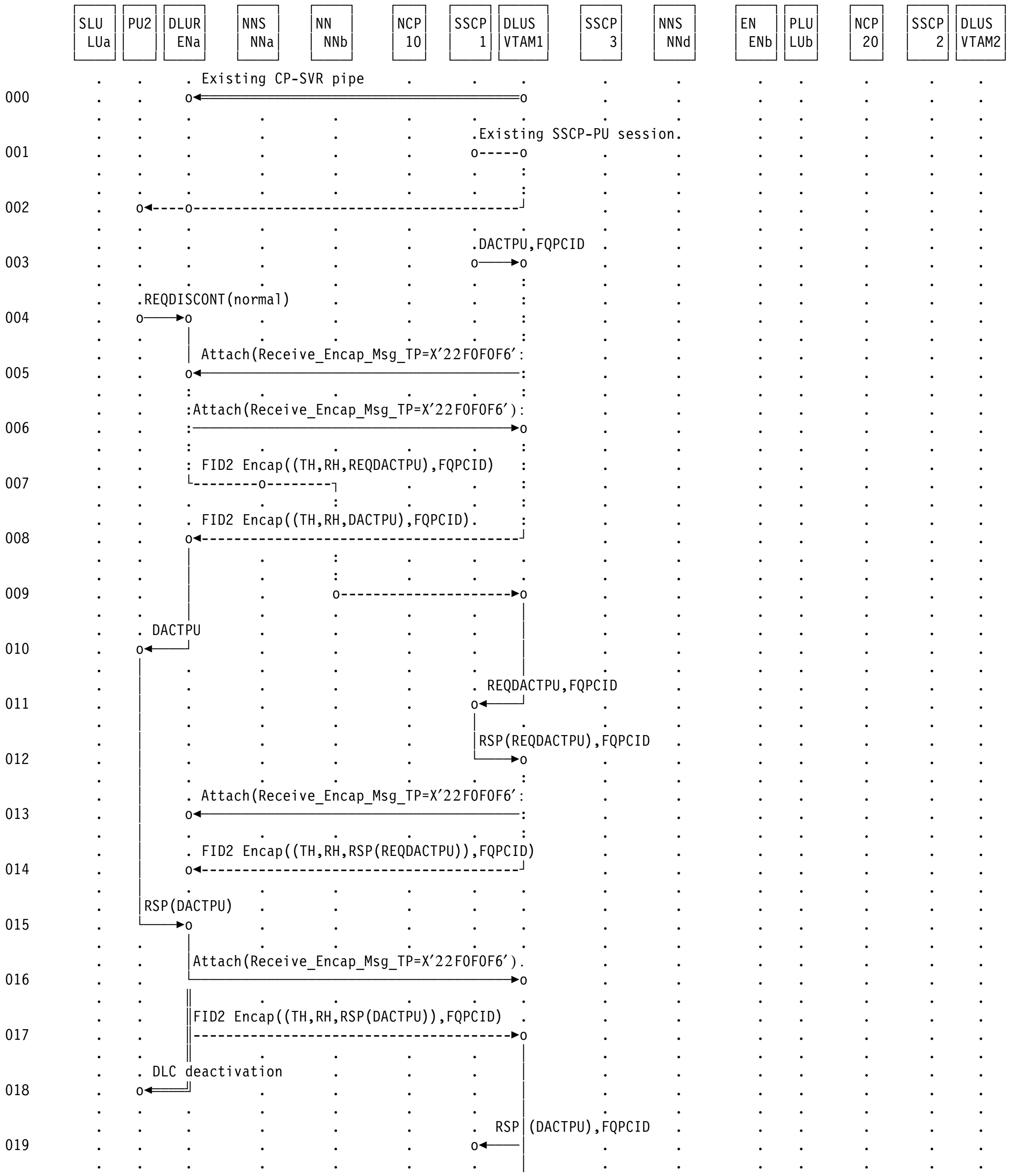


Figure 5-20. SSCP-PU session deactivation race - DACTPU/REQDACTPU

```
         ┌────┐ ┌───┐ ┌────┐    ┌────┐    ┌────┐    ┌───┐  ┌────┐ ┌─────┐    ┌────┐    ┌────┐    ┌────┐ ┌───┐   ┌───┐  ┌────┐ ┌─────┐
         │SLU │ │PU2│ │DLUR│    │NNS │    │NN  │    │NCP│  │SSCP│ │DLUS │    │SSCP│    │NNS │    │EN  │ │PLU│   │NCP│  │SSCP│ │DLUS │
         │ LUa│ │   │ │ ENa│    │ NNa│    │ NNb│    │ 10│  │   1│ │VTAM1│    │   3│    │ NNd│    │ ENb│ │LUb│   │ 20│  │   2│ │VTAM2│
         └────┘ └───┘ └────┘    └────┘    └────┘    └───┘  └────┘ └─────┘    └────┘    └────┘    └────┘ └───┘   └───┘  └────┘ └─────┘
           .      .     . UNBIND .          .         .       .       │          .         .          .     .       .       .       .
020        .      .     o◄──────────────────────────────────────────┘          .         .          .     .       .       .       .
           .      .     ║        .          .         .       .       │          .         .          .     .       .       .       .
           .      .     ║ UNBIND .          .         .       .       │          .         .          .     .       .       .       .
021        .      .     o◄──────────────────────────────────────────┘          .         .          .     .       .       .       .
           .      .     ║        .          .         .       .       .          .         .          .     .       .       .       .
           .      .     ║Remaining CP-SVR pipe deactivation flows.         .         .          .     .       .       .       .
022        .      .     ╚═══════════════════════════════════════════►o          .         .          .     .       .       .       .
           .      .     .        .          .         .       .       .          .         .          .     .       .       .       .
```

1. There are active sessions between SSCP1 and PU2 and SSCP1 and LUa. These are the only active SSCP-PU and SSCP-LU sessions using the CP-SVR pipe between VTAM1 and ENa.
2.
3. SSCP1 initiates normal deactivation of its session with PU2 by building a DACTPU and passing it on to VTAM1.
4. ENa receives a REQDISCONT from PU2.
5. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
6. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
7. To signal to SSCP1 that its SSCP-PU session with PU2 should be deactivated, ENa builds a REQDACTPU with a cause value of X′01′, encapsulates it, and forwards it to VTAM1 on its CP-SVR pipe.
8. The DLUS VTAM1 encapsulates the DACTPU and forwards it to the DLUR ENa.
9.
10. ENa de-encapsulates the DACTPU and forwards it to PU2; this is done as part of the SSCP-PU session deactivation race condition processing - since the requester is in an awaiting RSP(REQDACTPU) state and it receives a DACTPU from the DLUS for that PU, it should accept and process that DACTPU request.
11. VTAM1 de-encapsulates the REQDACTPU and forwards it to SSCP1, having also recognized the SSCP-PU session deactivation race condition.
12. SSCP1 responds to the REQDACTPU positively, passing the RSP(REQDACTPU) on to VTAM1 and remaining in a DACTPU processing state.
13. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
14. VTAM1 encapsulates the RSP(REQDACTPU) and forwards it to ENa; since it is still processing the DACTPU, the +RSP(REQDACTPU) will be accepted and DACTPU processing will continue.
15. PU2 deactivates the SSCP-PU session, converts the DACTPU into a positive RSP(DACTPU), and forwards it to ENa.
16. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
17. ENa then encapsulates the RSP(DACTPU) and sends it to VTAM1.
18. ENa initiates DLC deactivation to PU2.
19. VTAM1 de-encapsulates the RSP(DACTPU) and forwards it to SSCP1.
20. VTAM1 also determines that there are now no active or pending SSCP-PU sessions on its CP-SVR pipe to ENa, so it sends an UNBIND to ENa on VTAM1′s conwinner session to begin deactivation of the pipe.
21. VTAM1 also sends an UNBIND on its conloser session to ENa.
22. The DLUR node sends a RSP(UNBIND) to the DLUS for the conwinner session and another RSP(UNBIND) for the conloser session. For more details on CP-SVR pipe deactivation see 5.6, “CP-SVR Pipe Deactivation” on page 5-20.

5.8 Persistent CP-SVR Pipes

Normally, CP-SVR pipe activation/deactivation architecture requires:

- the DLUR and DLUS to activate a CP-SVR pipe only when a PU requires activation, and
- the DLUS to deactivate a CP-SVR pipe when there are no pending or active SSCP-PU sessions associated with it.

There are requirements, e.g., that DLUR′s duplicate Token-Ring Interface Coupler (TIC) support must match NCP′s, which mandate that some CP-SVR pipes be allowed to be activated and to remain active even when there are no associated PU or LU sessions.

For those situations when a CP-SVR pipe must be activated without an associated PU requiring activation, or when a CP-SVR pipe must remain active after all associated PUs have been deactivated, a ***persistent*** CP-SVR pipe can be used.

There remains a limit of one active CP-SVR pipe per DLUS/DLUR pair, and that pipe may or may not be persistent. If a pipe is activated as a persistent pipe, it cannot be changed while active to a non-persistent pipe, or vice versa.

5.8.1 CP-SVR Pipe Persistence Indicator

To indicate support for persistent CP-SVR pipes, the DLUR and DLUS will set a DLUR/S capabilities indicator in CV X′51′ which

- when set by the DLUS indicates support for CP-SVR pipe persistence in general
- when set by the DLUR indicates a request for CP-SVR pipe persistence for this particular pipe

Therefore, a DLUS which can support persistent pipes will always set the indicator on while a DLUR which can support persistent pipes will only set the indicator on for those pipes it wishes to be persistent pipes.

5.8.2 Persistent CP-SVR Pipe Activation

If the CP-SVR pipe persistence indicator is set by both the DLUS and the DLUR during pipe activation, then the CP-SVR pipe is considered persistent.

DLUR processing will vary depending on whether or not there is a PU needing service and whether the DLUS supports (uplevel) or does not support (downlevel) pipe persistence.

5.8.2.1 Persistent CP-SVR Pipe Activation - PU Needing Service

A DLUR which supports CP-SVR pipe persistence can activate a persistent CP-SVR pipe if there is an associated PU needing activation, just as long as there is no currently active pipe to that DLUS. In this case the DLUR will include a REQACTPU RU in the FID2 Encapsulation (X′1500′) GDS variable it sends to the DLUS during DLUR-initiated CP-SVR pipe activation. The DLUR will send a GDS X′1500′ with the RU and a CV X′51′ indicating support of CP-SVR pipe persistence.

5.8.2.1.1 PU Needing Service - Uplevel DLUS: A DLUS which supports CP-SVR pipe persistence will accept a FID2 Encapsulation (X′1500′) GDS variable with a REQACTPU RU when the CV X′51′ indicates support of CP-SVR pipe persistence, and the DLUS will respond with a GDS X′1500′ with a RSP(REQACTPU) RU and a CV X′51′ indicating support of CP-SVR pipe persistence.

5.8.2.1.2 PU Needing Service - Downlevel DLUS: A DLUS which does not support CP-SVR pipe persistence will accept a FID2 Encapsulation (X′1500′) GDS variable with a REQACTPU RU when the CV X′51′ indicates support of CP-SVR pipe persistence, and the DLUS will respond with a GDS X′1500′ with a RSP(REQACTPU) RU and a CV X′51′ indicating non-support of CP-SVR pipe persistence. At this point the DLUR can either:

- accept the non-persistent pipe and continue session activation, or
- it can terminate the pipe, using DLUR/S capabilities mismatch processing described in Figure 5-9 on page 5-38 in 5.6.2.2.2, "DLUR-Detected Capabilities Mismatch (DLUS-DLUR Pipe)" on page 5-37, and attempt to activate a persistent pipe with a different DLUS.

5.8.2.2 Persistent CP-SVR Pipe Activation - No PU Needing Service

A DLUR which supports CP-SVR pipe persistence can activate a persistent CP-SVR pipe even if there is no associated PU requiring activation, just as long as there is no currently active pipe to that DLUS. In this case the DLUR will not be required to include a REQACTPU RU in the FID2 Encapsulation (X′1500′) GDS variable it sends to the DLUS during DLUR-initiated CP-SVR pipe activation. The DLUR will send a GDS X′1500′ without an RU and with a CV X′51′ indicating support of CP-SVR pipe persistence.

5.8.2.2.1 No PU Needing Service - Uplevel DLUS: A DLUS which supports CP-SVR pipe persistence will accept a FID2 Encapsulation (X′1500′) GDS variable without an RU when the CV X′51′ indicates support of CP-SVR pipe persistence, and the DLUS will respond with a GDS X′1500′ without an RU and with a CV X′51′ indicating support of CP-SVR pipe persistence.

5.8.2.2.2 No PU Needing Service - Downlevel DLUS: A DLUS which does not support CP-SVR pipe persistence will, upon receipt of a FID2 Encapsulation (X′1500′) GDS variable without an RU, deactivate the CP-SVR pipe using UNBIND-initiated forced deactivation with the following sense data:

- X′08A0 000C′ DLUS-DLUR Session Deactivation (Disruptive): The DLUS received a CP-SVR pipe activation request without a REQACTPU; DLUR should activate a DLUS-DLUR session with a different DLUS if possible.

(For more information about UNBIND-initiated CP-SVR pipe deactivation, see 5.6.2.1, "UNBIND-Initiated CP-SVR Pipe Deactivation" on page 5-30)

A DLUR receiving an UNBIND with this sense data will understand that it attempted to activate a persistent pipe with a downlevel DLUS and will then try to activate a persistent pipe with another DLUS (if any).

Downlevel DLUSs will be PTF′ed (APAR OW34062) to send the X′08A0 000C′ sense data.

5.8.3 Persistent CP-SVR Pipe Deactivation

When the number of current and pending SSCP-PU sessions associated with a persistent CP-SVR pipe reaches zero, the DLUS will leave the pipe active.

The DLUS can deactivate a persistent CP-SVR pipe using UNBIND-initiated forced deactivation with disruptive or nondisruptive sense data as appropriate.

If the number of current and pending SSCP-PU sessions is zero, the DLUR can deactivate a persistent CP-SVR pipe using UNBIND-initiated forced deactivation with the following persistent sense data:

- X′08A0 000D′ DLUS-DLUR Session Deactivation (Persistent): The DLUR is initiating deactivation of a persistent DLUS-DLUR session.

(For more information about UNBIND-initiated CP-SVR pipe deactivation, see 5.6.2.1, “UNBIND-Initiated CP-SVR Pipe Deactivation” on page 5-30)

5.9 CP-SVR Pipe Performance Considerations

5.9.1 Pacing Considerations

It is recommended that DLUR products close their network priority pacing windows last (instead of the more typical simultaneous closing of all pacing windows regardless of priority). Since SSCP-PU/LU session activation and deactivation commands flow at network priority, they should be allowed to flow as long as possible so that they can be used to help reduce network congestion.

5.10 Additional Error Cases

5.10.1 CP-SVR Pipe Limit Exceeded

DLUR products will have the option to limit the number of concurrently active CP-SVR pipes (what number will be selected as the limit will also be a product option). If a limit is implemented and reached, any attempts to activate additional CP-SVR pipes will be rejected by sending an UNBIND request with sense data X′0812 0000′.

5.10.2 GDS Variable Other Than X′1500′ Received After Receive_Encap Attach

If the Receive_Encap_Msg_TP is attached and the GDS variable received is not X′1500′, the DLUR will send an UNBIND request with sense data X′1001 0000′.

5.10.3 CPSVRMGR Session Over Non-TCP/IP LEN Connection

If a CPSVRMGR session is established over a LEN connection which is not a connection into a TCP/IP network, the DLUS will send an UNBIND request with sense data X′0877 0056′.

Chapter 6. SSCP-PU/SSCP-LU Session Encapsulation

Once the CP-SVR pipe has been established between server and requester nodes, SSCP-PU and SSCP-LU sessions must be established to provide SSCP services to the dependent LUs. These sessions will be encapsulated on the CP-SVR pipe. This will require a special encapsulation and de-encapsulation transaction program (TP) on either side of the conwinner and conloser CPSVRMGR sessions. These TPs are known as the Send_Encap_Msg_TP and Receive_Encap_Msg_TP respectively. Since the Receive_Encap_Msg_TP must be attached by the Send_Encap_Msg_TP, it has been assigned X′22F0F0F6′ as its architected TP name.

6.1 Encapsulation Addressing

The information carried on the SSCP-PU and SSCP-LU sessions will be encapsulated in a new FID2 Encapsulation GDS variable X′1500′. This GDS variable will contain the entire FID2 message that is normally carried between the SSCP Boundary Function and peripheral node. This includes the Transmission Header (TH), Request/Response Header (RH), and Request Unit (RU) fields (the RH will not be present in a FID2 message which is a middle or last segment - see 6.1.4, “Encapsulating Message Segments” on page 6-6 for more information about segmentation of messages on the CP-SVR pipe). The TH field will provide the appropriate PU and LU addressing as it does today, and the RH and RU fields will enable the PU to interpret the RUs and function the same way in both the DLUR/S and SSCP subarea environments. Because a given CP-SVR pipe may support multiple SSCP-PU sessions, it will be important to distinguish which SSCP-PU/LU flows go with which PU image. This will be done by carrying a Fully Qualified Procedure Correlator Identifier (FQPCID) control vector (CV X′60′) on each FID2 Encapsulation GDS variable. The FQPCID will be generated by the node creating the first encapsulated flow. This will be the DLUR node when REQACTPU is used, and it will be the DLUS node when an unsolicited ACTPU is sent. Once assigned, the CV X′60′ flows on each ensuing encapsulated flow having to do with that particular PU image (or its dependent LUs). This FQPCID essentially extends the ability for a single node to support more than 255 dependent LUs by providing addressing to multiple PU images. In addition, the FQPCID provides the DLUS with a mechanism to quickly index resource-related information for control flows. Because the FQPCID is being used in this manner, it is essential that it is unique for each DLUS node. Since the FQPCID is fully qualified, this implies that each node must ensure that it does not generate duplicate active FQPCIDs.

An FQPCID includes a PCID and a network-qualified CP name, and it is used in conjunction with an optional PCID modifier. The network-qualified CP name will be defined as follows:

- DLUS-generated - the network-qualified SSCP name
- DLUR-generated - the network-qualified CP(DLUR) name

6.1.1 Multiple PU Images

As mentioned above, it is possible for a DLUR node to support more than 255 dependent LUs. In this case, it will be necessary for DLUR nodes to emulate multiple PU images. PU images that are being served by the same SSCP (DLUS node) will share the CP-SVR pipe. It is possible, however, for different PU images to be served by different SSCPs (DLUS nodes). In this case, a given DLUR node may have multiple CP-SVR pipes to multiple DLUS nodes. In either case, a given SSCP-PU and its subsequent SSCP-LU sessions will all be encapsulated on the same CP-SVR pipe. When a DLUR node is supporting multiple CP-SVR pipes to multiple DLUS nodes, the DLUR node may (or may not) run multiple instances of the Send_Encap_Msg_TP and the Receive_Encap_Msg_TP; it must keep track of which PU images are using which CP-SVR pipes.

6.1.2 Internal PU Identification

(***Note*** - As described in 5.3.1, “DLUR PU Identification” on page 5-2, a DLUR can identify a PU using a PU TG descriptor containing a PU name, signaling information, or both. The following section pertains to defining signaling information for an internal PU.)

Whether a DLUR node has a single PU or multiple PU images, if any of its PUs are internal (not down-stream PUs), each internal PU will be uniquely identified by either the DLUR′s CP name or the internal PU′s own IDBLK/IDNUM field. A unique identifier is necessary to identify which internal PU is being activated; therefore, at most one internal PU per DLUR can be optionally identified by the DLUR′s CP name instead of IDBLK/IDNUM in the PU TG field.

SF X′92′ will carry IDBLK/IDNUM and SF X′93′ will carry CP name. The SF combinations that will be permitted when SF X′91′ = INTPU are:

Table 6-1. Internal PU Identification

Option	**SF X′92′**	**SF X′93′**
1	internal PU′s IDBLK/IDNUM	(not included)
2a	(not included)	CP name
2b	CP′s IDBLK/IDNUM	CP name

6.1.2.1 Internal PU Identification Option 1

The internal PU′s IDBLK/IDNUM will be carried in the PU TG field (CV X′4691′ = INTPU, CV X′4692′ = IDBLK/IDNUM) in the GDS variable X′1500′ for:

- REQACTPU and ACTPU when PU activation is DLUR-initiated
- ACTPU when PU activation is DLUS-initiated

This IDBLK/IDNUM may also be carried in the XID field (X′81′) in the GDS variable X′1500′ for:

- REQACTPU when PU activation is DLUR-initiated
- RSP(ACTPU) when PU activation is DLUS-initiated

No CV X′4693′ will be included.

6.1.2.2 Internal PU Identification Option 2a

The DLUR′s CP name will be carried in the PU TG field (CV X′4691′ = INTPU, CV X′4693′ = CP name) in the GDS variable X′1500′ for:

- REQACTPU and ACTPU when PU activation is DLUR-initiated
- ACTPU when PU activation is DLUS-initiated

When a CP name is carried in the PU TG field, the DLUR CP′s IDBLK/IDNUM may be carried in the XID field (CV X′81′) in the GDS variable X′1500′ for:

- REQACTPU when PU activation is DLUR-initiated
- RSP(ACTPU) when PU activation is DLUS-initiated

No CV X′4692′ will be included.

6.1.2.3 Internal PU Identification Option 2b

The DLUR CP′s IDBLK/IDNUM and CP name will be carried in the PU TG field (CV X′4691′ = INTPU, CV X′4692′ = IDBLK/IDNUM, CV X′4693′ = CP name) in the GDS variable X′1500′ for:

- REQACTPU and ACTPU when PU activation is DLUR-initiated
- ACTPU when PU activation is DLUS-initiated

When a CP name is carried in the PU TG field, the DLUR CP′s IDBLK/IDNUM may be carried in the XID field (CV X′81′) in the GDS variable X′1500′ for:

- REQACTPU when PU activation is DLUR-initiated
- RSP(ACTPU) when PU activation is DLUS-initiated

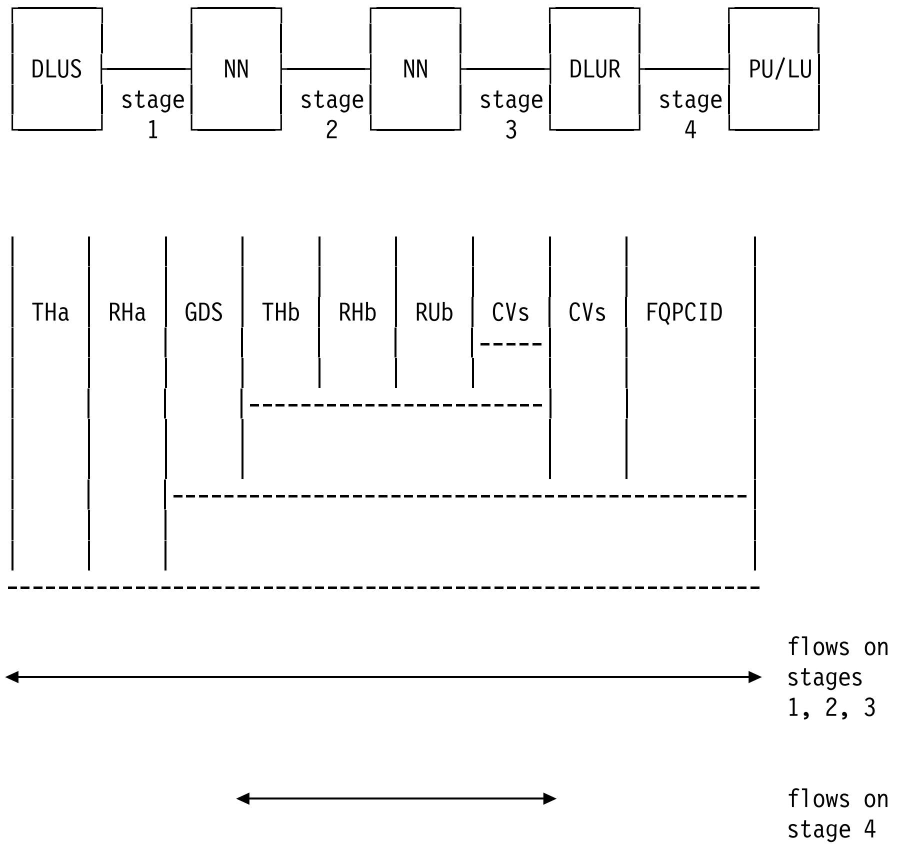
When this PU TG field is received, it will be compared against defined PUs for a match. See 5.3.1, “DLUR PU Identification” on page 5-2 and 5.4.1, “New PU Definition Parameters” on page 5-6 for details about the search order used by a DLUR to find a match.

6.1.3 Addressing Per Stage

Much of this is derived from the Address Space Manager (ASM) chapter in *SNA APPN Architecture Reference*.

Local Form Session Identifiers (LFSIDs) are the addresses carried in FID2 THs; these addresses have three parts: Session Identifier High (SIDH) (1 byte), Session Identifier Low (SIDL) (1 byte), Origin Destination Assignment Indicator (ODAI) (1 bit).

6.1.3.1 SSCP-PU And SSCP-LU Sessions



SSCP-PU	THa	THb
SIDH	X'02'-X'FE'	X'00'
SIDL	X'00'-X'FF'	X'00'
ODAI	negotiated during XID exchange	0

SSCP-LU	THa	THb
SIDH	X'02'-X'FE'	X'00'
SIDL	X'00'-X'FF'	X'01'-X'0F'
ODAI	negotiated during XID exchange	0

Figure 6-1. Addressing For SSCP-PU And SSCP-LU Sessions

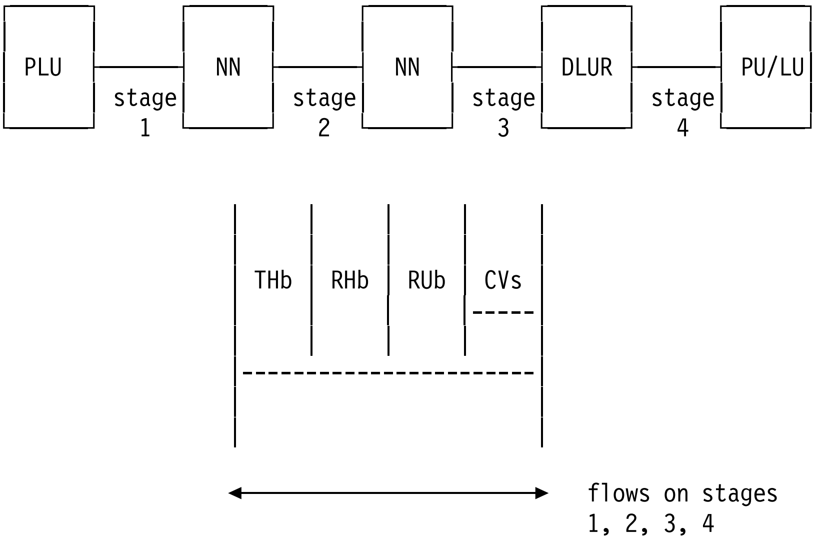
At the DLUR and DLUS nodes, the FQPCID is used to identify the proper PU.

The SIDL value in THb is used to identify the proper LU.

LFSIDs for THa are independently set for each stage and for each direction; in the configuration shown, there will be six different THa LFSIDs for a message sent from DLUS to DLUR and back to DLUS.

The ODAI bit is set to zero in THb on stage 4 because the DLUR node must always be the primary link station on the stage 4 connection. DLUR nodes should always negotiate to be the primary link station in relation to DLUR-supported downstream PUs.

6.1.3.2 LU-LU Sessions



LU-LU THb	stages 1, 2, 3	stage 4
SIDH	X'02'-X'FE'	X'01'
SIDL	X'00'-X'FF'	X'01'-X'FF'
ODAI	negotiated during XID exchange	0

Figure 6-2. Addressing For LU-LU Sessions

For stages 1, 2, and 3, LFSIDs for THb are independently set for each stage and for each direction; in the configuration shown, there will be six different THb LFSIDs for a message sent from PLU to DLUR and back to PLU. For stage 4, SIDL should be the same value used for the SIDL in THb for that LU's SSCP-LU session.

6.1.4 Encapsulating Message Segments

The DLUR can encapsulate FID2 PIU segments in separate GDS X'1500's. The only PIUs which can be sent segmented are FMD requests, FMD responses with QRI=1, and non-expedited DFC requests. The segments will be reassembled at the DLUS.

6.1.5 Blocking Multiple FID2 Encapsulations

The DLUR can send or receive multiple FID2 Encapsulation (X′1500′) GDS variables in a single BIU on the CP-SVR pipe. The DLUR indicates its support of FID2 Encapsulation blocking by setting on byte 9, bit 5 in the DLUR/S Capabilities (X′51′) control vector at CP-SVR pipe activation. The DLUR will perform blocking only if the DLUS also sets the same bit on in its CV X′51′ during CP-SVR pipe activation.

It is a DLUR implementation option to determine when to block; however, no timers should be created for this function - just block if there is more than one GDS X′1500′ to be sent.

6.2 SSCP-PU Session Activation

To support dependent SLUs in or downstream from the DLUR node, the DLUS node must initiate SSCP-PU and SSCP-LU sessions over the CP-SVR pipe to the T2.0 PU image supporting the dependent SLUs.

This begins with the establishment of an SSCP-PU session. Once the DLUS conwinner CPSVRMGR session has been established, an ACTPU message must be sent over the CP-SVR pipe to the T2.0 PU to establish an SSCP-PU session. This can be done in two ways:

1. ***The DLUS node may activate a predefined PU.***

In this case, the SSCP (DLUS node) has a predefined PU image associated with the DLUR node. When the DLUS node explicitly activates the PU, it will know the DLUR name and downstream addressing associated with the PU as defined in 5.4.1, "New PU Definition Parameters" on page 5-6. This enables the DLUS to establish a CP-SVR pipe to the DLUR. Once the CP-SVR pipe has been activated, the SSCP will use the system-defined information and knowledge that the CP-SVR pipe has been activated, to generate an ACTPU and send it to the downstream PU.

2. ***The DLUR node may request dynamic PU activation via a REQACTPU.***

REQACTPU is a new message that identifies to the DLUS node that the DLUR node is requesting an ACTPU be sent for a previously defined or undefined downstream PU. The DLUS node will use information contained in the REQACTPU (see 5.3.1, "DLUR PU Identification" on page 5-2 for more details) to search for a predefined PU definition or use the Configuration Services XID exit support to create an ACTPU and send it to the downstream PU.

In either case, the FID2 Encapsulation GDS variable will be used to transport the REQACTPU|ACTPU messages over the CP-SVR pipe. In order to do this, the FQPCID and FID2 TH, RH, and RU fields must be properly assigned. The flows in this section describe how these fields are assigned and the PU images are activated in both cases.

Data link control (DLC) activation must precede the forwarding of the ACTPU to the PU. In a subarea network, DLC activation may require several RU flows, e.g., ACTLINK, CONNOUT, CONTACT, CONTACTED. The DLUR will perform DLC activation function, but without the receipt of these DLC activation commands. This activation will either be triggered by operator command, external contact from the PU, or by the receipt of an ACTPU from the DLUS. So that the DLUR has the necessary information to perform DLC activation, the REQACTPU and the ACTPU will be encapsulated with XID (X′81′) and TG Descriptor (X′46′) control vectors (both of which the DLUR will process before passing the ACTPU on to the PU). The XID control vector is identified in the flows as XID, and the TG Descriptor control vector as PU TG.

The DLUS will include the PU name with the ACTPU RU (the name will be defined in a Network Name (X′0E′) control vector type X′F1′. This name will be network-qualified if the PU′s NETID is different than the SSCP′s NETID. Unless it is indicated in the ACTPU to forward the PU name (for more details about PU name forwarding, see 6.2.4, "PU/LU Network Name Forwarding" on page 6-25), the DLUR will strip off this control vector before passing the ACTPU on to the PU; in either case, it will store the PU name for use in network management.

In a subarea network, activation of an SSCP-PU session may require several RU flows, e.g., Request Network Address Assignment (RNAA), SETCV, and ACTPU. For a DLUR-attached PU, these RUs will be combined into one, ACTPU. Therefore, ACTPU will also be encapsulated with an Extended SDLC Station (X′43′) control vector, which the DLUR will process before passing the ACTPU on to the PU. The control vector is identified in the flows as PU CHAR.

6.2.1 Predefined PU Activation

In Figure 6-3 on page 6-10, the DLUS node has a predefined PU definition that it wishes to activate. In Figure 6-4 on page 6-12, the PU definition has been dynamically created and the DLUS operator has issued a command to activate/reactivate the PU. In this case, the PU definition must have the new DLUR and PU TG parameters as described in 5.4.1, “New PU Definition Parameters” on page 5-6.

Notes:

1. These scenarios assume the CP-SVR pipe between DLUS and DLUR is already active and has been activated by a previous SSCP-PU session activation sequence. Reference Figure 5-2 on page 5-15 for flows to describe the CP-SVR pipe activation associated with predefined PU activation.
2. Refer to Table 3-1 on page 3-2 for an explanation of the notation used in the diagrams in this chapter.

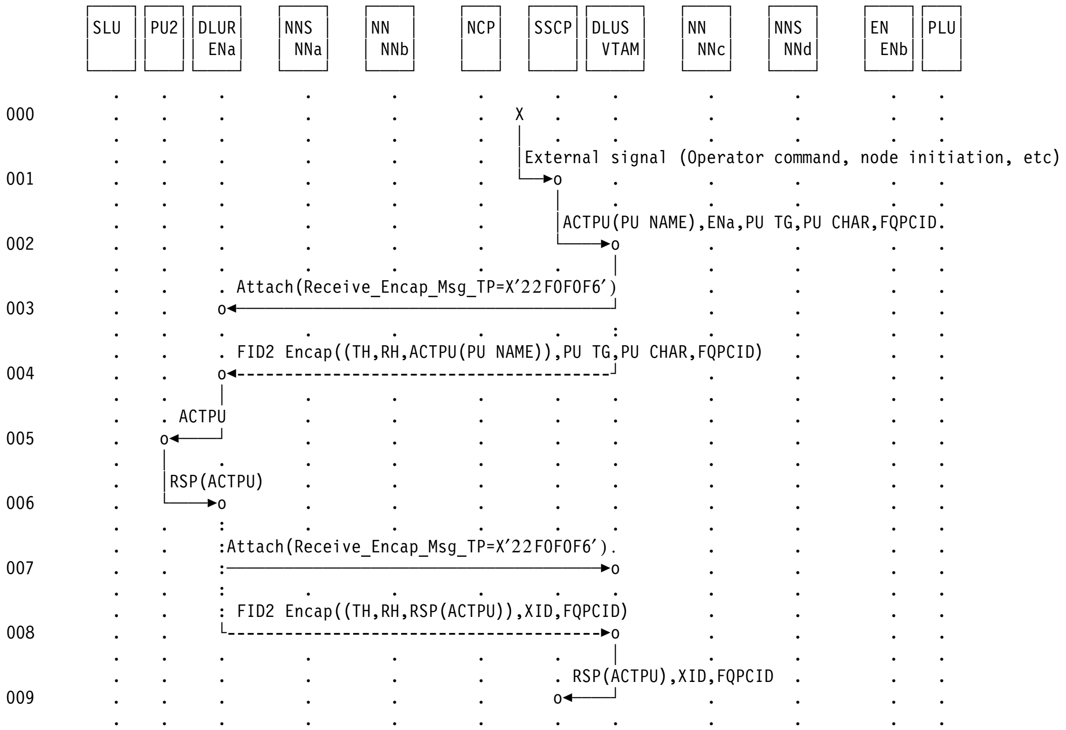


Figure 6-3. Predefined PU activation

1. When an external operator command or internal process signals to the SSCP that a PU should be activated, a signal is sent to the SSCP to generate the associated ACTPU.
2. The ACTPU is sent from the SSCP to DLUS component of the NN. The ACTPU includes the name of the PU, as well as the FQPCID generated by the SSCP to identify and index the PU control block.
3. The DLUS component uses the DLUR node specification in the PU definition to determine which CP-SVR pipe the encapsulated ACTPU should flow on. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
4. The ACTPU FID2 PIU is sent on the CP-SVR pipe to the DLUR node. The FQPCID generated by the SSCP to identify the PU is included in the FID2 Encapsulation GDS variable. The TH of the FID2 will address both the SSCP and PU in the origin address field (OAF) and destination address field (DAF) respectively. The DLUS node includes the PU CHAR field with the ACTPU; this field carries PU information such as data mode and retry limits. The DLUS node also includes the PU TG identifier with the ACTPU RU; this will be used by the DLUR node to identify which PU is being activated.
5. The DLUR node receives the encapsulated ACTPU command. It must look in the FID2 Encapsulation GDS variable for the PU TG vector to identify the PU being activated. If the PU is downstream, the PU TG will identify a physical link to the downstream node. The DLUR node may have to initiate DLC activation on the link to the PU; the PU TG and PU CHAR fields carry information needed to activate the DLC. If the PU is internal to the DLUR, the PU TG is used as a local correlator to the DLUR to identify the PU image. It must then store the FQPCID associated with this PU so that subsequent data flows on the SSCP-PU session are forwarded to the correct PU. The DLUR strips off the GDS variable, including the PU TG, PU CHAR, and FQPCID fields, removes the PU name (storing it for network management) unless it is indicated in the ACTPU to forward the PU name (for more details about PU name forwarding, see 6.2.4, “PU/LU Network Name Forwarding” on page 6-25), and forwards the ACTPU to the appropriate PU.
6. The PU generates a RSP(ACTPU) and sends it to the DLUR component.
7. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
8. The DLUR component uses the previously stored FQPCID to return the RSP(ACTPU) in a FID2 Encapsulation GDS variable on the CP-SVR pipe. This FQPCID is supplied to allow the DLUS to quickly access the PU control block.
9. The DLUS node receives the encapsulated RSP(ACTPU) and strips off the GDS variable. The RSP(ACTPU) is forwarded from the DLUS to the SSCP component. The FQPCID is now used on all subsequent SSCP-PU data flows to identify the PU to the DLUR. Both the DLUS and DLUR node will set this field to the same value when referring to the SSCP-PU session for this PU.

6.2.2 Dynamic PU Activation

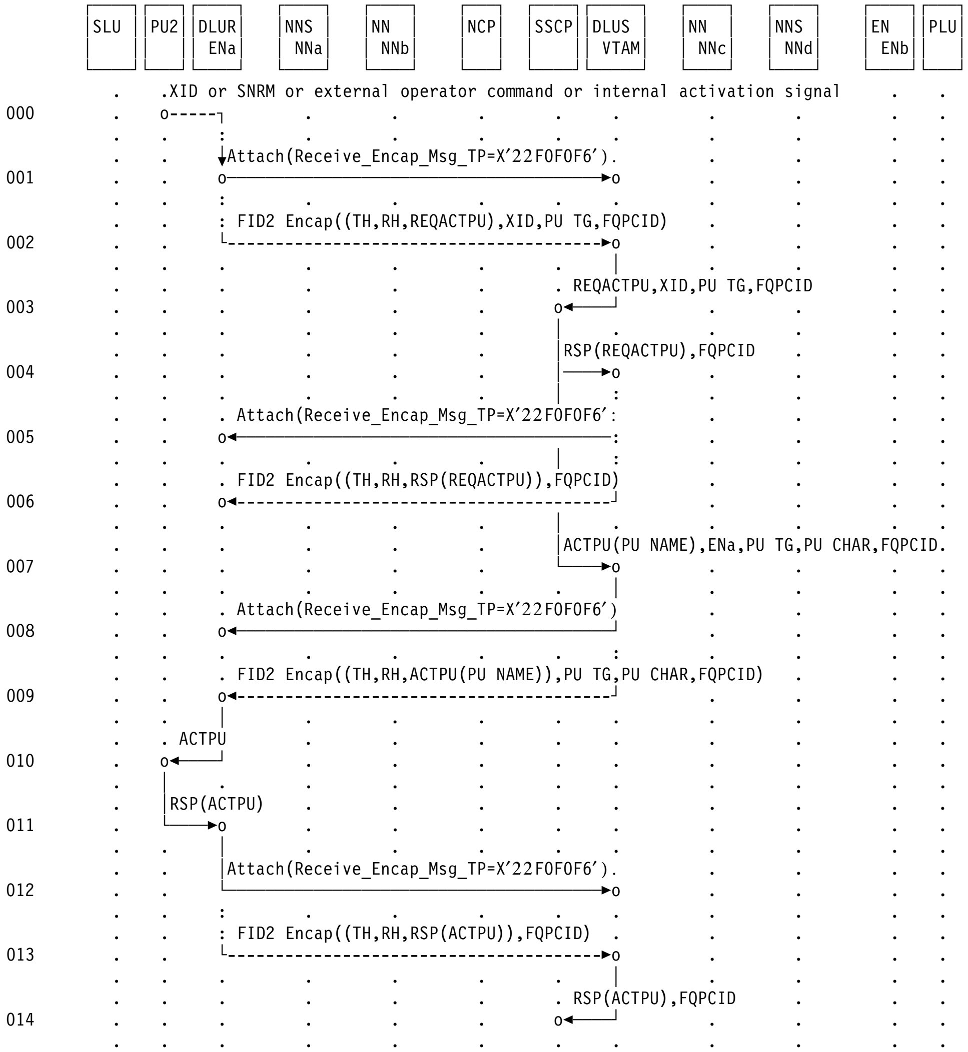


Figure 6-4. Dynamic PU activation

1. Sometime after activation of the CP-SVR pipe between DLUS and DLUR nodes, the DLUR node indicates that it wishes to activate a PU. This could be in response to an operator command, internal signal, or external contact from a downstream PU (for cases where the CP-SVR pipe has not already been established, see Figure 5-1 on page 5-10). The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.

2. The DLUR signals the desire to activate a PU by sending an encapsulated REQACTPU on the CP-SVR pipe. The DLUR node will generate an FQPCID to identify the PU to the SSCP. It will also remember this FQPCID to identify the specific PU instance (and associated PU TG information if applicable) to itself. The TH DAF network address field is set to 0 identifying the SSCP, and the OAF address is set to 0 identifying the PU. The FID2 Encapsulation GDS variable carrying the REQACTPU may contain an XID0, XID1, or XID3 that will be used by the SSCP to search for a pre-defined PU image. In the case where a downstream PU has exchanged XID′s with the DLUR, this will be the true XID image. In the case where this has not taken place, the DLUR will generate an XID0 image as a means to supply a IDBLK/IDNUM value to the SSCP. The FID2 Encapsulation GDS variable carrying the REQACTPU also may carry a PU TG identifier to describe the link attaching the downstream PU to the DLUR.

3. The DLUS component receives the encapsulated REQACTPU and passes it and the DLUR-generated FQPCID to the SSCP after stripping off the FID2 Encapsulation GDS variable.

4. The SSCP must verify that the FQPCID generated by the DLUR is unique and the PU can be activated. A RSP(REQACTPU) is then generated and sent to the DLUR node via the DLUS component. Before sending a positive RSP(REQACTPU), the SSCP must verify that the FQPCID is unique for this node and either find a predefined PU definition or generate a PU definition via the Configuration Services XID Exit support mentioned in 3.2.2, “Configuration Services XID Exit Support” on page 3-5. The SSCP may also chose to reject the REQACTPU regardless of Configuration Services XID Exit support, for local security or other product/customer-specific reasons. If the above conditions are met, the PU can be activated, and a positive response is generated. If the FQPCID is not unique or the PU cannot be activated, a negative response and sense data is generated. The RSP(REQACTPU) is sent along with the FQPCID to the DLUS component of the DLUS node.

5. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.

6. The DLUS component encapsulates the RSP(REQACTPU) PIU and uses the FQPCID generated by the DLUR component to identify the PU to the DLUR node. This encapsulated message is sent over the CP-SVR pipe to the DLUR node.

7. The SSCP follows a positive RSP(REQACTPU) with an ACTPU for the PU. This ACTPU includes the name of the PU.

8. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.

9. The ACTPU PIU is encapsulated along with the FQPCID to identify the PU to the DLUR node. It is sent to the DLUR node.

10. The DLUR node uses the FQPCID field to identify the PU to receive the encapsulated flow. The FID2 Encapsulation GDS variable and its PU TG and PU CHAR fields are stripped by the DLUR. The DLUR then removes the PU name (storing it for network management) from the ACTPU unless it is indicated in the ACTPU to forward the PU name (for more details about PU name forwarding, see 6.2.4, “PU/LU Network Name Forwarding” on page 6-25), and forwards the ACTPU to the appropriate PU.

11. The PU generates a RSP(ACTPU) and sends it to the DLUR node.

12. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.

13. The DLUR component encapsulates the RSP(ACTPU) PIU along with the FQPCID and sends it on the CP-SVR pipe to the DLUS node.
14. The DLUS component removes the FID2 Encapsulation GDS variable and forwards it to the SSCP. Both the DLUS and DLUR nodes use the FQPCID to uniquely identify this SSCP-PU session.

6.2.3 SSCP-PU Session Activation Race Conditions

When a DLUS and a DLUR attempt to initiate SSCP-PU sessions with the same PU, the DLUS-initiated session will be started and the DLUR-initiated session will be rejected. When two DLUSs attempt to initiate SSCP-PU sessions with the same PU, the first activation request received by the DLUR will be accepted and the second rejected. When a DLUS and a non-DLUS attempt to initiate SSCP-PU sessions with the same PU, the first activation request received by the DLUR will be accepted and the second rejected.

6.2.3.1 Single DLUS Race Conditions

Race conditions can exist between a DLUR and one DLUS, when both are trying to start an SSCP-PU session for the same PU at the same time. The DLUR will send a REQACTPU to the DLUS while the DLUS is sending an ACTPU to the DLUR. The resolution of this situation is for the DLUR to process the ACTPU and for the DLUS to reject the REQACTPU.

Figure 6-5 on page 6-16 illustrates an example of this race condition:

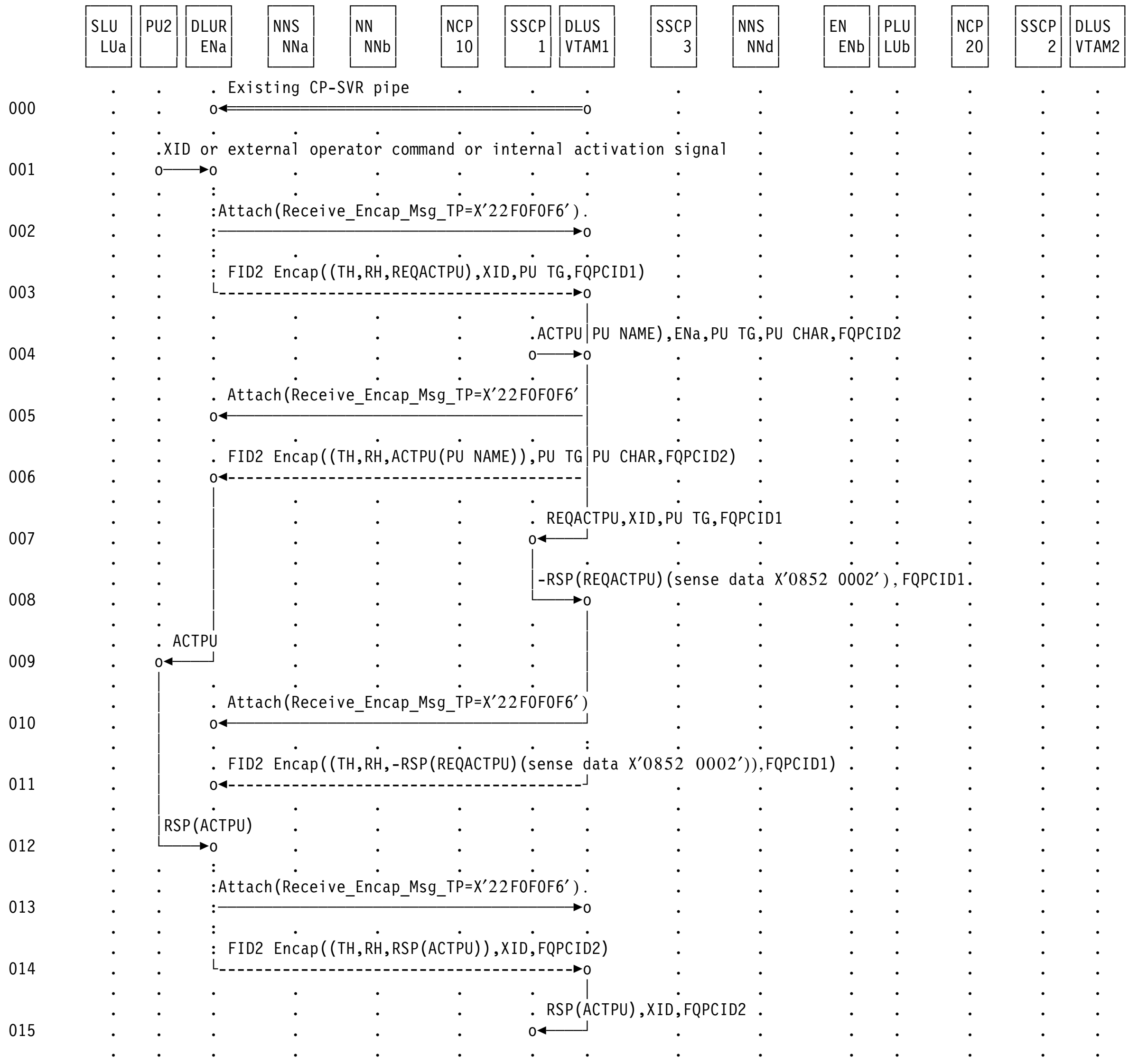


Figure 6-5. SSCP-PU session activation race - DLUR- and DLUS-initiated requests on same CP-SVR pipe

1. There is an active CP-SVR pipe between VTAM1 and ENa. ENa receives some indication that PU2 requires activation.
2. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
3. ENa encapsulates and sends to VTAM1 a REQACTPU.
4. Meanwhile SSCP1 also receives some indication to activate PU2, so it builds an ACTPU to be sent to PU2.
5. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
6. VTAM1 encapsulates the ACTPU and sends it to ENa.
7. VTAM1 de-encapsulates the REQACTPU and forwards it to SSCP1.
8. SSCP1 recognizes that it already has a pending SSCP-PU session with PU2, so it negatively responds to the REQACTPU with sense data X′0852 0002′.
9. ENa recognizes that it already has a pending SSCP-PU session with PU2. Knowing that SSCP1 will reject the REQACTPU, ENa de-encapsulates the ACTPU and forwards it to PU2.
10. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
11. VTAM1 encapsulates the -RSP(REQACTPU) and sends it to ENa.
12. PU2 returns a positive RSP(ACTPU) to SSCP1 via ENa.
13. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
14. ENa encapsulates the RSP(ACTPU) and sends it to VTAM1.
15. VTAM1 de-encapsulates the RSP(ACTPU) and passes it on to SSCP1.

6.2.3.2 Multi-DLUS Race Conditions

Race conditions can exist between a DLUR and two DLUSs; the rule the DLUR applies to resolve these races is the same: **always accept the first ACTPU received**. Here′s how this rule is applied in different race conditions:

- ***race between DLUS-initiated and DLUR-initiated session activations*** - In this case the DLUR will send a REQACTPU to one DLUS while the other DLUS is sending an ACTPU to the DLUR for the same PU. The resolution of this situation is for the DLUR to process the DLUS-initiated ACTPU and reject the ACTPU received in response to the REQACTPU.
- ***race between two DLUS-initiated session activations*** - In this case both DLUSs will send ACTPUs to the DLUR for the same PU. The resolution of this situation is for the DLUR to both process the first received DLUS-initiated ACTPU and reject the second ACTPU.

Figure 6-6 on page 6-19 through Figure 6-7 on page 6-22 illustrate examples of these race conditions:

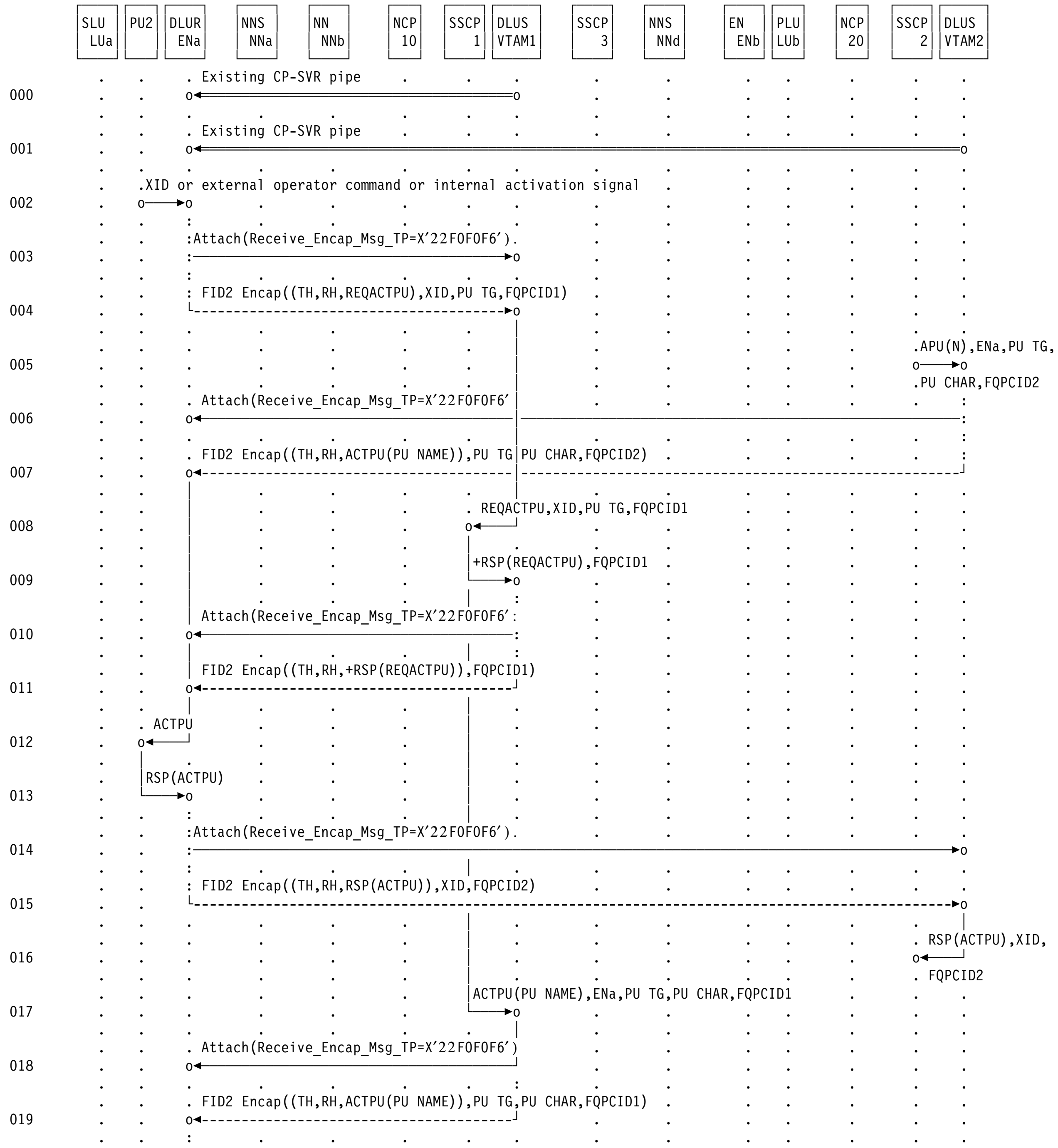


Figure 6-6. SSCP-PU session activation race - DLUR- and DLUS-initiated requests on different CP-SVR pipes

```
      ┌────┐ ┌───┐ ┌────┐   ┌────┐   ┌────┐    ┌───┐  ┌────┐ ┌─────┐  ┌────┐   ┌────┐    ┌────┐ ┌───┐  ┌───┐  ┌────┐ ┌─────┐
      │SLU │ │PU2│ │DLUR│   │NNS │   │NN  │    │NCP│  │SSCP│ │DLUS │  │SSCP│   │NNS │    │EN  │ │PLU│  │NCP│  │SSCP│ │DLUS │
      │ LUa│ │   │ │ ENa│   │ NNa│   │ NNb│    │ 10│  │   1│ │VTAM1│  │   3│   │ NNd│    │ ENb│ │LUb│  │ 20│  │   2│ │VTAM2│
      └────┘ └───┘ └────┘   └────┘   └────┘    └───┘  └────┘ └─────┘  └────┘   └────┘    └────┘ └───┘  └───┘  └────┘ └─────┘
         .     .    :Attach(Receive_Encap_Msg_TP=X'22F0F0F6').     .         .          .     .       .       .      .
020      .     .    :──────────────────────────────────────────►o      .         .          .     .       .       .      .
         .     .    :         .        .        .      .     .      .         .          .     .       .       .      .
         .     .    : FID2 Encap((TH,RH,-RSP(ACTPU)(sense data X'082C 0002')),FQPCID1)      .     .       .       .      .
021      .     .    └- - - - - - - - - - - - - - - - - - - - - ►o      .         .          .     .       .       .      .
         .     .    .         .        .        .      .     │      .         .          .     .       .       .      .
         .     .    .         .        .        .      . -RSP(ACTPU)(sense data X'082C 0002'),FQPCID1 .       .      .
022      .     .    .         .        .        .      o◄────┘      .         .          .     .       .       .      .
         .     .    .         .        .        .      .     .      .         .          .     .       .       .      .
```

1. There are active CP-SVR pipes between VTAM1 and ENa and between VTAM2 and ENa.
2. ENa receives some indication that PU2 requires activation.
3. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
4. ENa encapsulates and sends to VTAM1 a REQACTPU.
5. Meanwhile SSCP2 also receives some indication to activate PU2, so it builds an ACTPU (represented as APU) to be sent to PU2.
6. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
7. VTAM2 encapsulates the ACTPU and sends it to ENa.
8. VTAM1 de-encapsulates the REQACTPU and forwards it to SSCP1.
9. SSCP1 recognizes that it has no active or pending SSCP-PU session with PU2, so it positively responds to the REQACTPU and passes the response to VTAM1.
10. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
11. VTAM1 encapsulates the +RSP(REQACTPU) and sends it to ENa.
12. ENa recognizes that it already has a pending SSCP-PU session with PU2. It must wait for the ACTPU from SSCP1 to reject that session activation attempt. Meanwhile, ENa de-encapsulates the ACTPU from SSCP2 and forwards it to PU2.
13. PU2 returns a positive RSP(ACTPU) to SSCP2 via ENa.
14. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
15. ENa encapsulates the RSP(ACTPU) and sends it to VTAM2.
16. VTAM2 de-encapsulates the RSP(ACTPU) and passes it on to SSCP2.
17. SSCP1 builds an ACTPU for PU2 and passes it to VTAM1.
18. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
19. VTAM1 encapsulates the ACTPU and sends it to ENa.
20. ENa recognizes that there already is an SSCP-PU session with PU2. It negatively responds to the ACTPU with sense data X′082C 0002′. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
21. ENa encapsulates the -RSP(ACTPU) and sends it to VTAM1.
22. VTAM1 de-encapsulates the -RSP(ACTPU) and passes it on to SSCP1.

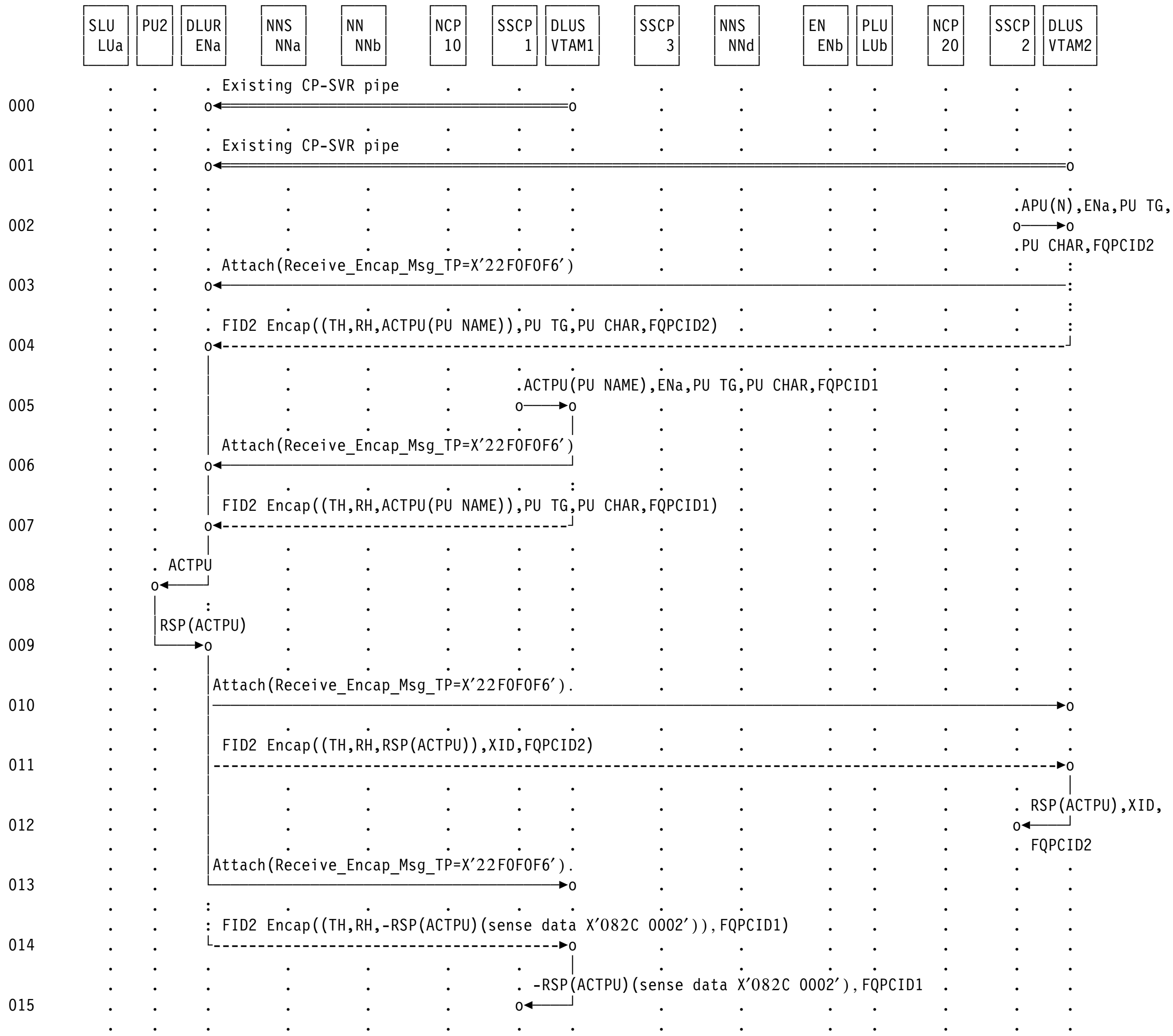


Figure 6-7. SSCP-PU session activation race - two DLUS-initiated requests

1. There are active CP-SVR pipes between VTAM1 and ENa and between VTAM2 and ENa.
2. SSCP2 receives some indication to activate PU2, so it builds an ACTPU (represented as APU) to be sent to PU2.
3. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
4. VTAM2 encapsulates the ACTPU and sends it to ENa.
5. Meanwhile SSCP1 also receives some indication to activate PU2, so it builds an ACTPU to be sent to PU2.
6. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
7. VTAM1 encapsulates the ACTPU and sends it to ENa.
8. ENa will process the ACTPU from SSCP2 (since it arrived first) reject the ACTPU from SSCP1. ENa de-encapsulates the ACTPU from SSCP2 and forwards it to PU2.
9. PU2 returns a positive RSP(ACTPU) to SSCP2 via ENa.
10. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
11. ENa encapsulates the RSP(ACTPU) and sends it to VTAM2.
12. VTAM2 de-encapsulates the RSP(ACTPU) and passes it on to SSCP2.
13. ENa recognizes that there already is an SSCP-PU session with PU2. It negatively responds to the ACTPU with sense data X′082C 0002′. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
14. ENa de-encapsulates the -RSP(ACTPU) and sends it to VTAM1.
15. VTAM1 de-encapsulates the -RSP(ACTPU) and passes it on to SSCP1.

6.2.3.3 TG and CP-SVR Pipe Race Conditions

Race conditions can exist involving either a DLUS and a DLUR that are adjacent or a DLUR, a non-adjacent DLUS, and an adjacent non-DLUS capable of sending ACTPU. Both race conditions have in common receiving two ACTPUs for a DLUR-supported PU, one over a TG (requested in an XID3 with the Suppress ACTPU Indicator off) and one encapsulated in a CP-SVR pipe. As in the previous section, the rule the DLUR applies to resolve these races is the same: **always accept the first ACTPU received**, with all subsequent ACTPUs received for the same PU rejected by the DLUR while the initial SSCP-PU session remains active. Here′s how this rule is applied in the different race conditions:

- ***race between ACTPUs over a TG and a CP-SVR pipe / DLUR and DLUS are adjacent*** - In this case the DLUS could send an ACTPU over the TG in response to the XID3 and later send another ACTPU for the same PU over the CP-SVR pipe to the DLUR. If this happens, the DLUR will accept the first ACTPU and reject the second. The same rule is followed if the first ACTPU arrived over the pipe and the second over the TG.
  - Since in this case the DLUS and the DLUR are adjacent, there ought to be a mechanism for the DLUR to give the DLUS enough information to prevent it from sending two ACTPUs. Such a mechanism will be provided as follows:
    - A new XID3 bit (byte 15, bit 1) has been defined to be the DLUR ACTPU Indicator. This bit will have significance when the ACTPU Suppression Indicator is reset (signifying an ACTPU is requested of the adjacent node). Setting the new bit will signify that the DLUR would prefer to receive the ACTPU over the CP-SVR pipe; resetting the new bit signifies no preference.
    - When a DLUS receives an XID3 with the ACTPU Suppression Indicator reset, it checks the DLUR ACTPU Indicator:
      - if the DLUR ACTPU Indicator is reset, the DLUS sends an ACTPU over the TG;
      - if the DLUR ACTPU Indicator is set, the DLUS searches its definitions for a DLUR-supported PU matching the PU for which the XID3 is being sent:
        - if a definition is found, the DLUS does not send an ACTPU over the TG - subsequent pipe activation mechanisms will cause an ACTPU to be sent over the CP-SVR pipe;
        - if no definition is found, the DLUS sends an ACTPU over the TG.

    Another way for the race condition to be avoided is for the DLUR to send XID3 with the ACTPU Suppression Indicator set. Some DLUR implementations will be able to decide when to set and when to reset the ACTPU Suppression Indicator and therefore will not need to set the DLUR ACTPU Indicator. Other DLUR implementations will not be able to be selective and will opt to always reset the ACTPU Suppression Indicator; these implementations should also always set the DLUR ACTPU Indicator to avoid the adjacent DLUS and DLUR ACTPU race condition.
- ***race between ACTPUs over a TG and a CP-SVR pipe / DLUR and DLUS are not adjacent*** - In this case a non-DLUS node could send an ACTPU over the TG in response to the XID3 and later a DLUS could send an ACTPU for the same PU over the CP-SVR pipe to the DLUR. If this happens, the DLUR will accept the first ACTPU and reject the second. The same rule is followed if the first ACTPU arrived over the pipe and the second over the TG.

6.2.4 PU/LU Network Name Forwarding

The DLUR BF will usually remove PU and LU names received in ACTPU and ACTLU RUs before forwarding the RUs on to the destination resource. However, a user can indicate to the DLUS that, for a particular PU and its associated LUs, the DLUR should not remove the PU and LU names from the ACTPU and ACTLU RUs when forwarding them.

6.2.4.1 Network Name Forwarding Capability Indicator

At CP-SVR pipe activation, both DLURs and DLUSs will indicate support for this network name forwarding function by setting on the network name forwarding supported indicator in the DLUR/S Capabilities (X′51′) control vector.

6.2.4.2 DLUS Processing

In this section a DLUS or DLUR which supports this function will be considered uplevel, while a DLUS or DLUR which does not support this function will be considered downlevel.

6.2.4.2.1 Uplevel DLUS Sending To Uplevel DLUR When a DLUS which supports the network name forwarding function is about to send an ACTPU for a PU2/2.1 to a DLUR which also supports this function,

- the DLUS will check the PU′s system definition to see whether the user requested the DLUR to strip off the Network Name (X′0E′) control vector or to forward it
- the DLUS will include a PU Capabilities (X′80′) ACTPU control vector along with the Network Name control vector in both format 0 and format 1 ACTPU RUs
- if the Network Name control vector is to be stripped off, the DLUS will reset the network name forwarding request indicator in the PU Capabilities control vector; if the Network Name control vector is to be forwarded, the DLUS will set the indicator

6.2.4.2.2 Uplevel DLUS Sending To Downlevel DLUR In this case the DLUS performs the same processing as a downlevel DLUS would, i.e.,

- for format 0 ACTPU, the DLUS will only include the Network Name control vector
- for format 1 ACTPU, the DLUS will include both the PU Capabilities and Network Name control vectors, and it will reset the network name forwarding request indicator in the PU Capabilities control vector

6.2.4.3 DLUR Processing

In this section a DLUS or DLUR which supports this function will be considered uplevel, while a DLUS or DLUR which does not support this function will be considered downlevel.

Whether removing or forwarding a PU or LU name, the DLUR will store the name for network management and other uses.

6.2.4.3.1 Uplevel DLUR Receiving From Uplevel DLUS When the DLUR receives the ACTPU,

- it will check the network name forwarding request indicator in the PU Capabilities control vector:
  - if the indicator is off, the DLUR will remove the Network Name and PU Capabilities control vectors from format 0 ACTPUs and remove the Network Name control vector from format 1 ACTPUs

- if the indicator is on, the DLUR will leave the control vectors on the ACTPU

- the DLUR will save the network name forwarding request indicator value so that it can be used when processing ACTLUs for the LUs associated with the PU:
  - if the indicator is off, the DLUR will remove the Network Name control vector from the ACTLU
  - if the indicator is on, the DLUR will leave the Network Name control vector on the ACTLU

6.2.4.3.2 Uplevel DLUR Receiving From Downlevel DLUS In this case the DLUR performs the same processing as a downlevel DLUR would, i.e., removing the Network Name control vector from ACTPU and ACTLU RUs.

6.3 SSCP-LU Session Activation

Once the ACTPU and RSP(ACTPU) have flowed, the DLUS node will activate the dependent LUs. This is done by sending an ACTLU over the SSCP-PU session from the DLUS to DLUR node. Since dependent LUs may be either predefined or dynamically created, there are two ways in which the ACTLU messages can be sent:

1. ***The DLUS node may activate predefined LUs.***

In this case, the DLUS node will use system-defined information to send ACTLU messages over the SSCP-PU session to the DLUR node.

2. ***The DLUR node may request dynamic activation of its dependent LUs.***

In this case, the DLUR node will use Dynamic Definition of Dependent LUs (DDDLU) support in VTAM to request activation of its dependent LUs (see 3.2.3, “Dynamic Definition of Dependent LUs Support” on page 3-5).

In either case, the DLUR node must also process the ACTLUs it receives to maintain a table associating dependent LU names with their supporting PU and PU connectivity information. This must be done because BINDs for dependent LUs must be routed by SLU name. The Route Selection Control Vector (RSCV) for the extended BIND will terminate in the DLUR node, and if the dependent LU is actually located on a downstream PU, the DLUR node will have to know how to send the BIND to the downstream PU/LU.

The information the DLUR needs to do this is the LU name, included by the DLUS in the ACTLU RU. The LU name will be defined in a Network Name (X′0E′) control vector type X′F3′. This name will be network-qualified if the PU′s NETID is different than the SSCP′s NETID. Unless it was indicated in the ACTPU to forward the LU name (for more details about LU name forwarding, see 6.2.4, “PU/LU Network Name Forwarding” on page 6-25), the DLUR will strip off this control vector before passing the ACTLU on to the PU; in either case, it will store the LU name for several uses, including PU/LU correlation, BIND routing, and network management.

In a subarea network, activation of an SSCP-LU session may require several RU flows, i.e., RNAA, SETCV, and ACTLU. For a DLUR-attached LU, these RUs will be combined into one, ACTLU. Therefore, ACTLU will now be encapsulated with an Assign LU Characteristics (X′30′) control vector, which the DLUR will process and strip from the RU before passing the ACTLU on to the LU. The control vector is identified in the flows as LU CHAR.

Figure 6-8 on page 6-28 illustrates an example of the activation of predefined dependent LUs.

6.3.1 Activation Of Predefined Dependent LUs

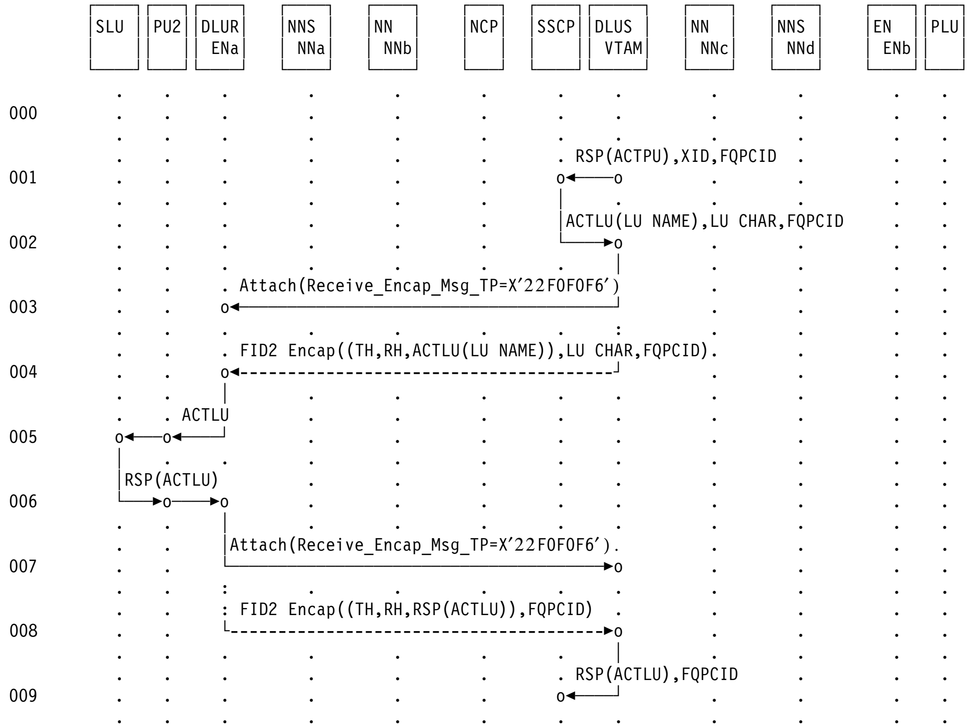


Figure 6-8. Activation of predefined LUs

1. When dependent LUs are predefined at the SSCP, the DLUS node will activate each of the LUs following activation of the SSCP-PU session.
2. The SSCP signals an ACTLU to the DLUS component. The DLUS node encapsulates the LU CHAR field with the ACTLU; this field carries LU information such as pacing parameters.
3. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
4. The DLUS component will encapsulate the ACTLU PIU with the FQPCID of the PU, and send it over the CP-SVR pipe to the DLUR node.
5. The DLUR node strips off the FID2 Encapsulation GDS variable (including the LU CHAR and FQPCID CVs), strips the Network Name (X′0E′) CV (which includes the LU name) from the ACTLU RU unless it was indicated in the ACTPU to forward the LU name (for more details about LU name forwarding, see 6.2.4, "PU/LU Network Name Forwarding" on page 6-25), and sends the ACTLU to the PU which forwards it to the LU.
6. The LU generates an RSP(ACTLU), sends it to the PU, which in turn sends the response to the DLUR component.
7. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
8. The DLUR component then uses the FQPCID of the PU and encapsulates the RSP(ACTLU) PIU. This is sent over the CP-SVR pipe to the DLUS node.
9. The DLUS node strips the FID2 Encapsulation GDS variable and sends the RSP(ACTLU) to the SSCP. Both nodes now use the FQPCID and FID2 TH addressing to reference the SSCP-LU session.

6.3.2 Dynamic Registration Of Dependent LUs

Since VTAM will provide DDDLU support, DLUR nodes will be able to dynamically register and obtain SSCP services for their dependent LUs. This is done by sending an NMVT with Product Set Identifier (PSID) for each dependent LU on the SSCP-PU session from the DLUR to DLUS node (see SNA Networking Update: Dynamic Definition of Dependent Logical Units (published by SNA Networking Vendor Enablement) for further details).

Note: In order to utilize this function, downstream or local PUs supported by DLUR nodes must provide the PSID and DDDLU support required.

Based upon information received in the PSID subvector, the SSCP will use model definitions to dynamically create the necessary resource definitions and control blocks. A subarea address will be assigned and the SSCP will respond with an ACTLU to the DLUR node. The DLUS node will use the FQPCID of the SSCP-PU session and the addressing in the TH of the FID2 to build an appropriate FID2 Encapsulation GDS variable and send the ACTLU to the DLUR node via the CP-SVR pipe. When the DLUR node receives the ACTLU, it will strip the FID2 Encapsulation GDS variable and deliver it to the appropriate LU. The LU will respond with a RSP(ACTLU) which the DLUR node will encapsulate and send to the DLUS node.

Figure 6-9 on page 6-31 illustrates dynamic registration and activation of dependent LUs.

6.3.3 Dynamic Registration And Activation Of Dependent LUs

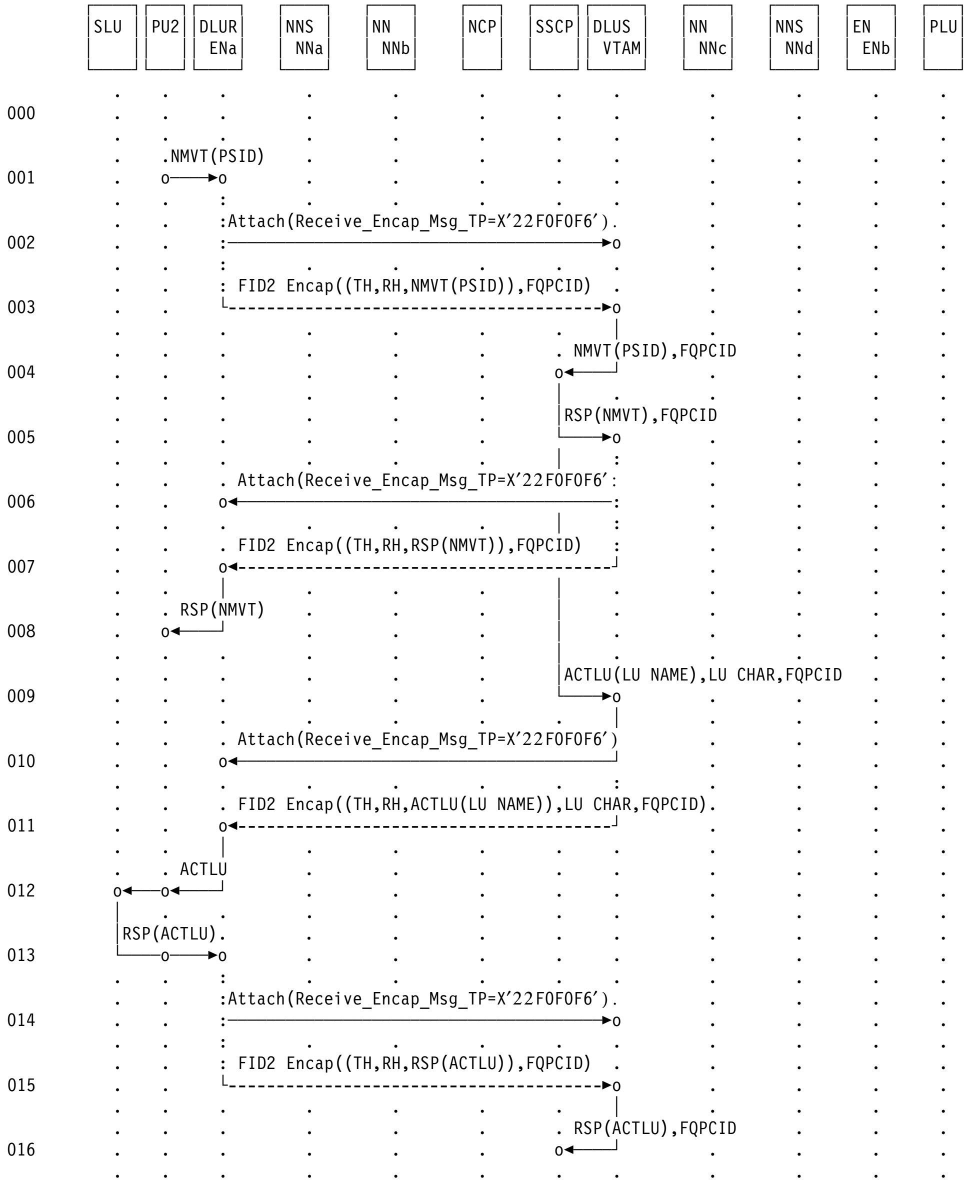


Figure 6-9. Dynamic dependent LU activation

1. PUs that have the necessary Dynamic Definition of Dependent LUs (DDDLU) support will generate an NMVT with the Product Set ID (PSID) information specific to a given LU and send it on the SSCP-PU session to the host. PUs with this support in a DLUR node will send this NMVT to the DLUR component.
2. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
3. The DLUR component will use the FQPCID of the SSCP-PU session to encapsulate the NMVT on the SSCP-PU session and send it up the CP-SVR pipe to the DLUS node.
4. The DLUS node will strip the FID2 Encapsulation GDS variable and send the NMVT to the SSCP.
5. The SSCP will generate a RSP(NMVT) and pass it to VTAM.
6. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
7. VTAM encapsulates the RSP(NMVT) and sends it to ENa.
8. ENa de-encapsulates the RSP(NMVT) and passes it on to PU2.
9. The SSCP will use the DDDLU support to generate an LU name, internal control blocks, and an ACTLU.
10. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
11. The DLUS component will then use FQPCID of the SSCP-PU session to identify the PU to the DLUR node. The LU address will be in the DAF field of the TH of the ACTLU PIU.

    Note: The NVMT carries a SNA Address List Subvector that identifies the local address of the LU to be activated. This, in turn, is included in the DAF field of the ACTLU PIU sent to the PU.
12. The DLUR node removes the FID2 Encapsulation GDS variable (including the LU CHAR and FQPCID CVs) and strips the Network Name (X′0E′) CV (which includes the LU name) from the ACTLU RU unless it was indicated in the ACTPU to forward the LU name (for more details about LU name forwarding, see 6.2.4, “PU/LU Network Name Forwarding” on page 6-25). The DLUR uses the FQPCID to pass the ACTLU command to the appropriate PU. The DLUR node uses the LU name to build a table associating LUs with PUs. This enables the DLUR to route subsequent BINDs (by SLU name0 to the proper PU. The ACTLU is forwarded from PU to LU.
13. The RSP(ACTLU) is generated by the LU, passed to the PU, and forwarded to the DLUR component.
14. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
15. The DLUR component uses the FQPCID of the PU and encapsulates the RSP(ACTLU) PIU on the CP-SVR pipe to the DLUS node.
16. The DLUS node strips the FID2 Encapsulation GDS variable and passes the RSP(ACTLU) to the SSCP. Both the DLUS and DLUR nodes will now use the FQPCID of SSCP-PU session and the FID2 TH addressing of the PIUs to reference the SSCP-LU session for this LU.

6.4 SSCP-PU/SSCP-LU Session Deactivation (DLUS-Initiated)

(For DLUR-initiated SSCP-PU and SSCP-LU session deactivation, see 5.6.3, “Abnormal SSCP-PU/SSCP-LU Session Deactivation (DLUR-Initiated)” on page 5-47)

The DLUS has several mechanisms to deactivate existing SSCP-PU and SSCP-LU sessions with DLUR-attached resources:

- ***normal*** - SSCP-LU sessions are deactivated first followed by the corresponding SSCP-PU session; any active LU-LU sessions are deactivated during the corresponding SSCP-LU session deactivation.

  Usually LU-LU sessions are deactivated first before deactivating the SSCP-LU session. Optionally the deactivation of the last LU-LU session can trigger the deactivation of the SSCP-LU session, and if it is the last active SSCP-LU session, this can trigger the deactivation of the SSCP-PU session, and if it is the last active SSCP-PU session, this can trigger the deactivation of the CP-SVR pipe. Another option is to keep the pipe active even when the last SSCP-PU session using it has been deactivated.

- ***forced*** - deactivation of the SSCP-PU session forces deactivation of any active SSCP-LU sessions associated with that PU; any active LU-LU sessions are deactivated during the SSCP-PU session deactivation

  The DLUS in this case will build and forward to its SSCP a RSP(DACTPU) right after sending the DACTPU request to the DLUR, not waiting for the PU-generated RSP(DACTPU). This means:

  – if the DLUS receives the RSP(DACTPU) from the DLUR, it should discard the response (see Figure 6-11 on page 6-38) - this could occur when other active SSCP-PU session(s) will keep the DLUS from deactivating the pipe - and if the DLUS receives a SESSEND from the DLUR, it should discard the request

  – if the DLUR receives an UNBIND for the CP-SVR pipe before it can send the RSP(DACTPU) from the PU back on the pipe, it should discard the response (see Figure 5-15 on page 5-54) - this could occur when the DLUS decides to deactivate the pipe before the DLUR has completed handling the DACTPU request and response

- ***giveback*** - deactivation of the SSCP-PU session forces deactivation of any active SSCP-LU sessions associated with that PU; any active LU-LU sessions are left active during the SSCP-PU session deactivation if the PU is defined with ANS=CONT - if the PU is defined with ANS=STOP, the active LU-LU sessions are also deactivated (the ANS value is carried in the PU CHAR (CV X′43′) field on ACTPU).

6.4.1 DLUR ANS Support

It is strongly recommended that DLUR implementations support both ANS=CONT and ANS=STOP. However, for those DLURs that can only support ANS=STOP, the following architecture is included so that the DLUS will know the DLUR′s ANS support level:

- A support indicator in the DLUR/S Capabilities (X′51′) control vector has been defined to indicate the level of ANS support provided by a DLUR:

  – 0 = full ANS support (both ANS=CONT and ANS=STOP supported)

  – 1 = limited ANS support (only ANS=STOP supported)

- When building a CV X′51′, the DLUR will set this indicator based on its level of ANS support.

6.4.2 SSCP-PU/SSCP-LU Session Deactivation Flows

Figure 6-10 on page 6-35 through Figure 6-13 on page 6-42 illustrate examples of the types of deactivation described in 6.4, "SSCP-PU/SSCP-LU Session Deactivation (DLUS-Initiated)":

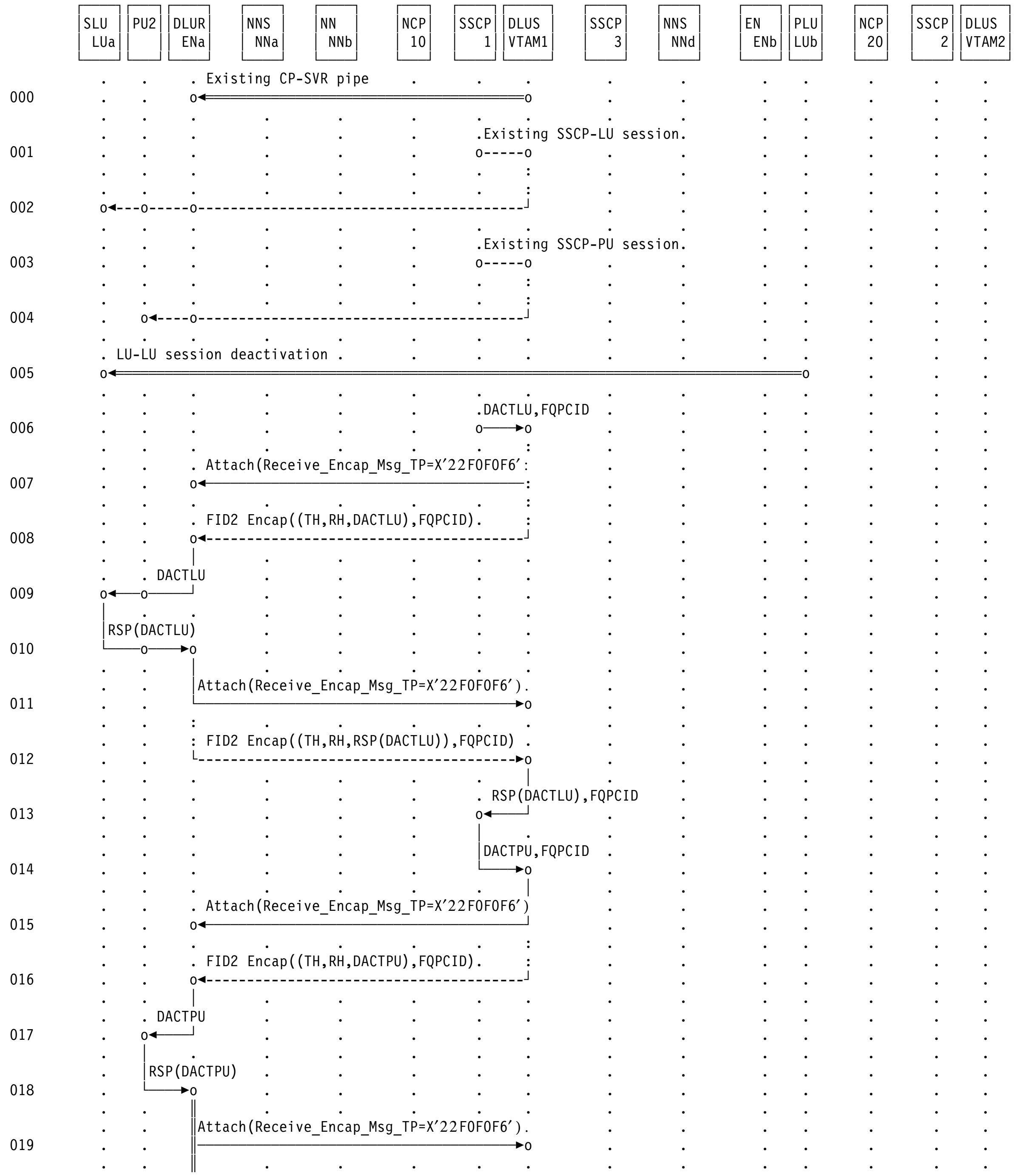


Figure 6-10. Normal SSCP-PU/SSCP-LU session deactivation

```
      ┌────┐ ┌───┐ ┌────┐   ┌────┐   ┌────┐    ┌───┐  ┌────┐┌─────┐   ┌────┐   ┌────┐    ┌────┐┌───┐  ┌───┐  ┌────┐┌─────┐
      │SLU │ │PU2│ │DLUR│   │NNS │   │NN  │    │NCP│  │SSCP││DLUS │   │SSCP│   │NNS │    │EN  ││PLU│  │NCP│  │SSCP││DLUS │
      │ LUa│ │   │ │ ENa│   │ NNa│   │ NNb│    │ 10│  │   1││VTAM1│   │   3│   │ NNd│    │ ENb││LUb│  │ 20│  │   2││VTAM2│
      └────┘ └───┘ └────┘   └────┘   └────┘    └───┘  └────┘└─────┘   └────┘   └────┘    └────┘└───┘  └───┘  └────┘└─────┘
         .     .     ║ FID2 Encap((TH,RH,RSP(DACTPU)),FQPCID) .         .        .          .    .      .      .     .
020      .     .     ║---------------------------------------►o         .        .          .    .      .      .     .
         .     .     ║        .        .        .       .     │         .        .          .    .      .      .     .
         .     .     ║        .        .        .       .  RSP (DACTPU),FQPCID   .          .    .      .      .     .
021      .     .     ║        .        .        .       o◄────┘         .        .          .    .      .      .     .
         .     .     ║        .        .        .       .     .         .        .          .    .      .      .     .
         .     . DLC deactivation      .        .       .     .         .        .          .    .      .      .     .
022      .     o◄════╝        .        .        .       .     .         .        .          .    .      .      .     .
         .     .     .        .        .        .       .     .         .        .          .    .      .      .     .
```

1. There is an active session between SSCP1 and LUa. It is the only active SSCP-LU session associated with PU2. There is also an active session between LUb and LUa.
2.
3. There is an active session between SSCP1 and PU2, but it is not the only active SSCP-PU session using the CP-SVR pipe between ENa and VTAM1. PU2 is the only active PU on the connection between PU2 and ENa.
4.
5. LUb terminates its session with LUa, triggering SSCP1 to begin normal deactivation of its session with LUa.
6. SSCP1 initiates normal deactivation of its session with LUa by building a DACTLU and passing it on to VTAM1.
7. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
8. The DLUS VTAM1 encapsulates the DACTLU and forwards it to the DLUR ENa.
9. The DLUR ENa de-encapsulates the DACTLU and forwards it to LUa.
10. LUa returns a positive RSP(DACTLU) to SSCP1 via ENa.
11. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
12. ENa encapsulates the RSP(DACTLU) and sends it to VTAM1.
13. VTAM1 de-encapsulates the RSP(DACTLU) and forwards it to SSCP1.
14. SSCP1 initiates normal deactivation of its session with PU2 by building a DACTPU and passing it on to VTAM1.
15. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
16. The DLUS VTAM1 encapsulates the DACTPU and forwards it to the DLUR ENa.
17. The DLUR ENa de-encapsulates the DACTPU and forwards it to PU2.
18. PU2 returns a positive RSP(DACTPU) to SSCP1 via ENa.
19. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
20. ENa encapsulates the RSP(DACTPU) and sends it to VTAM1.
21. VTAM1 de-encapsulates the RSP(DACTPU) and forwards it to SSCP1. VTAM1 also determines that there are other active or pending SSCP-PU sessions on its CP-SVR pipe to ENa, so it does not take down the pipe.
22. ENa initiates DLC deactivation if PU2 is a downstream PU.

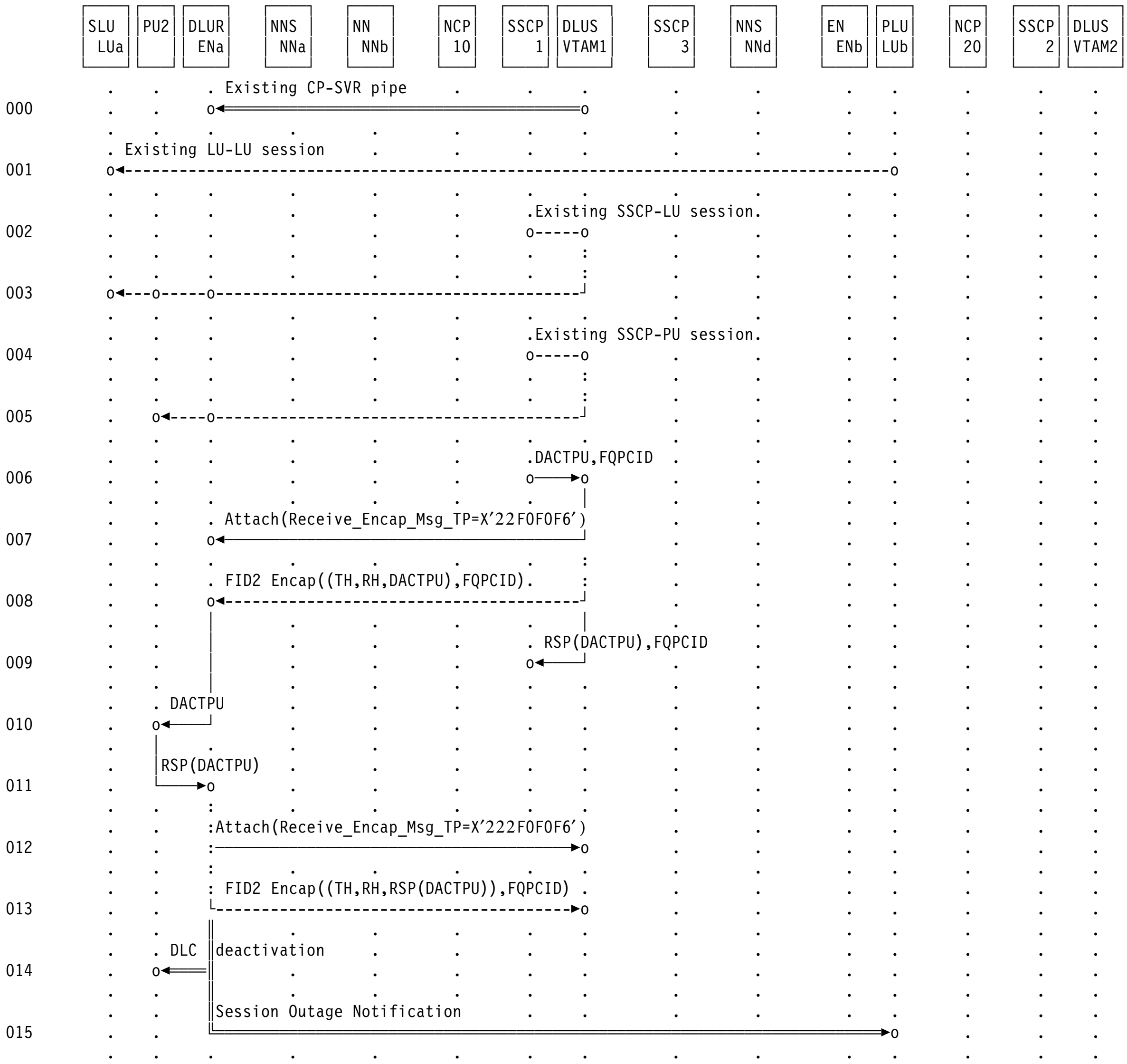


Figure 6-11. Forced SSCP-PU/SSCP-LU session deactivation - CP-SVR pipe remains active

1. There is an active session between LUb and LUa.
2. There is an active session between SSCP1 and LUa. It is the only active SSCP-LU session associated with PU2.
3.
4. There is an active session between SSCP1 and PU2, but it is not the only active SSCP-PU session using the CP-SVR pipe between ENa and VTAM1. PU2 is the only active PU on the connection between PU2 and ENa.
5.
6. SSCP1 initiates forced deactivation of its session with PU2 by building a DACTPU and passing it on to VTAM1. SSCP1 also resets its SSCP-LU session with LUa.
7. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
8. The DLUS VTAM1 encapsulates the DACTPU and forwards it to the DLUR ENa.
9. VTAM1 builds a RSP(DACTPU) and forwards it to SSCP1, not waiting for the response to return from the PU. VTAM1 also determines that there are other active or pending SSCP-PU sessions on its CP-SVR pipe to ENa, so it does not take down the pipe.
10. The DLUR ENa de-encapsulates the DACTPU and forwards it to PU2.
11. PU2 returns a positive RSP(DACTPU) to SSCP1 via ENa. PU2 also resets the SSCP-LU session between SSCP1 and LUa.
12. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
13. ENa encapsulates the RSP(DACTPU) and sends it to VTAM1. VTAM1 de-encapsulates the RSP(DACTPU) and discards it, since VTAM1 has already sent a RSP(DACTPU) to SSCP1.
14. ENa initiates DLC deactivation if PU2 is a downstream PU.
15. ENa cleans up its status about the SSCP-LU session between SSCP1 and LUa and the LU-LU session between LUb and LUa. ENa also initiates Session Outage Notification to LUb, so that it too will reset the LU-LU session. During this process, if ENa builds and sends a SESSEND to VTAM1, the DLUS will discard it.

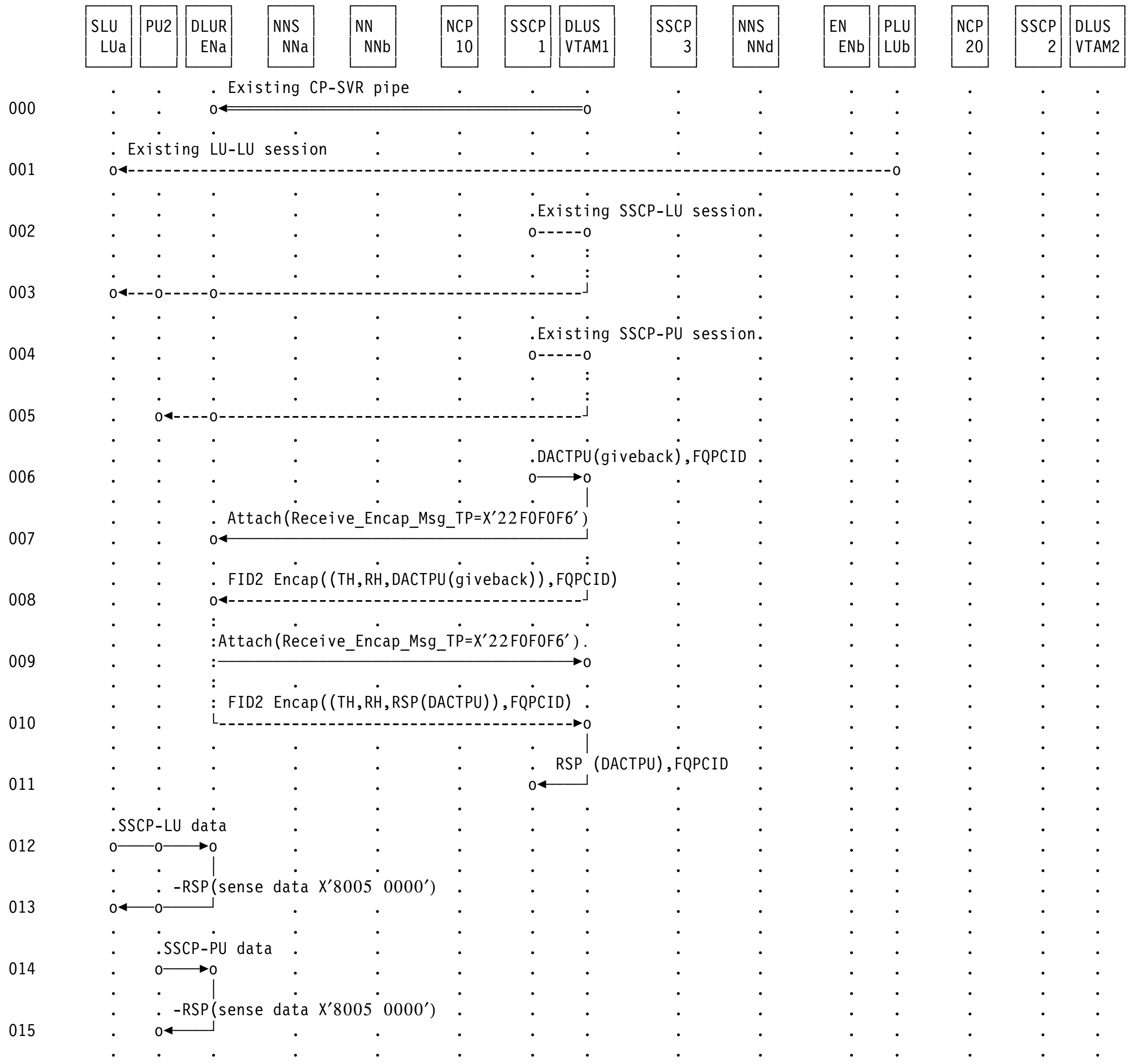


Figure 6-12. Giveback SSCP-PU/SSCP-LU session deactivation - ANS=CONT

1. There is an active session between LUb and LUa.
2. There is an active session between SSCP1 and LUa. It is the only active SSCP-LU session associated with PU2.
3.
4. There is an active session between SSCP1 and PU2, but it is not the only active SSCP-PU session using the CP-SVR pipe between ENa and VTAM1. PU2 is the only active PU on the connection between PU2 and ENa, and PU2 has been defined with ANS=CONT.
5.
6. SSCP1 initiates giveback deactivation of its session with PU2 by building a DACTPU(giveback) and passing it on to VTAM1. SSCP1 also resets its SSCP-LU session with LUa.
7. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
8. The DLUS VTAM1 encapsulates the DACTPU(giveback) and forwards it to the DLUR ENa.
9. The DLUR ENa de-encapsulates the DACTPU(giveback); since PU2 is defined as ANS=CONT, the LU-LU sessions will remain active while the SSCP-PU and SSCP-LU sessions are reset. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
10. ENa converts the DACTPU(giveback) into a RSP(DACTPU), encapsulates it, and sends it to VTAM1. ENa must remember that the SSCP-PU and SSCP-LU sessions are reset in SSCP1. ENa also cleans up its status about the SSCP-LU session between SSCP1 and LUa. ENa leaves the LU-LU session and the associated DLC active.
11. VTAM1 de-encapsulates the RSP(DACTPU) and forwards it to SSCP1. VTAM1 also determines that there are other active or pending SSCP-PU sessions on its CP-SVR pipe to ENa, so it does not take down the pipe.
12. Later, LUa sends a status message to SSCP1, believing the SSCP-LU session is active.
13. ENa, knowing that the SSCP-LU session actually is reset, converts the command into a -RSP with sense data X′8005 0000′ (no session) and returns it to LUa.
14. PU2 sends a status message to SSCP1, believing the SSCP-PU session is active.
15. ENa, knowing that the SSCP-PU session actually is reset, converts the command into a -RSP with sense data X′8005 0000′ (no session) and returns it to PU2.

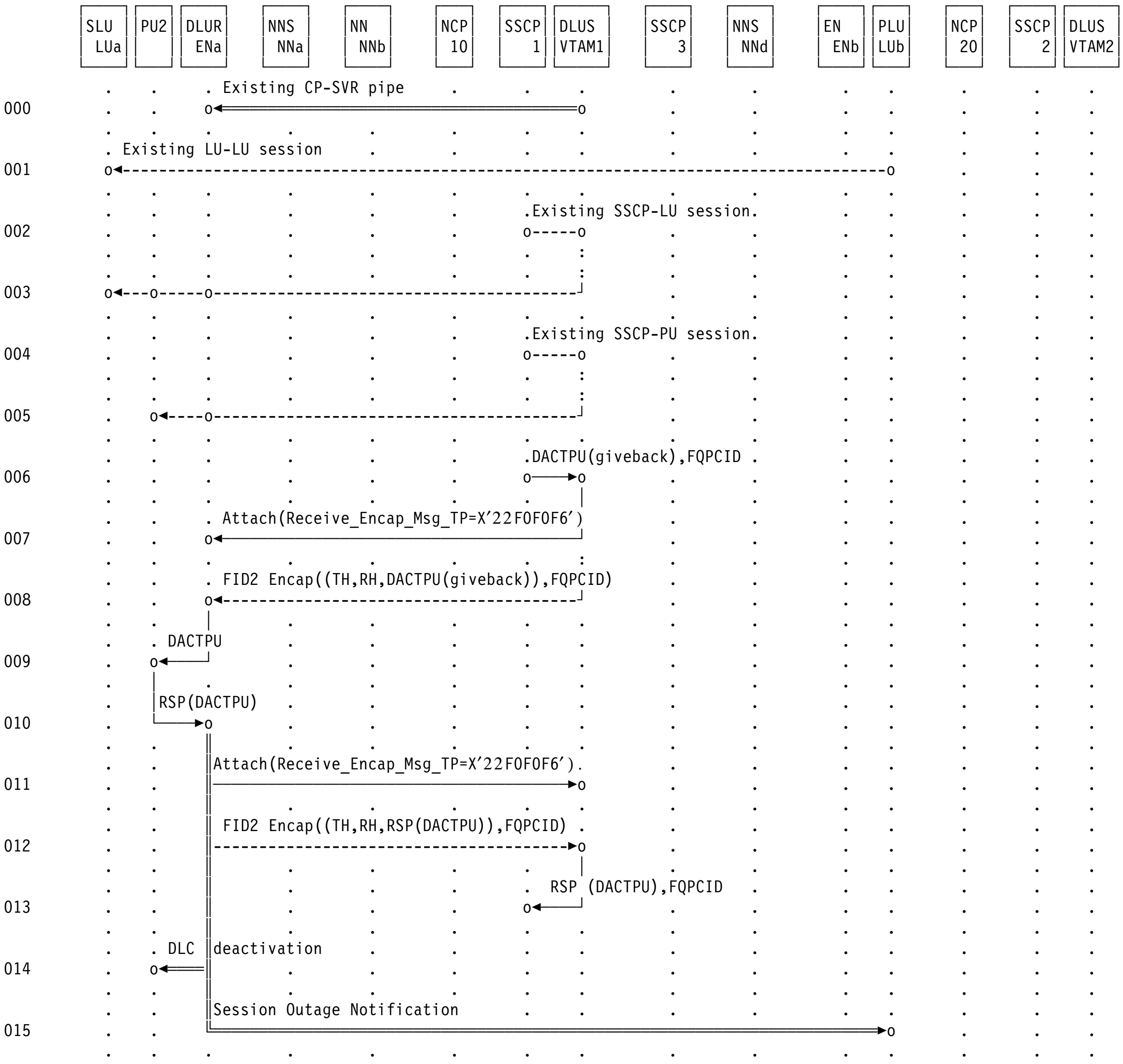


Figure 6-13. Giveback SSCP-PU/SSCP-LU session deactivation - ANS=STOP

1. There is an active session between LUb and LUa.
2. There is an active session between SSCP1 and LUa. It is the only active SSCP-LU session associated with PU2.
3.
4. There is an active session between SSCP1 and PU2, but it is not the only active SSCP-PU session using the CP-SVR pipe between ENa and VTAM1. PU2 is the only active PU on the connection between PU2 and ENa, and PU2 has been defined with ANS=STOP.
5.
6. SSCP1 initiates giveback deactivation of its session with PU2 by building a DACTPU(giveback) and passing it on to VTAM1. SSCP1 also resets its SSCP-LU session with LUa.
7. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
8. The DLUS VTAM1 encapsulates the DACTPU(giveback) and forwards it to the DLUR ENa.
9. The DLUR ENa de-encapsulates the DACTPU(giveback); since PU2 is defined as ANS=STOP, the LU-LU sessions will be deactivated along with the SSCP-PU and SSCP-LU sessions. This can be accomplished by normal deactivation procedures. So ENa converts the DACTPU(giveback) into a normal DACTPU and forwards it to PU2.
10. PU2 returns a positive RSP(DACTPU) to SSCP1 via ENa. PU2 also resets the SSCP-LU session between SSCP1 and LUa.
11. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
12. ENa encapsulates the RSP(DACTPU) and sends it to VTAM1.
13. VTAM1 de-encapsulates the RSP(DACTPU) and forwards it to SSCP1. VTAM1 also determines that there are other active or pending SSCP-PU sessions on its CP-SVR pipe to ENa, so it does not take down the pipe.
14. ENa initiates DLC deactivation if PU2 is a downstream PU.
15. ENa cleans up its status about the SSCP-LU session between SSCP1 and LUa and the LU-LU session between LUb and LUa. ENa also initiates Session Outage Notification to LUb, so that it too will reset the LU-LU session.

6.4.3 PU Activation After Deactivation

If a PU was initially activated with ANS=STOP, the deactivation process will deactivate any active LU-LU sessions associated with the PU. When the PU is reactivated, whether with ANS=STOP or ANS=CONT, there are no associated sessions to handle.

If, however, the PU is initially activated with ANS=CONT, the deactivation process will not tear down the active LU-LU sessions. If the PU is reactivated with ANS=CONT, the LU-LU sessions stay up and will be identified to the new SSCP in a CV X′2A′ included in RSP(ACTLU) by the DLUR (see 10.1.2, “DLUR/S Dependent LU Session Awareness” on page 10-2 for more information). If the PU is reactivated with ANS=STOP, the DLUR will deactivate any active LU-LU sessions as part of the PU activation process.

To summarize:

Table 6-2. PU Reactivation Effects On Active LU-LU Sessions		
PU Activation/ANS Type	**Second/STOP**	**Second/CONT**
First/STOP	no LU-LU sessions to handle	no LU-LU sessions to handle
First/CONT	deactivate active LU-LU sessions	leave active LU-LU sessions alone

6.5 Additional Error Cases

6.5.1 Received GDS Variable X′1500′ Without CV X′60′

Any PIU request encapsulated in a GDS variable X′1500′ which does not include a CV X′60′ will be discarded.

6.5.2 Received GDS Variable X′1500′ (ACTPU Encapsulated) With Missing Or Incomplete CV X′46′

Any ACTPU request encapsulated in a GDS variable X′1500′ which does not include a complete CV X′46′ (e.g., CV missing its signalling information or CV missing completely) will be rejected with sense data X′0806 0000′.

6.5.3 Received GDS Variable X′1500′ (ACTPU Encapsulated) Without CV X′51′

Any ACTPU request encapsulated in a GDS variable X′1500′ which does not include a CV X′51′ when one is expected (first ACTPU after the CP-SVR pipe is activated) will be rejected with sense data X′086C 5100′.

6.5.4 Received GDS Variable X′1500′ (RSP(REQACTPU) Encapsulated) Without CV X′51′

Any REQACTPU response encapsulated in a GDS variable X′1500′ which does not include a CV X′51′ when one is expected (first REQACTPU after the CP-SVR pipe is activated) will be rejected by rejecting the ensuing corresponding ACTPU with sense data X′086C 5100′.

Chapter 7. Unformatted Session Services Management

Once the SSCP-LU session has been activated, the SSCP will respond by sending the appropriate USSMSG10 (“Good Morning” message) to the dependent LU via the CP-SVR pipe. If the RSP(ACTLU) indicates that the LU is enabled, then the SSCP will respond immediately with the USSMSG10. If the RSP(ACTLU) indicates that the LU is not enabled, the SSCP (DLUS node) will wait to receive a NOTIFY(CV X′0C′) indicating that the LU has been enabled before sending the USSMSG10.

Figure 7-1 illustrates USSMSG10 processing:

Note: Refer to Table 3-1 on page 3-2 for an explanation of the notations used in the diagram in this chapter.

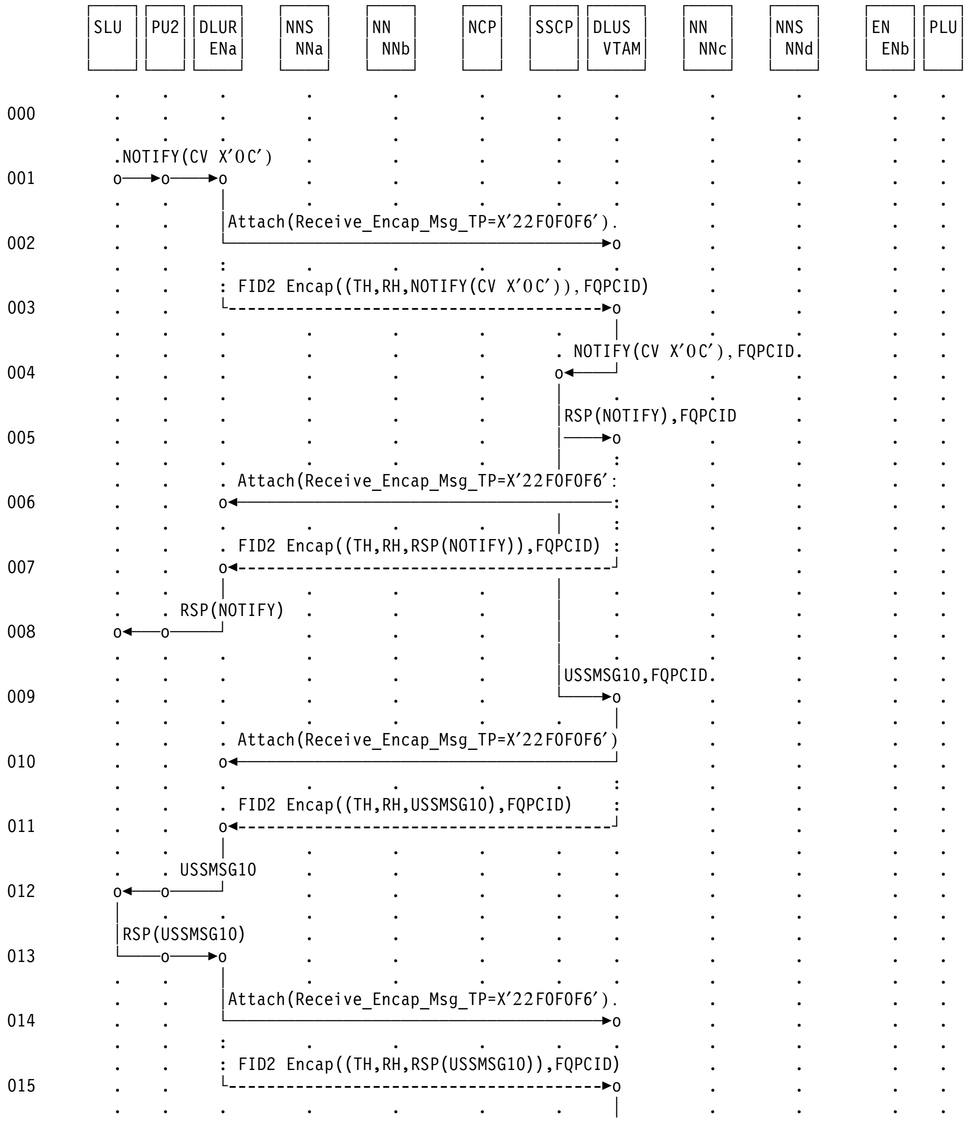


Figure 7-1 (Part 1 of 2). Unformatted Session Services Management


```
         .   .    .       .       .       .      . RSP(USSMSG10),FQPCID  .        .   .
016      .   .    .       .       .       .      o<─────┘       .        .        .   .
         .   .    .       .       .       .      .    .         .        .        .   .
```

Figure 7-1 (Part 2 of 2). Unformatted Session Services Management

1. In the case where the RSP(ACTLU) indicates that the LU is not currently enabled, USSMSG10 is not sent to the LU until it is enabled. The LU indicates that it is enabled by sending a NOTIFY(CV X′0C′).
2. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
3. The DLUR component uses the FQPCID of the PU to encapsulate the NOTIFY(CV X′0C′) PIU and sends it up the CP-SVR pipe to the DLUS node.
4. The DLUS node removes the FID2 Encapsulation GDS variable and sends the NOTIFY to the SSCP.
5. The SSCP sends a RSP(NOTIFY).
6. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
7. The DLUS node uses the FQPCID in the FID2 Encapsulation GDS variable and sends the RSP(NOTIFY) to the DLUR node.
8. The DLUR node removes the FID2 Encapsulation GDS variable and forwards the RSP(NOTIFY) to the PU and LU.
9. After detecting the LU is enabled (either via ACTLU enabled or NOTIFY(CV X′0C′)), the SSCP sends a USSMSG10 to the LU. This is done by sending the USSMSG10 format from the SSCP to the DLUS component of the DLUS node.
10. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
11. The DLUS component of the node uses the FQPCID to encapsulate the USSMSG10 and send it on the CP-SVR pipe to the DLUR node.
12. The DLUR node removes the FID2 Encapsulation GDS variable and forwards the USSMSG10 message to the PU and LU.
13. The LU generates an acknowledgment to the USSMSG10 and sends the response to the PU and DLUR component.
14. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
15. The DLUR component will use the FQPCID to encapsulate the positive response and send it over the CP-SVR pipe to the DLUS node.
16. The DLUS node receives the response, strips the FID2 Encapsulation GDS variable, and passes the response to the SSCP.

In addition to the USSMSG10 flows described above, the SSCP component of the DLUS node will also be required to support other session services message flows for both formatted and unformatted session initiation requests. The DLUS node will also be responsible for session limit management of its dependent (DLUR-attached) LUs. In order for the SSCP component to provide these services, SLU and PLU nodes must send Session Start (SESSST) and Session End (SESSEND) messages to the SSCP. This will be done over the CP-SVR pipe, Session Services Extensions Locate flows, or SSCP-PU4 flows depending upon the location of the PLU node. These message flows are indicated throughout the following chapters that describe various session initiation/termination flows. They are also described in Chapter 10, “Dependent LU Session Awareness Management” on page 10-1.

Chapter 8. LU-LU Session Routing

SSCP services (or Session Services) will be provided to the dependent SLUs by the DLUS node, but session routing will be directly between the PLU and SLU nodes. The DLUS node is the owner of the dependent SLUs. As such, it is the CP responsible for responding to searches for the SLUs it serves. In this sense, it acts as the CP(SLU). Since the SLU is actually located on or directly attached to the DLUR node, the DLUS will appear in the associated resource hierarchy of Locate Find and Locate Found flows as if it were the NNS(SLU) and the DLUR node as the CP(SLU). In this way, other nodes will cache the DLUS node as the NNS of the SLU and will, therefore, send any ensuing session initiation requests to the DLUR node via the DLUS node. In order to get session routing directly between the PLU and SLU nodes, the associated resource hierarchy will also carry the tail vectors (TGVs) associated with the DLUR node. In the case where the DLUR is a T2.1 EN, these tail vectors are the real DLUR node TGVs. In the case where the DLUR node is a T2.1 NN, the Cdinit does not carry any tail vectors. [1]

In the first case (when the DLUR node is an EN), Topology and Routing Services (TRS) at the NNS(PLU) will compute a route from the origin node to one of the partner NNs specified in the DLUR TGVs. The final hop is constructed using the TG number specified in the associated TGV and the CP name of the DLUR node.

In the second case (when the DLUR node is an NN), SS at the NNS(PLU) should pass a REQUEST_ROUTE message to TRS indicating the DLUR (NN) as the target node with no TGVs. Since the DLUR is actually a NN (and therefore part of the topology database), TRS should respond with the appropriate route.

8.1 BIND Routing

The DLUR must examine RUs on the SSCP-PU and SSCP-LU sessions and save information from these RUs for BIND processing.

8.1.1 ACTLU

Since the RSCV for incoming BINDs will terminate at the DLUR node, the DLUR node must maintain a knowledge of dependent LU locations. This is so that the DLUR node can route incoming BIND messages by SLU name to the dependent SLU. In order to do this, DLUR nodes will maintain a relationship between dependent LU names, their supporting PU, and the physical connectivity to the PU. This is done by examining the ACTLU commands as they are received at the DLUR node (as mentioned in 6.3, “SSCP-LU Session Activation” on page 6-27).

Part of the information maintained by the DLUR will be a session count field, initialized to 0. When the DLUR receives a PLU-initiated BIND, it does not know whether the DLUS has been involved in the LU-LU session activation process or rather that the RU is a surprise BIND. To keep from passing a BIND to an LU already in an LU-LU session, the DLUR will manage the session count field and only

1 This assumes that the NNS of APPN PLU nodes will support receipt of a Locate_Found indicating an NNCP, ENCP hierarchy with a Cdinit that contains no TGVs. Product polls to date indicate this is not a problem. See A.1, “LU-LU Session Route Calculation” on page A-1 for a description of an alternate approach to provide direct routing between the CP(PLU) and DLUR node, if TGVs are required.

pass a BIND to an LU whose current session count is zero (maximum session count value for a dependent LU is one).

The DLUR will zero out all indicators (ACTLU RU byte 1) except the type activation requested field (bits 6-7). Other indicators have been defined (and appear in the licensed formats) for use by the NCP BF and are not recognized by the LU; the NCP BF zeroes out these indicators and so will the DLUR.

The DLUR will remember the function management (FM) and transmission services (TS) profiles (which together make up ACTLU RU byte 2). This information will be used when converting a 2 byte RSP(ACTLU) into a 3 byte RSP(ACTLU).

The DLUR will remember two fields from the Assign LU Characteristics (X′30′) control vector included in the X′1500′ GDS variable encapsulating the ACTLU. The following fields will be used in processing a BIND for adaptive session pacing and for window size setting:

- Route extension (REX) stage pacing parameter (byte 4, bit 0)
- REX stage pacing window size (byte 4, bits 2-7)

8.1.2 +RSP(ACTLU)

The DLUR must also examine the +RSP(ACTLU), since it carries the X′0C′ (LU-LU Session Services Capabilities) control vector.

- If the RSP(ACTLU) does not have a CV X′0C′, the DLUR will build one:
  - if the RSP(ACTLU) includes a CV X′0C′, continue processing
  - if the RSP(ACTLU) only includes a CV X′00′, build a CV X′0C′ and add it to the RU
  - if the RSP(ACTLU) has no CVs, build a CV X′00′ and a CV X′0C′ and add them to the RU
    - if the RSP(ACTLU) RU is 2 bytes long, add a third byte before adding the CVs; the contents of the added byte will be the remembered ACTLU RU byte 2
  - when building a CV X′00′, set maximum RU size to 256 bytes and all other fields to 0
  - when building a CV X′0C′, set
    - secondary LU capability to enabled
    - LU-LU session limit to 1
    - LU-LU session count to 1 if there is an active LU-LU session or 0 otherwise (part of session count processing described below)
    - subarea node extended BIND support to 1 (part of extended BIND support processing described below)
    - SESSST support to 1 (part of new DLUR support described below)
    - all other fields to 0
- In the CV X′0C′ are LU-LU session limit and count fields (bytes 3-4 and 5-6, respectively). The DLUR will set:
  - LU-LU session limit to 1 if the LU is not PLU-capable
  - LU-LU session count to 1 if there is an active LU-LU session or 0 otherwise
- In the CV X′0C′ is an indicator as to whether or not the dependent LU supports SESSSTs (byte 7, bit 2). The DLUR will set this indicator on, regardless of the value set by the dependent LU for this

indicator. This support includes all aspects of session awareness, i.e., SESSEND and CV X′2A′ as well as SESSST (see 10.1.2, “DLUR/S Dependent LU Session Awareness” on page 10-2 for more information on session awareness).

- In the CV X′0C′ are two indicators as to whether or not the dependent/subarea LU supports extended BINDs (byte 7, bits 4 and 6). The DLUR will remember these indicators′ settings with its other information about the dependent LU. Both bits off will indicate that the LU does not support extended BINDs, while either bit on will indicate the LU does support extended BINDs. The DLUR will also set the peripheral node extended BIND support to 0 and the subarea node extended BIND support to 1.
- In the CV X′0C′ is an indicator as to whether or not the dependent LU supports network-qualified names (byte 7, bit 5). The DLUR will remember this indicator′s setting with its other information about the dependent LU.

If there already is an active LU-LU session when the +RSP(ACTLU) is received, the active LU-LU session awareness information will be sent to the SSCP in the DLUS by way of the CV X′2A′ added to the +RSP(ACTLU) by the DLUR (see 10.1.2, “DLUR/S Dependent LU Session Awareness” on page 10-2 for more information on session awareness).

8.1.3 INIT-SELF

When the DLUR node receives an INIT-SELF from a DLUR-attached dependent LU,

- if the INIT-SELF format is Format 0, the DLUR will create a User Request Correlation (URC) (X′0A′) control vector, include the control vector in the RU, and remember that the URC is DLUR-generated;
- if the INIT-SELF format is Format 1,
  - if the URC field is not present, the DLUR will create a URC, include it in the RU, and remember that the URC is DLUR-generated.

For all INIT-SELF RUs received from a DLUR-attached dependent LU the DLUR will save the PLU name and the User Request Correlation (URC) field from the RU and associate this information with the LU.

8.1.4 BIND

When the DLUR node receives a BIND whose RSCV terminates at the DLUR, the DLUR must check its LU table to determine the location of the SLU name specified in the RU. Once located,

- ***SLU session count processing -*** the LU′s session count will be checked:
  - if the count is zero, it will be incremented to one, and processing will continue;
  - if the count is already one, the BIND will be rejected with sense data X′0805 000A′;
- ***extended BIND processing -*** the BIND will be unextended if necessary, based on the extended BIND support indicators in the CV X′0C′ in the +RSP(ACTLU) - this involves removing all keyed control vectors, turning off the CVs included indicator, truncating trailing zeroes and the SLU name (for non-negotiable BINDs), and updating the new length of the BIND RU;
- ***network-qualified name processing -*** the NETIDs (if any) will be stripped from the LU names in the BIND if necessary, based on the network-qualified names support indicator in the CV X′0C′ in the +RSP(ACTLU);
- ***name substitution processing -*** the PLU name will be replaced if necessary, based on whether or not the saved and BIND′s URCs match:

- if the URCs do not match, BIND processing will continue;
- if the URCs match,
  - the BIND PLU name will be replaced with the saved PLU name,
  - the DLUR will discard the saved URC and the saved PLU name, and
  - if the URC was DLUR-generated, the URC will be removed from the BIND;

- ***adaptive session pacing processing -*** the Adaptive Session-Level Pacing support indicator (byte 9, bit 0 in the BIND) value for the APPN stage (from the DLUR into the APPN network) will be remembered by the DLUR, which will then set the indicator for the REX stage (from the DLUR to the downstream PU) as follows - note that this processing is performed on non-negotiable BINDs as well as negotiable BINDs:
  - if the BIND is unextended set the ASP indicator off;
  - if the BIND is extended set the ASP indicator to the same value as the REX stage pacing parameter (byte 4, bit 0) in the Assign LU Characteristics (X′30′) control vector included in the X′1500′ GDS variable encapsulating the ACTLU (note that the supported/not supported settings are defined opposite to each other in the BIND and in the CV X′30′);
- ***pacing window size processing -*** the DLUR will put the REX stage pacing window size into the BIND only if the following three conditions are all true:
  1. the DLUR changed the BIND′s ASP indicator from 1 (adaptive) to 0 (fixed), and
  2. the BIND is negotiable, and
  3. the BIND pacing window size is zero;

and the BIND (if not rejected) will be forwarded to the appropriate internal or external component.

When a DLUR receives a BIND for a dependent LU and there is no active CP-SVR pipe servicing that dependent LU, the DLUR as a product option can forward the BIND to the LU.

8.1.5 +RSP(BIND)

When the DLUR node receives a +RSP(BIND),

- ***short RSP(BIND) processing -*** the short +RSP(BIND) is converted into an extended short form RSP(BIND) by:
  - zeroing 31 (X′1F′) bytes for the extended short form RSP(BIND)
  - setting the RU code to BIND (X′31′)
  - setting the BIND type to nonnegotiable
  - including the SLU max send size and the PLU max send size from the BIND request
  - setting the pacing fields in the RSP(BIND) for the REX stage to the BIND request values
  - setting the limited resource bit if the link station is a limited resource link (LU 6.2 RSP(BIND)s only)

  and the pacing values for the SLU stage are saved so that fixed session pacing will be used with the window sizes specified in the BIND request;
- ***extended BIND processing -*** the +RSP(BIND) will be reextended if necessary, based on the extended BIND support indicators in the CV X′0C′ in the +RSP(ACTLU) - this involves reinserting

the FQPCID control vector and the Route Selection control vector (if one was received in the BIND request) and turning on the CVs included indicator;

- ***WBRI processing -*** a DLUR can, as an implementation option, support segment reassembly at its BF; whether or not a DLUR provides this support is reflected in how it sets the Whole BIUs Required Indicator (WBRI) - WBRI=0 means sending node supports receipt of segments on this session while WBRI=1 means sending node does not support receipt of segments (this processing could apply to RSP(BIND)s built as well as received by the DLUR):
  - if WBRI is valid (not a reserved field), use the received value
  - if WBRI is reserved, act as if DLUR received WBRI=0; if segments are received from the PLU, the DLUR can either:
    - reassemble the segments and attempt to send the whole message to the SLU (only if the DLUR supports segment reassembly); if the reassembled message is too large, the DLUR can segment the message and send the segments to the SLU
    - attempt to send the segments received from the PLU to the SLU; if a segment is too large, the DLUR can segment the PLU segment and send the new segments to the SLU
  - if WBRI=1 and DLUR BF can support segment reassembly, set WBRI to 0
  - if WBRI=1 and DLUR BF cannot support segment reassembly, set WBRI to 1;
- ***adaptive session pacing processing -*** the Adaptive Session-Level Pacing support indicator value will be restored to its remembered value, unless the SLU is locally attached and the negotiable or non-negotiable RSP(BIND) does not indicate adaptive session pacing support (ASP indicator set to 0), in which case the DLUR can leave the ASP indicator at 0 regardless of the remembered value;
- ***session awareness processing -*** SESSST will be built and sent to the SSCP on the CP-SVR pipe
  - When a DLUR receives a +RSP(BIND) for a dependent LU and there is no active CP-SVR pipe servicing that dependent LU, the DLUR should try to activate a CP-SVR pipe. When the CP-SVR pipe is activated, the SESSST should not be sent on the new CP-SVR pipe. The active LU-LU session awareness information will be sent to the SSCP in the DLUS by way of the CV X′2A′ added to the +RSP(ACTLU) by the DLUR (see 10.1.2, “DLUR/S Dependent LU Session Awareness” on page 10-2 for more information on session awareness).

and the +RSP(BIND) will be forwarded to the appropriate internal or external component.

8.1.6 LU-LU Session Deactivation

When the DLUR node receives a -RSP(INIT-SELF) or detects a link failure affecting an LU-LU session, it will discard the saved URC and the saved PLU name associated with the dependent LU.

When the DLUR node receives a -RSP(BIND), UNBIND, or detects a link failure affecting an LU-LU session, it will reset the session count for the associated dependent LU to zero.

The UNBIND will be unextended if necessary, based on the extended BIND support indicators in the CV X′0C′ in the +RSP(ACTLU) - this involves:

- ***all UNBIND types except X′FE′ -*** truncating the UNBIND RU to only the request code plus the UNBIND type;
- ***UNBIND type X′FE′ -*** truncating the UNBIND RU to only the request code, the UNBIND type, and the sense data.

No reextending of the RSP(UNBIND) is required.

8.2 DLUR EN Tail Vector Registration

In order for the DLUS node to send Locate Find/Found flows with the appropriate associated resource hierarchy as described above, it will have to know the DLUR tail vectors. In the case where the DLUR node is a NN this will not be necessary. The EN DLUR node will be required to register its tail vectors with the DLUS node; this registration will be done only when the DLUS and DLUR are in the same subnet.

- The EN DLUR can determine whether or not it is in the same subnet with the DLUS by examining the Locates it receives during CP-SVR pipe activation, either:
  - the Locate sent by the DLUS to activate its conwinner CPSVRMGR session, or
  - the Locate reply sent by the DLUS in response to the Locate sent by the DLUR to activate its conwinner CPSVRMGR session

  In the Intersubnetwork Search (X′82′) Locate control vector there is an intersubnetwork TGs crossed count:

  - if the count is zero, the DLUS and DLUR are in the same subnet
  - if the count is nonzero, the DLUS and DLUR are not in the same subnet

An EN DLUR in the same subnet as the DLUS will register or not register its tail vectors dependent on the DLUS′s setting in the DLUR/S Capabilities control vector of the indicator for support of the RECEIVE_TDU_TP (X′22F0F0F4′) service transaction program (see 8.2.1, “DLUR/S Capabilities Exchange” on page 8-7 for more information about when the DLUS will set the indicator):

- if the indicator is off, the EN DLUR will not register its tail vectors
- if the indicator is on, the EN DLUR will register its tail vectors

DLUR tail vector registration will involve sending TOPOLOGY_DATABASE_UPDATEs (TDUs) to the DLUS node:

1. at initial TGV registration,
   a. if the DLUR and the DLUS are adjacent, then the DLUR may optionally send the TDUs over the CP-CP session between the two nodes
   b. otherwise the TDUs will be sent over the CP-SVR pipe to the DLUS node
2. if the CP-SVR pipe is selected for initial registration:
   a. continue to use it for subsequent TGV registration even if a CP-CP session between adjacent DLUR and DLUS is activated later
   b. if the CP-SVR pipe fails, once the pipe is reactivated (to either the same or a different DLUS), return to step 1 and send all tail vectors
3. if the CP-CP session is selected for initial registration:
   a. continue to use it for subsequent TGV registration
   b. if the CP-CP session fails, return to step 1 and send all tail vectors
   c. if the CP-SVR pipe fails, once the pipe is reactivated (to either the same or a different DLUS), return to step 1 and send all tail vectors

The DLUS node will receive, but not propagate, these TDUs.

This will require the DLUR and DLUS CPs to be modified such that the TDU send and receive TPs can also run over the CP-SVR pipe using the CPSVRMGR mode. No special CP-SVR pipe should be brought up for this purpose. The existing CP-SVR pipe should be used. Since a DLUR may have CP-SVR pipes with multiple DLUS nodes, these TDUs must be sent over each CP-SVR pipe to each partner DLUS node.

8.2.1 DLUR/S Capabilities Exchange

TDUs can be sent on the pipe any time after CP-SVR pipe activation completes. Activation includes the exchange of DLUR/S Capabilities control vectors, so TDUs can be sent by the DLUR after:

- DLUR-initiated pipe activation: receipt of +RSP(REQACTPU) from DLUS
- DLUS-initiated pipe activation: transmission of +RSP(ACTPU) to DLUS

Included within the DLUR/S Capabilities control vector are fields relevant to EN tail vector registration:

- a Flow Reduction Sequence Number (FRSN) -
  - the DLUR will always set this field to zero in the DLUR/S Capabilities control vector; this will tell the DLUS to throw away any tail vectors previously registered by this DLUR;
  - the DLUS will set this field to the last FRSN value received from the DLUR;
- an indicator for support of the RECEIVE_TDU_TP (X′22F0F0F4′) service transaction program - when set, the sending node supports receipt of TDUs on this session
  - the DLUS may set this indicator on unless the TDU information will not be used for route calculation;
  - the DLUR should set this indicator off.

8.2.2 Initial EN Tail Vector Registration

After CP-SVR pipe activation, the DLUR will send all of its tail vectors to the DLUS (for more information, see “Processing CP_STATUS Signals” in the *SNA APPN Architecture Reference*).

The DLUS will process the received TDUs as described in “TDU Processing” in the *SNA APPN Architecture Reference*, with one exception: these TDUs should not be propagated to other NNs.

8.2.3 EN Tail Vector Registration Updates

When there is a change to a previously registered tail vector, the DLUR will send an update TDU to the DLUS (for more information, see “Processing Resource Updates” in the *SNA APPN Architecture Reference*).

The DLUS will process the received TDUs as described in “TDU Processing” in the *SNA APPN Architecture Reference*, “TDU Processing” in the *SNA APPN Architecture Reference*, with one exception: these TDUs should not be propagated to other NNs.

Chapter 9. Dependent LU Session Establishment Flows

Figure 9-1 on page 9-2 through Figure 9-5 on page 9-14 illustrate how LU-LU sessions are established using the DLUR/S option sets.

Note: Refer to Table 3-1 on page 3-2 for an explantion of the notations used in this chapter.

9.1 USS Flows For SLU Init

Figure 9-1 represents the USS management flows for a SLU-initiated session regardless of the PLU location.

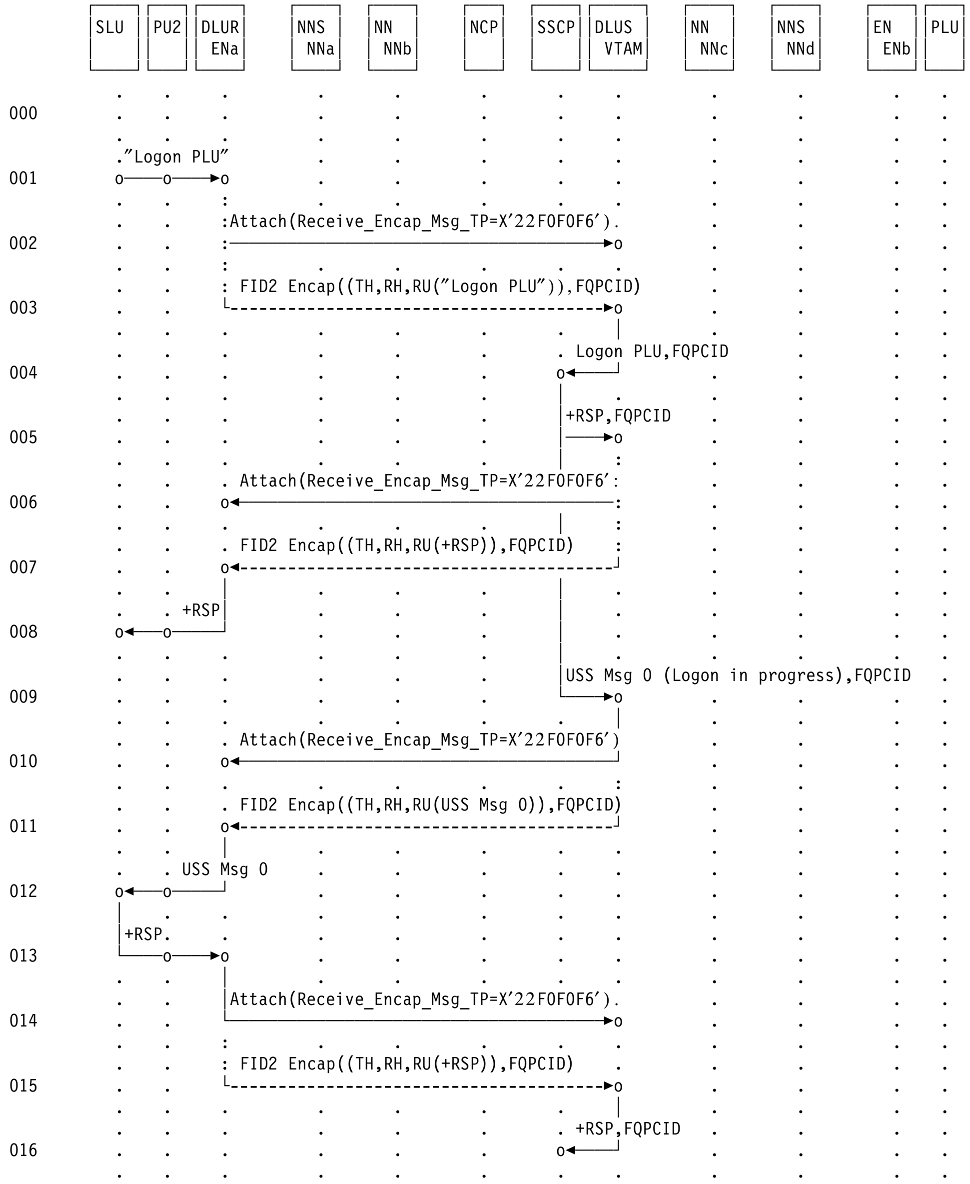


Figure 9-1. Unformatted Session Services SLU-initiated LU-LU session

1. The Logon PLU command is sent from the SLU through the PU to the DLUR component of the EN.
2. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
3. The DLUR component encapsulates the Logon command using the FQPCID and sends it up to the DLUS node via the CP-SVR pipe.
4. The DLUS node removes the FID2 Encapsulation GDS variable and sends the Logon PLU command to the SSCP.
5. The SSCP receives the Logon command and issues a positive response to the LU via the DLUS component.
6. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
7. The DLUS component uses the SSCP-LU session identifiers to encapsulate the +RSP and send it to the DLUR node via the CP-SVR pipe.
8. The DLUR node removes the FID2 Encapsulation GDS variable and forwards the +RSP to the LU via the PU.
9. The SSCP also follows the +RSP to the Logon command with a USS Msg 0 (logon in progress) message.
10. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
11. The DLUS component encapsulates the USS Msg 0 and sends it over the CP-SVR pipe to the DLUR node.
12. The DLUR node removes the FID2 Encapsulation GDS variable and sends the USS Msg 0 to the LU via the PU.
13. The LU responds to receipt of the USS Msg 0 with a +RSP.
14. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
15. The DLUR component encapsulates the +RSP and sends it up the pipe to the DLUS node.
16. The DLUS node removes the FID2 Encapsulation GDS variable and sends the +RSP to the SSCP. At this point, the USS MSG handling is complete, and the session initiation proceeds.

9.2 SLU Init To APPN PLU

Figure 9-2 on page 9-5 ilustrates the flows that occur between the DLUS node and an APPN PLU for a SLU-initiated session that eventually result in a BIND flowing directly between the APPN PLU and DLUR (SLU) nodes. These flows have been added to describe the relationship between DLUR/S architecture and Session Services Extensions.

Prior to the flows illustrated in the diagram, a SLU supported by a DLUR node has requested a session with the PLU. Flows of this type are described in Figure 9-1 on page 9-2. Flows in this diagram begin after the DLUS has been notified of the SLU session initiation request. In this case, the PLU is in the APPN portion of the network, so Session Services Extensions SLU Initiation session flows are used between the NN side of the DLUS interchange node and the APPN CP(PLU).

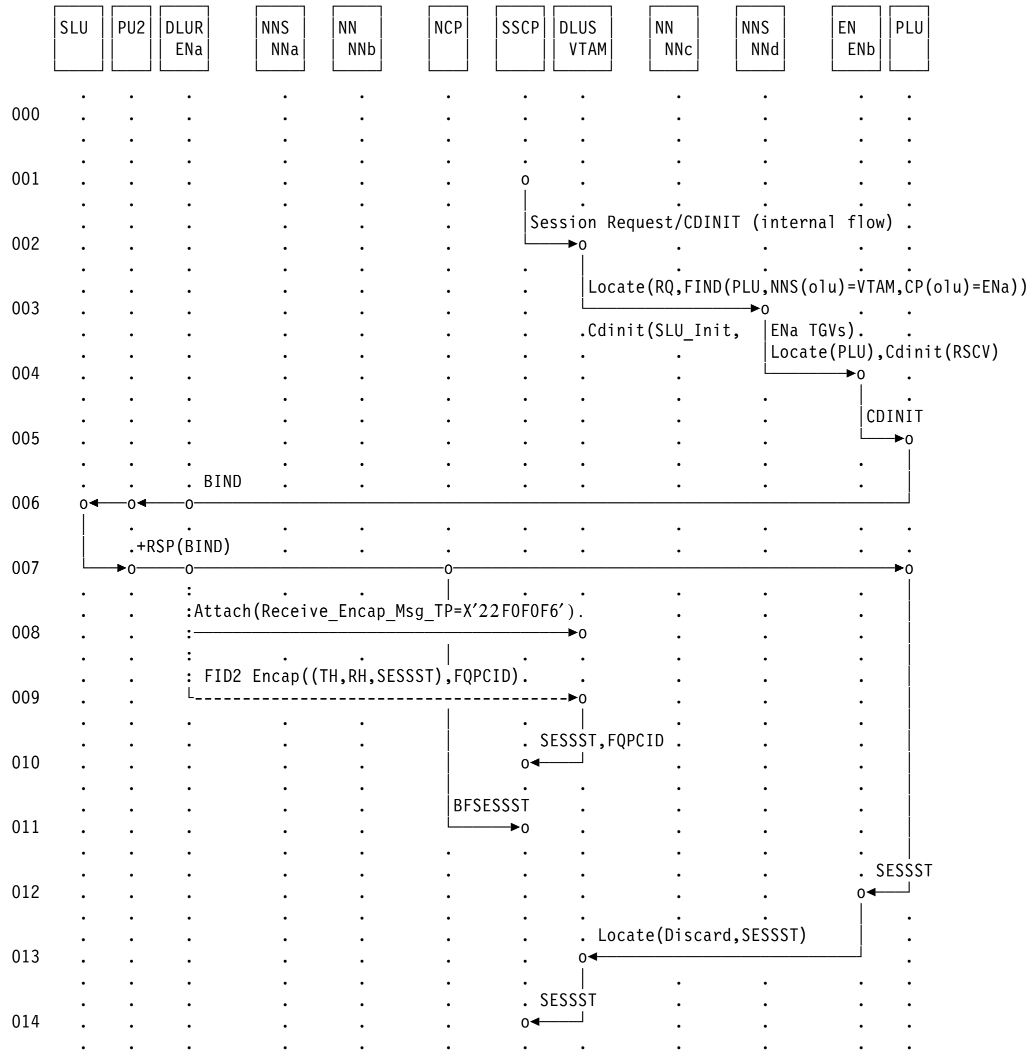


Figure 9-2. Unformatted Session Services SLU-initiated LU-LU session to APPN PLU

1. The session initiation request is passed from the SSCP portion of the interchange node to the NN portion. This signals to the NN that the APPN portion of the network should be searched for the PLU for a SLU-initiated session.
2. The NN sends a Locate Find Cdinit message into the APPN network to locate the PLU. In this case, the location of the PLU is already known, so a directed Locate is issued directly to the CP(PLU) via the NNS(PLU). The Locate hierarchy indicates the NNS(OLU)=VTAM and the CP(OLU)=ENa. This will cause the NNS(PLU) to cache the DLUS node as the NNS of the CP(OLU). The Cdinit specifies the OLU=SLU, initiation type of SLU Init, and the tail vectors of the DLUR node. The DLUR tail vectors are known from previous registration flows between the DLUR and DLUS nodes (see Chapter 8, "LU-LU Session Routing" on page 8-1 for further details). The DLUS-served LU indicator is turned on since the SLU is a dependent LU served by this DLUS.
3. When the NNS(PLU) receives the Locate Find Cdinit from the DLUS node (since it knows the location of the PLU from previous cached entries), it calculates a route between the CP(PLU) and CP(SLU)=DLUR node. This route is computed because the DLUR TGVs have been provided (see Chapter 8, "LU-LU Session Routing" on page 8-1 for further details). The Locate Find Cdinit with RSCV is then passed to the CP(PLU).
4. The CP(PLU) forwards the CDINIT with SLU Session Characteristics (CV X′31′ and CV X′65′) to the PLU.
5. The PLU and CP(PLU) use the session characteristics passed in the Cdinit and the RSCV to construct a BIND image and send it to the CP(SLU). The FQPCID generated by the CP(SLU) is used to identify the BIND. If for any reason the CP(PLU) is incapable of sending a BIND to the CP(SLU) (e.g., PLU at session limit), appropriate sense data is generated and returned to the DLUS node as a negative Cdinit reply.
6. The DLUR node receives the extended BIND image sent from the CP(PLU). Recognizing that this BIND is destined for a dependent LU, the DLUR node processes the BIND (optionally unextending it, removing NETIDs, performing name substitution, setting pacing indicator). This BIND is then forwarded to the SLU via PU2. In the case where the PU is physically downstream from the DLUR node, the DLUR node checks the SLU name in the NS field of the BIND and compares it to its LU name table described in 6.3, "SSCP-LU Session Activation" on page 6-27. This information is then used to determine which link to send the BIND over and what DAF to put in the TH field.
7. A +RSP(BIND) is generated by the SLU and flows along the session path to the PLU. The DLUR processes the +RSP(BIND) (optionally reextending it, setting pacing indicator).
8. Upon receipt of the +RSP(BIND), the DLUR node issues a SESSST RU to notify the SSCP that the session has started. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
9. The DLUR node encapsulates the SESSST RU with the SSCP-LU session identifiers and sends it via the CP-SVR pipe to the DLUS node.
10. The DLUS node receives the encapsulated SESSST command, de-encapsulates it and forwards it to the SSCP.
11. Upon receipt of the +RSP(BIND), NCP creates a BFSESSST and sends it to the SSCP.
12. When the PLU receives the +RSP(BIND), it generates a SESSST.
13. The APPN PLU node must then follow the Locate chain to the DLUS node with a Locate(discard) carrying the SESSST notification to the SSCP.
14. The SESSST signal is then passed from the DLUS component to the SSCP.

9.3 APPN PLU Init To A Dependent APPN SLU

Figure 9-3 illustrates an APPN PLU-initiated session to a dependent APPN SLU node supported by DLUR/S architecture.

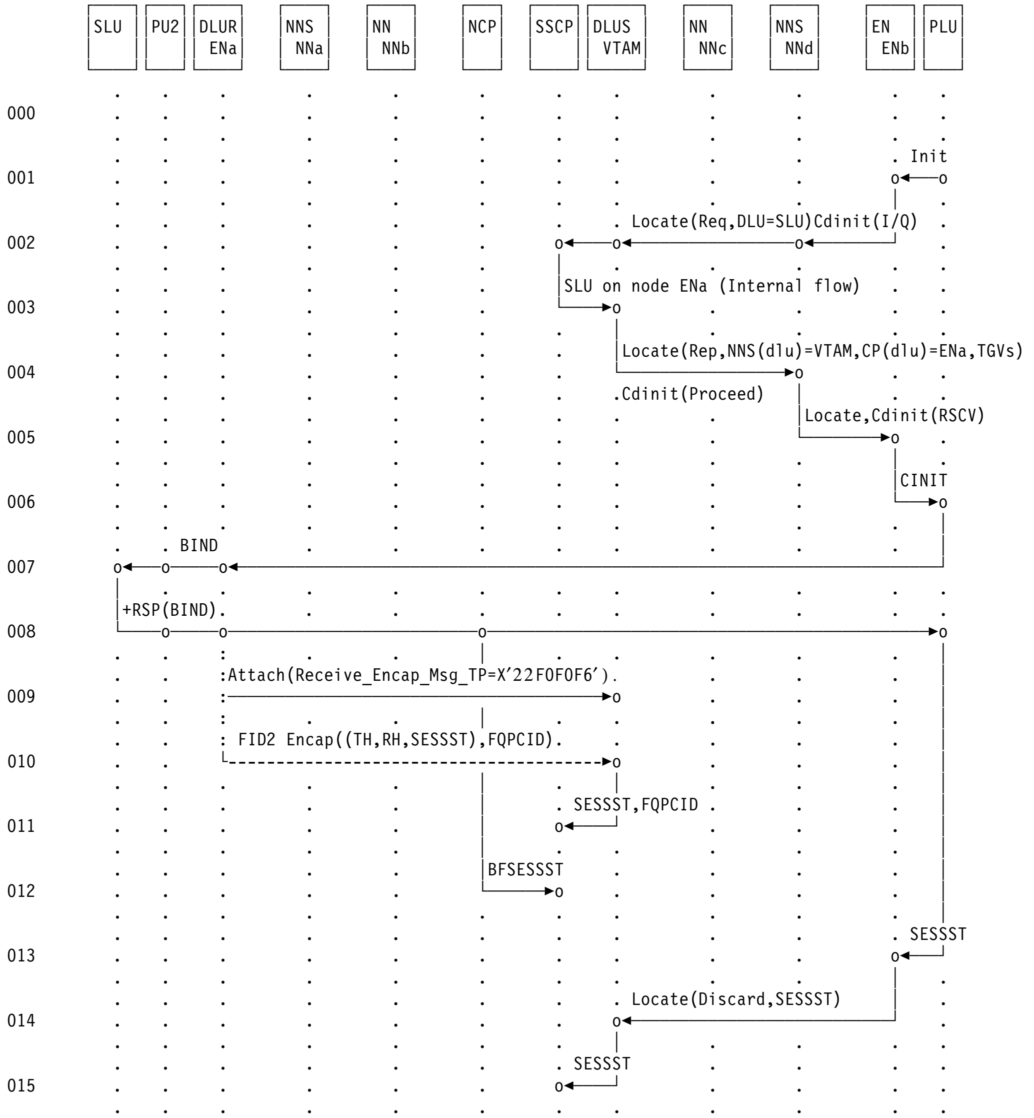


Figure 9-3. APPN PLU-initiated LU-LU session to a dependent SLU

1. The PLU issues an Init for the SLU.
2. The APPN CP(PLU) issues a Locate Find Cdinit for the SLU. This illustration assumes that the NNS(PLU) has already cached information regarding the location of the CP(SLU) and, therefore, sends a directed Locate to the NNS(SLU) or DLUS node. If this were not the case, the NNS(PLU) would issue a broadcast search. When the broadcast search request reached the CP(SLU) (or DLUS), it would respond Locate Found indicating that the CP(PLU) should "resubmit a directed" search to the DLUS node. ***It is important to note that the DLUS node is the NNS(CP(SLU)). The DLUR node would not respond positively to such a search unless the Owning CP Respond bit is on - see Chapter 11, "Multi-Subnet DLUR/S" on page 11-1 for more details.*** The SLU hierarchy would again indicate that the SLU exists on the CP(SLU) which is served by the DLUS node and has the real CP(SLU) tail vectors. Once the DLUS node receives the directed search request for the SLU, it will pass the SLU search request to the SSCP component.
3. The SSCP component indicates to the DLUS component that the SLU is located on the DLUR node.
4. The DLUS responds to the NNS(PLU) that the SLU has been located on the CP(SLU) which is served by the DLUS node (this is so that the NNS(PLU) will cache the DLUS node as the NNS of the dependent SLU) and has the ***real TGVs of the CP(SLU)***. The real TGVs of the CP(SLU) will cause the NNS(PLU) to calculate a route directly from the CP(PLU) to the CP(SLU) (see Chapter 8, "LU-LU Session Routing" on page 8-1 for further details). Since the original request was an Initiate or Queue request, the DLUS node responds with a Cdinit(Proceed). The DLUS-served LU indicator is turned on since the SLU is a dependent LU served by this DLUS.
5. When the NNS(PLU) receives the Locate Found, it uses the CP(SLU) TGVs to calculate a route from the CP(PLU) to the CP(SLU). This RSCV is returned to the CP(PLU).
6. The CP(PLU) receives the Locate Cdinit with SLU session characteristics and RSCV. It forwards the CINIT signal to the PLU which generates a BIND image to be sent to the CP(SLU).
7. The BIND image is then generated and passed to the APPN PLU node which appends an RSCV and FQPCID to extend the BIND. The BIND then flows through the APPN network to the DLUR node. When the DLUR node receives the BIND, it processes it (optionally unextending it, removing NETIDs, performing name substitution, setting pacing indicator) before passing it to the SLU via PU2. In the case of downstream PUs, the DLUR uses the SLU name specified in the NS field of the BIND and its LU table (see 6.3, "SSCP-LU Session Activation" on page 6-27) to determine the target PU and link (if applicable).
8. A +RSP(BIND) is generated by the SLU and flows along the session path to the PLU. The DLUR processes the +RSP(BIND) (optionally reextending it, setting pacing indicator).
9. Upon receipt of the +RSP(BIND), the DLUR node generates a SESSST to notify the SSCP that the session has started. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
10. The DLUR node encapsulates the SESSST with the SSCP-LU session identifiers and sends it up the CP-SVR pipe to the DLUS node.
11. The DLUS node receives the encapsulated SESSST command, de-encapsulates it and forwards it to the SSCP.
12. Upon receipt of the +RSP(BIND), NCP creates a BFSESSST and sends it to the SSCP.
13. When the PLU receives the +RSP(BIND) it generates a SESSST signal to the CP(PLU).
14. The CP(PLU) then uses a Locate(discard) to follow the Locate chain back to the DLUS node and carry the SESSST signal to the SSCP.
15. The SESSST signal is then passed from the DLUS component to the SSCP.

9.4 USS Flows For SLU Init To A Subarea PLU

Figure 9-4 on page 9-10 illustrates the flows for a DLUR-supported SLU that issues an USS SLU Init request when the PLU is located in the subarea portion of the network. Note that the USS MSG 0 (Logon in Progress) message is not issued until the PLU has been located and the CDINIT RU has been processed by the SSCP(PLU).

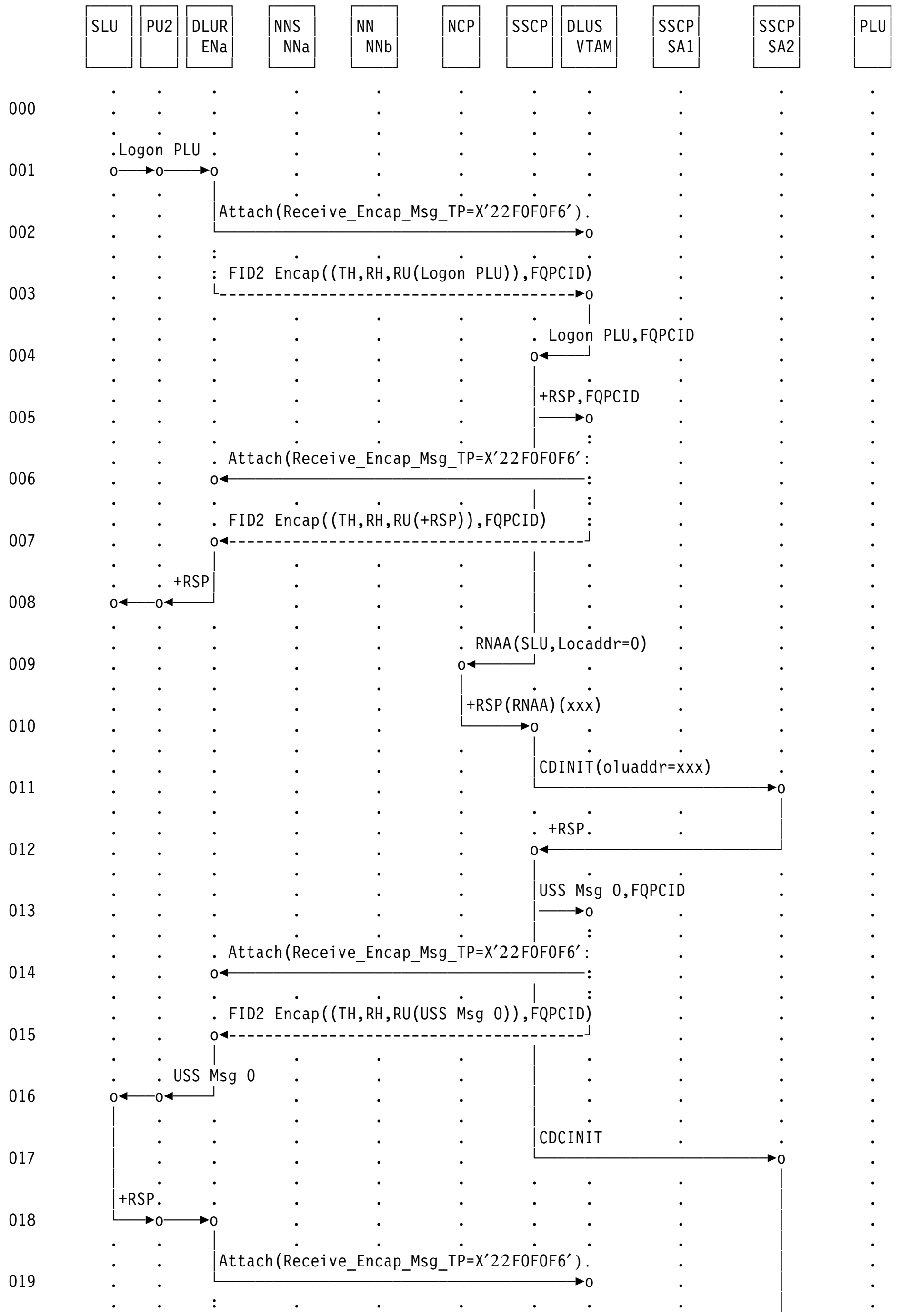
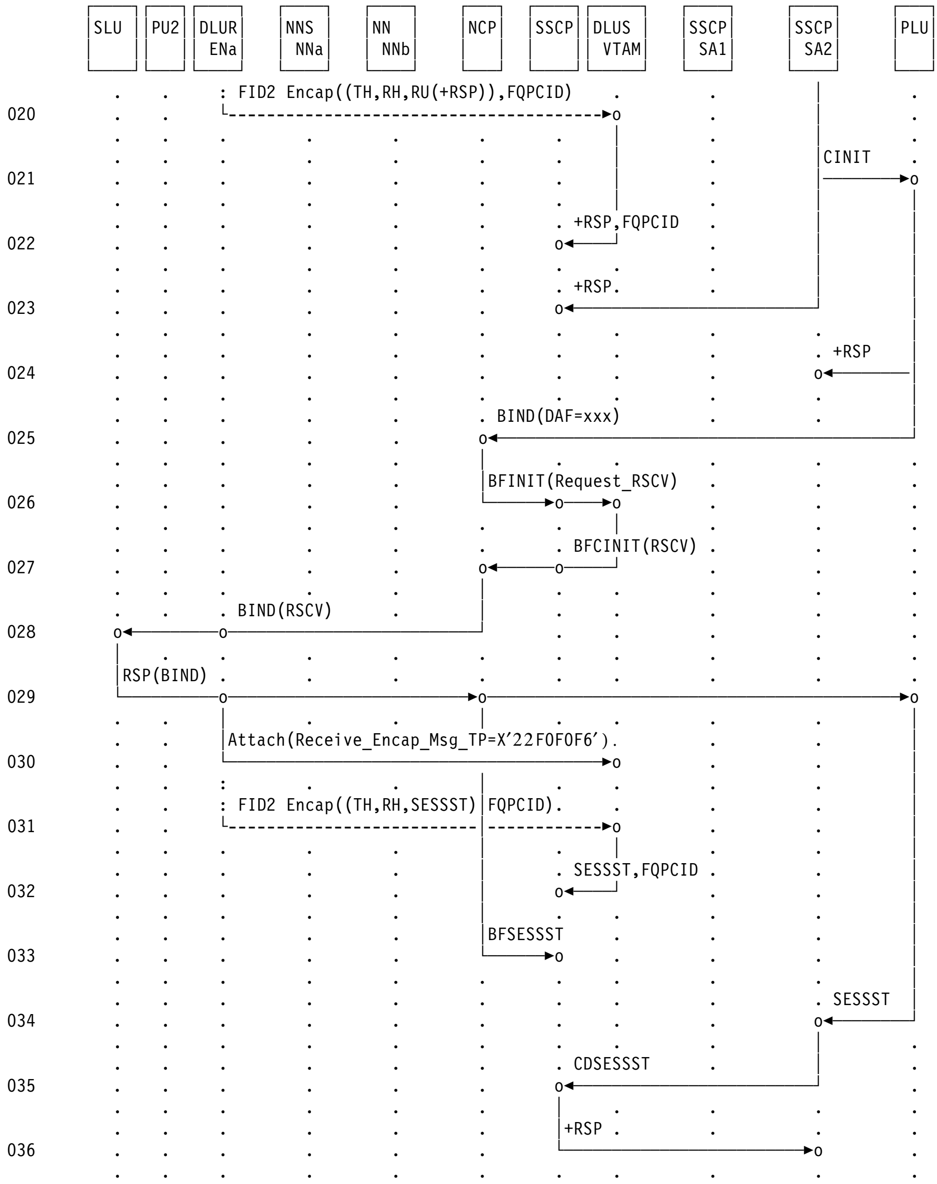


Figure 9-4. Unformatted Session Services SLU-initiated LU-LU session to subarea PLU


SLU
PU2
DLUR ENa
NNS NNa
NN NNb
NCP
SSCP
DLUS VTAM
SSCP SA1
SSCP SA2
PLU
020
FID2 Encap((TH,RH,RU(+RSP)),FQPCID)
021
CINIT
022
+RSP,FQPCID
023
+RSP
024
+RSP
025
BIND(DAF=xxx)
026
BFINIT(Request_RSCV)
027
BFCINIT(RSCV)
028
BIND(RSCV)
029
RSP(BIND)
030
Attach(Receive_Encap_Msg_TP=X'22F0F0F6')
031
FID2 Encap((TH,RH,SESSST) FQPCID)
032
SESSST,FQPCID
033
BFSESSST
034
SESSST
035
CDSESSST
036
+RSP

1. The SLU sends a “Logon PLU” command to the CP (DLUR node) via its PU.
2. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
3. The DLUR component encapsulates the Logon command on the SSCP-LU session and sends it up the CP-SVR pipe to the DLUS node.
4. The DLUS component removes the FID2 Encapsulation GDS variable and forwards the Logon command to the SSCP.
5. The SSCP issues a positive response (+RSP) to receipt of the Logon command.
6. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
7. The DLUS component encapsulates the +RSP and sends it on the encapsulated SSCP-LU session back to the SLU via the DLUR node.
8. The DLUR node removes the FID2 Encapsulation GDS variable and forwards the +RSP to the LU via the PU.
9. In response to the Logon PLU command received from the SLU, the SSCP issues an RNAA to the Boundary Function (BF) NCP. This establishes a subarea address for the SLU.
10. The BF responds with address xxx.
11. The SSCP then issues a CDINIT to SSCP SA2 specifying xxx as the subarea SLU address.
12. The SSCP SA2 issues a CDINIT RSP back to the DLUS SSCP.
13. Receipt of the +RSP to the CDINIT indicates to the SSCP that the PLU has been located. This causes the SSCP to send USS MSG 0 (Logon in progress) to the SLU. This is passed from the SSCP to the DLUS component.
14. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
15. The DLUS component encapsulates the USS MSG 0 on the SSCP-LU session and sends it to the DLUR node.
16. The DLUR node removes the FID2 Encapsulation GDS variable and sends the USS MSG 0 to the LU via PU2.
17. Meanwhile, the DLUS SSCP issues a CDCINIT to SA2 in response to the CDINIT RSP.
18. The LU issues a +RSP to the MSG 0 and sends it to the DLUR component.
19. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
20. The DLUR node encapsulates the message on the SSCP-LU session and sends it to the DLUS node.
21. Meanwhile after receiving the CDCINIT, SA2 forwards the CINIT signal to the PLU.
22. The DLUS component removes the FID2 Encapsulation GDS variable and forwards the USS MSG 0 +RSP to the SSCP.
23. SA2 provides a +RSP to the CDCINIT from the DLUS SSCP.
24. The PLU issues a +RSP to the CINIT from its SSCP, i.e., SA2.
25. The PLU follows the CINIT with a BIND specifying a destination address (DAF) of xxx which was assigned by the SLU BF in step 10.
26. When the BIND reaches the BF, it must request an RSCV to send the BIND into the APPN portion of the composite network. A BFINIT(request_rscv) is issued back to the interchange node.

27. The interchange node responds with a BFCINIT and an RSCV computed to the DLUR node. This RSCV is computed using information resident in the SSCP portion of the DLUS node that specifies the DLUR node supporting the SLU. The DLUR tail vector information is present from previous TGV registration activities (see Chapter 8, “LU-LU Session Routing” on page 8-1 for further information).
28. The BF re-issues the BIND in extended format with the FQPCID and RSCV into the APPN portion of the network. When the DLUR node receives the BIND, it forwards it to the LU. If the LU is on a downstream PU, the DLUR must use the LU table to find the appropriate target PU (see 6.3, “SSCP-LU Session Activation” on page 6-27).
29. Meanwhile, the SLU responds to the BIND with a +RSP(BIND) over the data path to the PLU.
30. Upon receipt of the +RSP(BIND), the DLUR node generates a SESSST to notify the SSCP that the session has started. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
31. The DLUR node encapsulates the SESSST with the SSCP-LU session identifiers and sends it up the CP-SVR pipe to the DLUS node.
32. The DLUS node removes the FID2 Encapsulation GDS variable and sends the SESSST to the SSCP.
33. When the subarea BF receives the +RSP(BIND), it forwards a BFSESSST to the DLUS SSCP.
34. When the PLU receives the +RSP(BIND), it sends a SESSST to its SSCP, i.e., SA2.
35. SA2 then sends a CDSESSST to the SSCP of the SLU, i.e., the DLUS node.
36. The DLUS node issues a +RSP to the CDSESSST.

9.5 USS Flows For LU-LU Session Termination

Figure 9-5 illustrates the flows for a DLUR-supported SLU that issues a USS SLU logoff request.

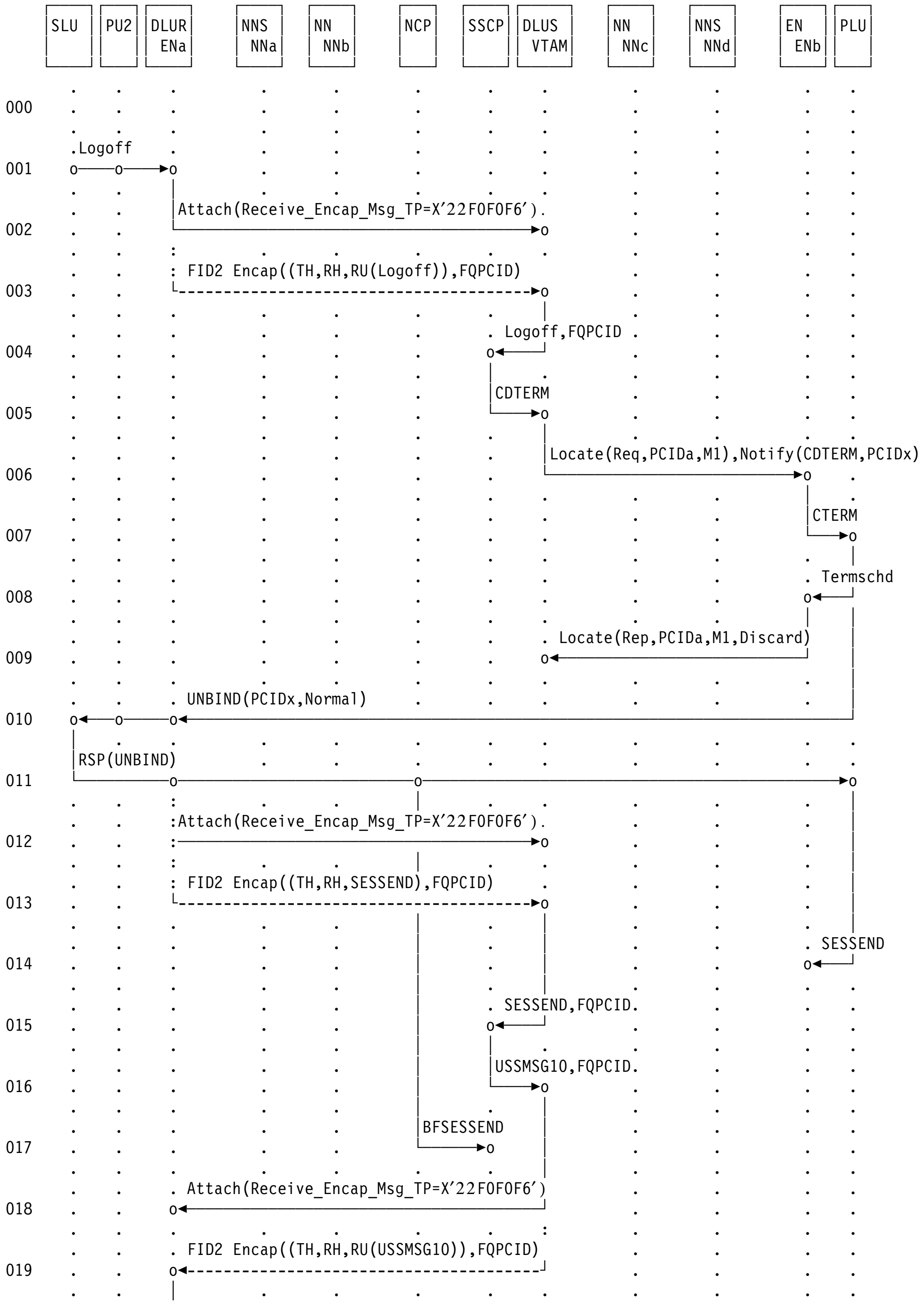


Figure 9-5. Unformatted Session Services LU-LU session termination

```
     ┌───┐ ┌───┐ ┌────┐      ┌────┐   ┌────┐     ┌───┐   ┌────┐┌─────┐     ┌────┐    ┌────┐     ┌────┐┌───┐
     │SLU│ │PU2│ │DLUR│      │NNS │   │NN  │     │NCP│   │SSCP││DLUS │     │NN  │    │NNS │     │EN  ││PLU│
     │   │ │   │ │ ENa│      │ NNa│   │ NNb│     │   │   │    ││ VTAM│     │ NNc│    │ NNd│     │ ENb││   │
     └───┘ └───┘ └────┘      └────┘   └────┘     └───┘   └────┘└─────┘     └────┘    └────┘     └────┘└───┘
       .     . USSMSG10        .        .          .        .     .           .         .           .    .
020    o<────o──────┘          .        .          .        .     .           .         .           .    .
       │     .      .          .        .          .        .     .           .         .           .    .
       │+RSP.       .          .        .          .        .     .           .         .           .    .
021    └─────o─────>o          .        .          .        .     .           .         .           .    .
       .     .      :          .        .          .        .     .           .         .           .    .
       .     .      :Attach(Receive_Encap_Msg_TP=X'22F0F0F6').    .           .         .           .    .
022    .     .      :────────────────────────────────────────────>o           .         .           .    .
       .     .      :          .        .          .        .     .           .         .           .    .
       .     .      : FID2 Encap((TH,RH,RU(+RSP)),FQPCID)         .           .         .           .    .
023    .     .      └-------------------------------------------->o           .         .           .    .
       .     .      .          .        .          .        .     │           .         .           .    .
       .     .      .          .        .          .        . +RSP,FQPCID     .         .           .    .
024    .     .      .          .        .          .        o<────┘           .         .           .    .
       .     .      .          .        .          .        .     .           .         .           .    .
```

1. The SLU sends a “Logoff PLU” command to the CP (DLUR node) via its PU.
2. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
3. The DLUR component encapsulates the Logoff command on the SSCP-LU session and sends it on the CP-SVR pipe to the DLUS node.
4. The DLUS component removes the FID2 Encapsulation GDS variable and passes the Logoff command to the SSCP.
5. The SSCP then issues a CDTERM to its DLUS component.
6. The DLUS issues a Locate Notify Cdterm for the PLU. The Locate request is sent with a new FQPCID (PCIDa) generated by the DLUS node. The Notify Cdterm indicates that the termination of the session is requested and contains the FQPCID (PCIDx) of the session to be terminated. Note that both of these FQPCIDs are different from the FQPCID included in the FID2 Encapsulation GDS variable which correlates encapsulated messages with the correct dependent LU′s PU.
7. When the Locate request is received by the CP(PLU), it forwards a CDTERM signal to the PLU.
8. Once termination has been scheduled, the PLU notifies the CP(PLU).
9. The CP(PLU) sends a Locate reply, with the “discard” indicator set, back to the DLUS node to indicate the status of the terminate request.
10. After the session has quiesced, the PLU generates an UNBIND image which flows along the session path to the SLU.
11. The SLU responds to the UNBIND with a RSP(UNBIND) over the data path to the PLU.
12. Upon receipt of the RSP(UNBIND), the DLUR node generates a SESSEND to notify the SSCP that the session has ended. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
13. The DLUR node encapsulates the SESSEND with the SSCP-LU session identifiers and sends it on the CP-SVR pipe to the DLUS node.
14. Upon receipt of the RSP(UNBIND), the PLU generates a SESSEND signal to the CP(PLU).
15. The DLUS node removes the FID2 Encapsulation GDS variable and sends the SESSEND to the SSCP.
16. The SSCP then sends a USSMSG10 (“Good Morning” message) to the DLUS component of the DLUS node.
17. Upon receipt of the RSP(UNBIND), the subarea BF creates a BFSESSEND and sends it to its SSCP.
18. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
19. The DLUS component of the node uses the SSCP-LU session identifiers to encapsulate the USSMSG10 and send it on the CP-SVR pipe to the DLUR node.
20. The DLUR node removes the FID2 Encapsulation GDS variable and forwards the USSMSG10 message to the PU and LU.
21. The LU generates an acknowledgment to the USSMSG10 and sends the response to the PU and DLUR component.
22. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
23. The DLUR component will use the SSCP-LU session identifier information to encapsulate the positive response and send it over the CP-SVR pipe to the DLUS node.

24. The DLUS node receives the response, strips the FID2 Encapsulation GDS variable, and forwards the response to the SSCP.

Chapter 10. Dependent LU Session Awareness Management

Session awareness (SAW) involves an SSCP being informed of the start and end of LU-LU sessions. For a given LU-LU session, session awareness RUs are sent for both the PLU and the SLU. Host application LUs send their own Session Started (SESSST) and Session Ended (SESSEND) RUs. The Boundary Function (BF) builds session awareness RUs for the other LUs, SESSST and SESSEND for dependent LUs, and BFSESSST and BFSESSEND for independent LUs.

Session awareness also includes informing a takeover SSCP of the current active LU-LU sessions. This information is carried in RSP(ACTLU) for dependent LUs, and in BFSESSINFO for independent LUs.

Since a DLUR will provide a BF for a dependent LU, the DLUR will provide session awareness information to the SSCP on the SSCP-LU session via the DLUR′s CP-SVR pipe with its DLUS.

A DLUR-attached dependent LU can only be a SLU; session awareness function for PLUs remains unchanged. Session awareness function for independent LUs also remains unchanged.

10.1 Functional Overview

10.1.1 Pre-DLUR/S Dependent LU Session Awareness

Prior to DLUR/S, the dependent LU′s SSCP-LU and LU-LU sessions must route through the same BF-NCP. For dependent LU session awareness, this means that:

- LU-LU Session Activation
  - Upon LU-LU session activation, the BF-NCP builds a SESSST and sends it to the SSCP.
- SSCP Takeover (Takeover and Giveback)
  - During SSCP takeover, the BF-NCP intercepts the RSP(ACTLU) built by the dependent LU and, if the dependent LU has an active LU-LU session, inserts several control vectors, including a Session Information (X′2A′) control vector in the RU before sending it on to the SSCP (the same also occurs during SSCP giveback).
- LU-LU Session Deactivation (Normal and Abnormal)
  - Upon LU-LU session deactivation, the BF-NCP builds a SESSEND and sends it to the SSCP.

10.1.2 DLUR/S Dependent LU Session Awareness

With the DLUR/S function, the DLUR will assume BF responsibilities previously handled by the BF-NCP for dependent LUs associated with it. The session awareness responsibilities will include building the SESSST and SESSEND; including in these RUs, as well as the RSP(ACTLU), the proper dependent LU session information; and sending all of these RUs to the DLUS′s SSCP on the SSCP-LU session encapsulated in the CP-SVR pipe.

As part of the shift of function to the DLUR, the BF-NCP will now consider a DLUR-attached dependent LU to be an independent LU. This means that there will be two sources of session awareness for the same dependent LU:

1. the DLUR will build and send dependent LU session awareness RUs to the DLUS′s SSCP;
2. the BF-NCP through which the dependent LU′s LU-LU session flows will build and send independent LU session awareness RUs to its SSCP.

The RUs built or updated by the DLUR and the BF-NCP for DLUR-attached dependent LU session awareness are listed in Table 10-1:

Table 10-1. DLUR/S Dependent LU Session Awareness RUs

	DLUR (dependent LU SAW)	**BF-NCP (independent LU SAW)**
LU-LU Session Activation	SESSST	BFSESSST
SSCP Takeover	RSP(ACTLU)	BFSESSINFO
LU-LU Session Deactivation	SESSEND	BFSESSEND

There are several considerations which are introduced with having two sources of session awareness for the same dependent LU:

- The contents of the dependent LU and independent LU SAW RUs will not be identical. The DLUR will not be aware of the network address of the dependent LU′s LU-LU session partner; this information will only appear in the independent LU SAW RUs .
- The SSCP receiving a dependent LU SAW RU may not be the same SSCP which will receive the corresponding independent LU SAW RU. This may happen since, with DLUS, the dependent LU′s SSCP-LU and LU-LU sessions may now route through different BF-NCPs, which may be owned by different SSCPs. The SSCP receiving the independent LU SAW RU need not be in an interchange node with a DLUS; the SSCP receiving the dependent LU SAW RU must have a DLUS associated with it.
- The LU-LU session′s route may not include a BF-NCP at all, in which case no independent LU SAW RU will be built.
- Even if both the dependent LU and independent LU SAW RUs are sent to the same SSCP, there is no guarantee as to which will arrive first.

So the DLUS′s SSCP may receive both RUs and correlate them, or it may only receive one of the RUs. Correlation will be done via the FQPCID (X′60′) control vector which is included in each of these RUs.

Not only are the contents of the dependent LU and independent LU SAW RUs different, but also the contents of the dependent LU SAW RUs built by a DLUR and a BF-NCP are different. New formats of the dependent LU SAW RUs have been defined for the DLUR to build and send to the DLUS′s SSCP; these formats have several items excluded (see 13.3, “Dependent LU Session Awareness Format Changes” on page 13-7 for details):

- The DLUR is not aware of the network address of the dependent LU′s LU-LU session partner; this information only appears in the corresponding independent LU SAW RUs. The new formats of some of the dependent LU SAW RUs (SESSST, SESSEND) do not include a session key, which is where the PLU and SLU network addresses are carried today.
- Control vectors included in some of the dependent LU SAW RUs (SESSST, RSP(ACTLU)) by the BF-NCP, such as VR-ER Mapping Data (X′1E′) and XRF/Cryptography (X′68′), are not included by the DLUR in its dependent LU SAW RUs, since it has no VR, ER, or XRF information.

10.2 Dependent LU Session Awareness Flows

Note: Readers should reference Figure B-2 on page B-3 for an illustration of the sample network configuration used in the flows described in this section of the document.

10.2.1 LU-LU Session Activation Flows

The following are different dependent LU session awareness flow scenarios in the DLUR/S environment for LU-LU session activation:

1. the dependent LU′s SSCP-LU and LU-LU sessions route through the same BF-NCP
   - the SESSST and BFSESSST are received by the DLUS′s SSCP
2. the dependent LU′s SSCP-LU and LU-LU sessions route through different BF-NCPs owned by the same SSCP (the DLUS′s SSCP)
   - the SESSST and BFSESSST are received by the DLUS′s SSCP
3. the dependent LU′s SSCP-LU and LU-LU sessions route through different BF-NCPs owned by different SSCPs
   - only the SESSST is received by the DLUS′s SSCP; the BFSESSST is received by a different SSCP
4. the dependent LU′s LU-LU session does not route through a BF-NCP
   - no BFSESSST is built; only the SESSST is received by the DLUS′s SSCP

The DLUR will build a new format of SESSST, Format 3; it differs from the current Format 1 in that:

- it will not carry several control vectors found in Format 1
- it will include the BIND image control vector

(see 13.3, “Dependent LU Session Awareness Format Changes” on page 13-7 for the new SESSST format).

10.2.1.1 SSCP-LU, LU-LU Sessions Routed Through Same BF-NCP

In Figure 10-1, SSCP-PU and SSCP-LU sessions have been established from SSCP1 to PU2 and LUa, respectively, over the CP-SVR pipe between ENa and VTAM1. This pipe routes through the BF in NCP10, as will the LU-LU session between LUa and LUb. NCP10 is owned by SSCP1, which will receive all of the session awareness RUs concerning LUa.

Note: Refer to Table 3-1 on page 3-2 for an explanation of the notations used in the diagrams in this chapter.

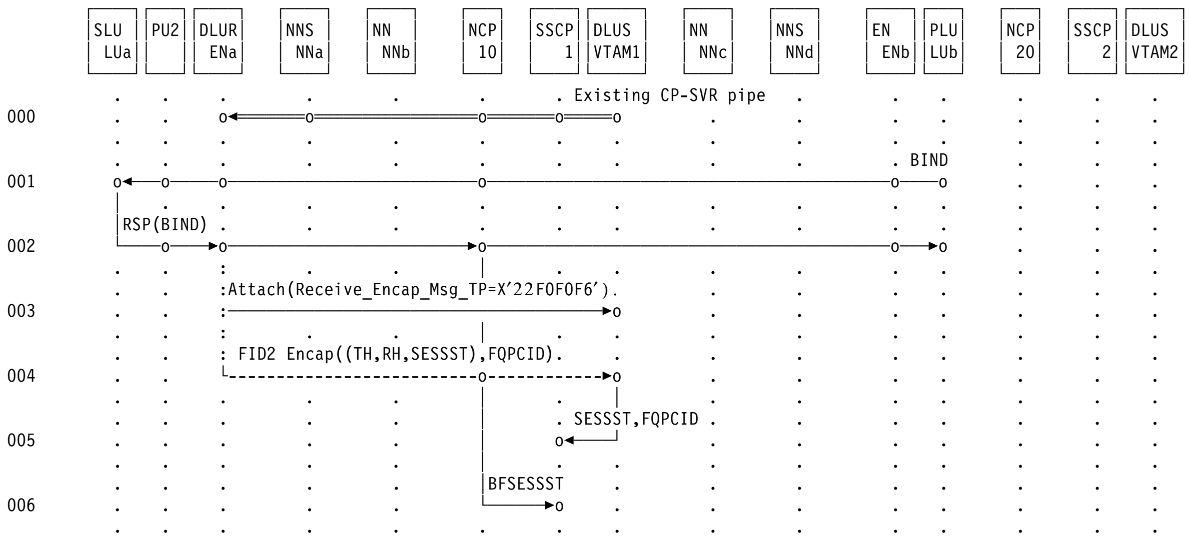


Figure 10-1. LU-LU session activation - SSCP-LU, LU-LU sessions routed through same BF-NCP

1. BIND is sent by LUb to LUa, routing through ENb, NNd, NNc, NCP10, NNb, NNa, ENa, and PU2.
2. RSP(BIND) is sent by LUa to LUb over the same route.
3. Upon receipt of RSP(BIND), the DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
4. The DLUR in ENa creates a Format 3 SESSST and encapsulates it, sending it on the CP-SVR pipe to the DLUS in VTAM1.
5. DLUS de-encapsulates the SESSST and forwards it to SSCP1.
6. Upon receipt of RSP(BIND), NCP10 creates a BFSESSST and sends it to SSCP1. SSCP1 correlates the BFSESSST with the SESSST by the FQPCID contained in each RU.

10.2.1.2 SSCP-LU, LU-LU Sessions Routed Through Different BF-NCPs (Same SSCP)

In Figure 10-2, SSCP-PU and SSCP-LU sessions have been established from SSCP1 to PU2 and LUa, respectively, over the CP-SVR pipe between ENa and VTAM1. This pipe routes through the BF in NCP10. The LU-LU session between LUa and LUb will route, however, through NCP20. Both NCPs are owned by SSCP1, so all session awareness RUs concerning LUa will be sent to SSCP1.

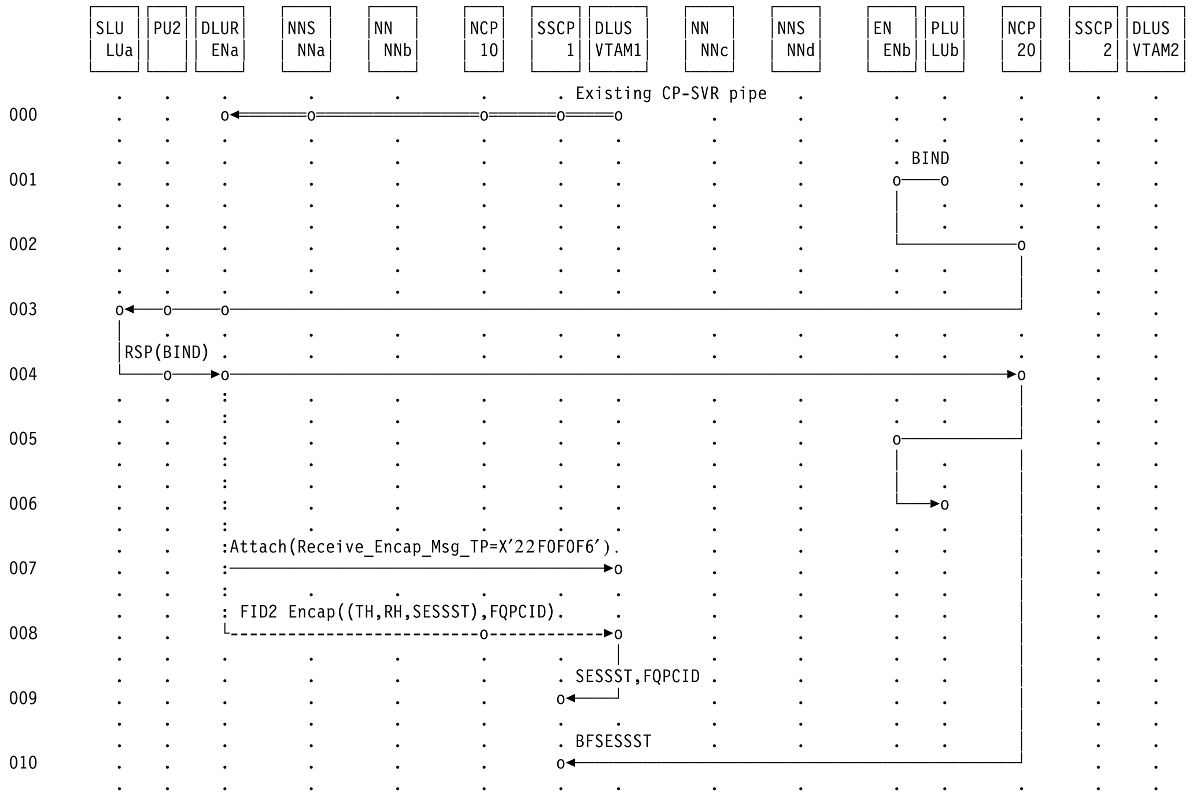


Figure 10-2. LU-LU session activation - SSCP-LU, LU-LU sessions routed through different BF-NCPs (same SSCP)

1. BIND is sent by LUb to LUa, routing through ENb, NNd, NNc, NCP20, NNb, NNa, ENa, and PU2.
2.
3.
4. RSP(BIND) is sent by LUa to LUb over the same route.
5.
6.
7. Upon receipt of RSP(BIND), the DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
8. The DLUR in ENa creates a Format 3 SESSST and encapsulates it, sending it on the CP-SVR pipe to the DLUS in VTAM1.
9. DLUS de-encapsulates the SESSST and forwards it to SSCP1.
10. Upon receipt of RSP(BIND), NCP20 creates a BFSESSST and sends it to SSCP1. SSCP1 correlates the BFSESSST with the SESSST by the FQPCID contained in each RU.

10.2.1.3 SSCP-LU, LU-LU Sessions Routed Through Different BF-NCPs (Different SSCPs)

In Figure 10-3, SSCP-PU and SSCP-LU sessions have been established from SSCP1 to PU2 and LUa, respectively, over the CP-SVR pipe between ENa and VTAM1. This pipe routes through the BF in NCP10. The LU-LU session between LUa and LUb will route, however, through NCP20. NCP10 is owned by SSCP1, and NCP20 is owned by SSCP2. Therefore, the dependent LU session awareness RUs concerning LUa will be sent to SSCP1, while LUa′s independent LU session awareness RUs will be sent to SSCP2.

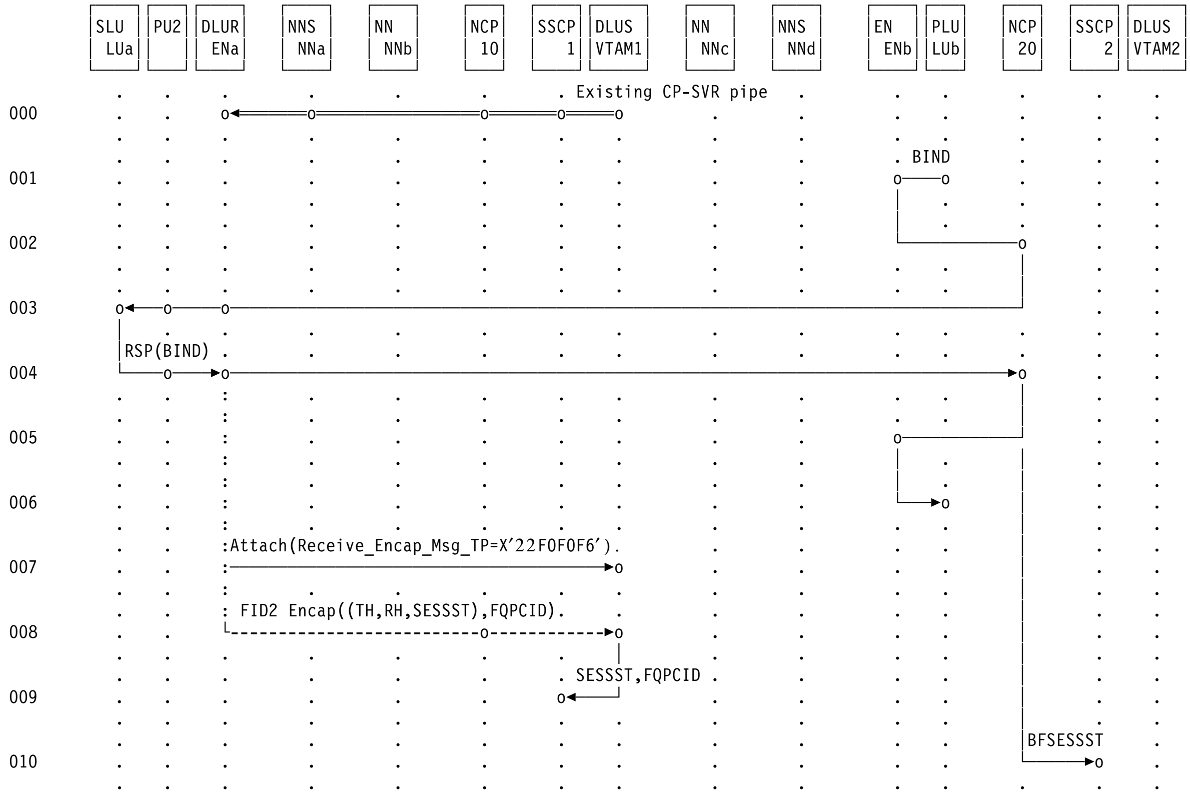


Figure 10-3. LU-LU session activation - SSCP-LU, LU-LU sessions routed through different BF-NCPs (different SSCPs)

1. BIND is sent by LUb to LUa, routing through ENb, NNd, NNc, NCP20, NNb, NNa, ENa, and PU2.
2.
3.
4. RSP(BIND) is sent by LUa to LUb over the same route.
5.
6.
7. Upon receipt of RSP(BIND), the DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
8. The DLUR in ENa creates a Format 3 SESSST and encapsulates it, sending it on the CP-SVR pipe to the DLUS in VTAM1.
9. DLUS de-encapsulates the SESSST and forwards it to SSCP1.
10. Upon receipt of RSP(BIND), NCP20 creates a BFSESSST and sends it to SSCP2.

10.2.1.4 LU-LU Session Not Routed Through A BF-NCP

In Figure 10-4, SSCP-PU and SSCP-LU sessions have been established from SSCP1 to PU2 and LUa, respectively, over the CP-SVR pipe between ENa and VTAM1. This pipe routes through the BF in NCP10. The LU-LU session between LUa and LUb will not route, however, through any NCP. Therefore, SSCP1 will only receive dependent LU session awareness RUs concerning LUa.

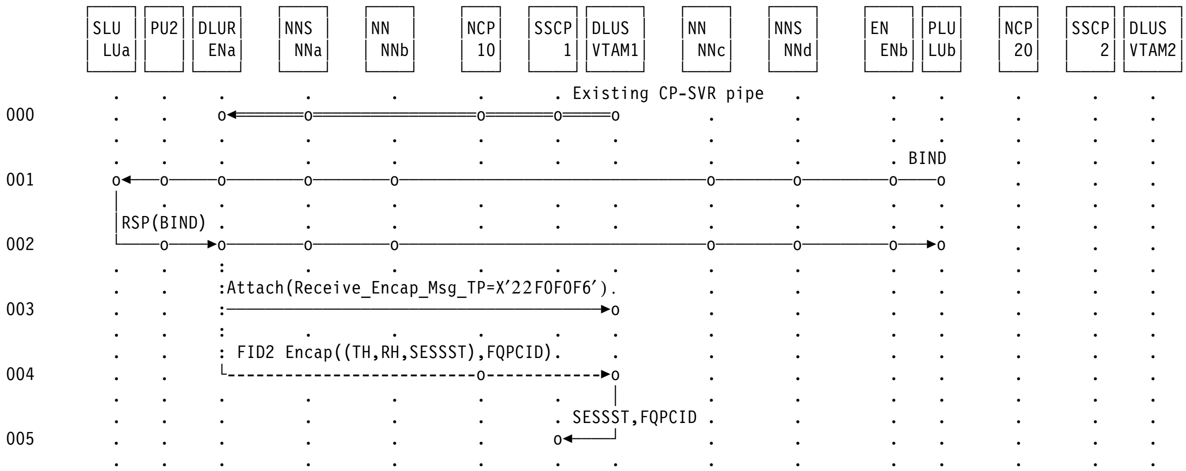


Figure 10-4. LU-LU session activation - LU-LU session not routed through a BF-NCP

1. BIND is sent by LUb to LUa, routing through ENb, NNd, NNc, NNb, NNa, ENa, and PU2.
2. RSP(BIND) is sent by LUa to LUb over the same route.
3. Upon receipt of RSP(BIND), the DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
4. The DLUR in ENa creates a Format 3 SESSST and encapsulates it, sending it on the CP-SVR pipe to the DLUS in VTAM1.
5. DLUS de-encapsulates the SESSST and forwards it to SSCP1.

10.2.2 SSCP Takeover and Giveback Flows

It is assumed in this section that the CP-SVR pipe is terminated due to the loss of the DLUS or of the route to the DLUS, and that the SSCP takeover includes the reestablishment of the CP-SVR pipe to the same or another DLUS. So the BF-NCP through which the CP-SVR pipe, which will include the encapsulated SSCP-LU session, is routed must be owned by the takeover DLUS′s SSCP.

If the dependent LU′s LU-LU session is routed through a different BF-NCP than the SSCP-LU session, and the LU-LU session′s BF-NCP loses its SSCP or the route to its SSCP, the takeover SSCP of that BF-NCP does not have to be a DLUS′s SSCP.

For SSCP takeover, the following are different dependent LU session awareness flow scenarios in the DLUR/S environment:

1. the dependent LU′s SSCP-LU and LU-LU sessions route through the same BF-NCP
   - the RSP(ACTLU) and BFSESSINFO are received by the same takeover DLUS′s SSCP
2. the dependent LU′s SSCP-LU and LU-LU sessions route through different BF-NCPs owned by the same SSCP
   - if both BF-NCPs are acquired by the same takeover SSCP, the RSP(ACTLU) and BFSESSINFO are received by the same takeover SSCP (the takeover DLUS′s SSCP)
   - if the BF-NCPs are acquired by different takeover SSCPs, the RSP(ACTLU) and BFSESSINFO are received by different takeover SSCPs
3. the dependent LU′s SSCP-LU and LU-LU sessions route through different BF-NCPs owned by different SSCPs
   - if the BF-NCP routing the SSCP-LU session is acquired by a takeover SSCP, the RSP(ACTLU) is received by the takeover SSCP
     - the takeover SSCP must be a DLUS′s SSCP
   - if the BF-NCP routing the LU-LU session is acquired by a takeover SSCP, the BFSESSINFO is received by the takeover SSCP
     - the takeover SSCP may or may not be the SSCP owning the other BF-NCP; the takeover SSCP need not be a DLUS′s SSCP
4. the dependent LU′s LU-LU session does not route through a BF-NCP
   - no BFSESSINFO is built; only the RSP(ACTLU) is received by the takeover DLUS′s SSCP

The DLUR modifies the RSP(ACTLU) differently than the BF-NCP. Several control vectors are not included, and the Session Information (X′2A′) control vector is different (see 13.3, “Dependent LU Session Awareness Format Changes” on page 13-7 for the new formats).

For SSCP giveback, the session awareness flows are similar to SSCP takeover. The main difference is in the search for a new DLUS, i.e., trying to activate a CP-SVR pipe with a DLUS other than the one whose SSCP issued the DACTPU(giveback).

All of the flows in this section assume ANS=CONT is defined for the DLUR-attached PU; if instead ANS=STOP is defined, any active LU-LU sessions are deactivated along with the SSCP-PU and SSCP-LU sessions, and no session awareness is sent during takeover or giveback.

10.2.2.1 SSCP-LU, LU-LU Sessions Routed Through Same BF-NCP

In Figure 10-5 on page 10-13, SSCP-PU and SSCP-LU sessions have been established from SSCP1 to PU2 and LUa, respectively, over the CP-SVR pipe between ENa and VTAM1. This pipe routes through the BF in NCP10, as does the LU-LU session between LUa and LUb. NCP10 is owned by SSCP1.

An outage occurs in the SSCP1/VTAM1 host, and after the host is reactivated, a new pipe is established between ENa and VTAM1 (there was no backup host designated to take over for SSCP1). SSCP1 receives all session awareness RUs concerning the active LU-LU session.

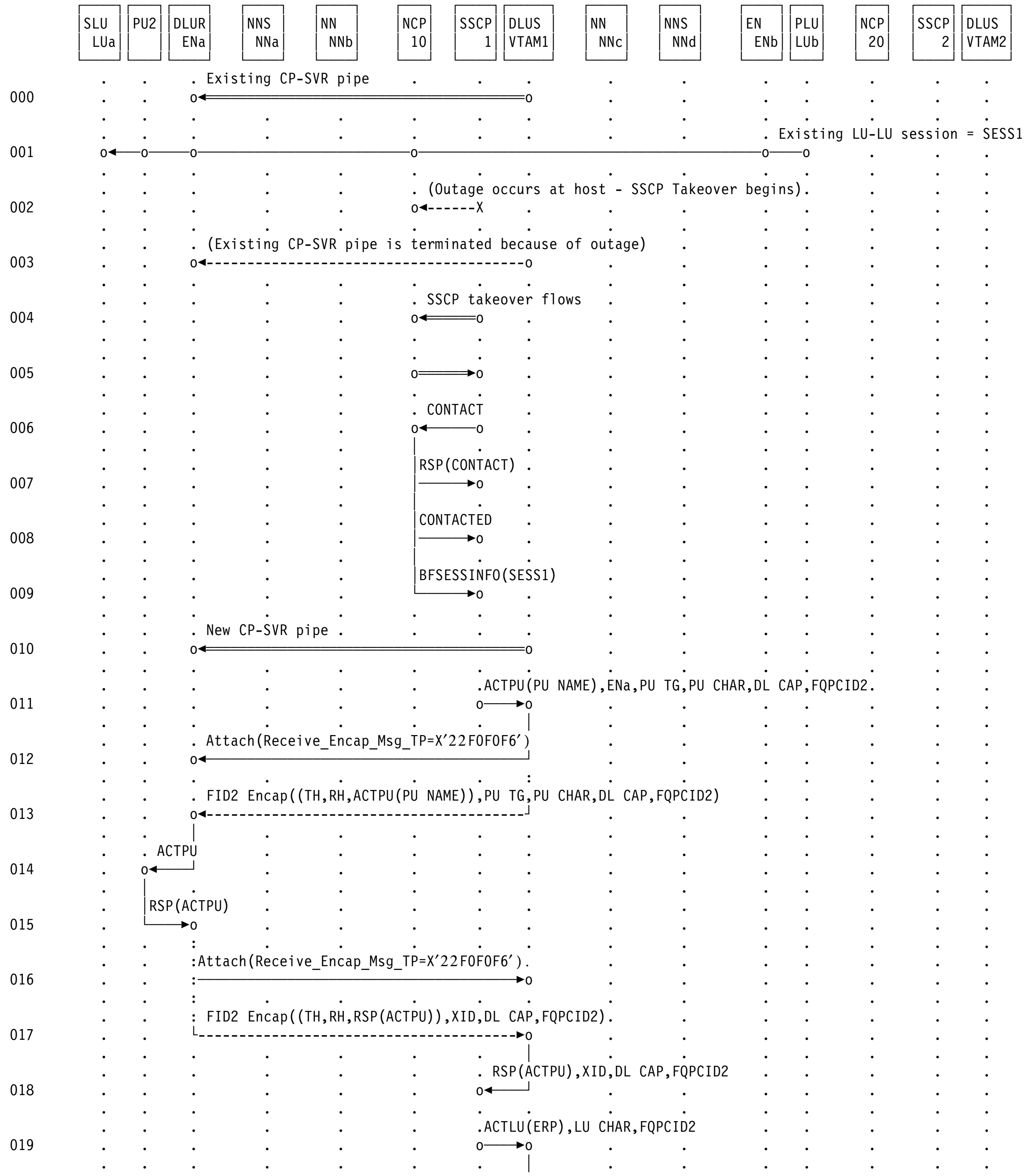
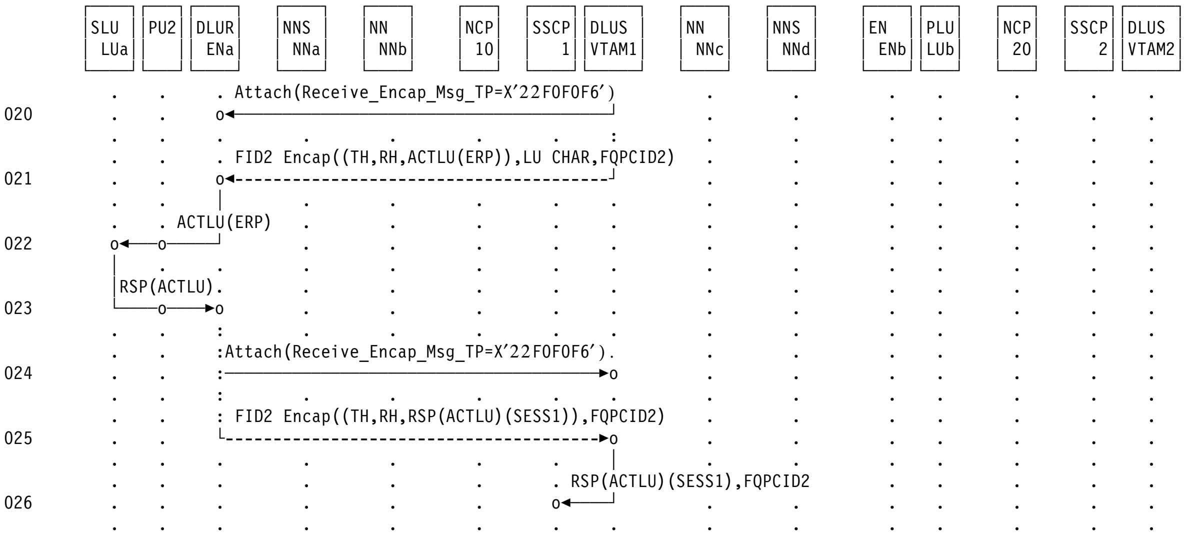


Figure 10-5. SSCP takeover - SSCP-LU, LU-LU sessions routed through same BF-NCP


SLU LUa
PU2
DLUR ENa
NNS NNa
NN NNb
NCP 10
SSCP 1
DLUS VTAM1
NN NNc
NNS NNd
EN ENb
PLU LUb
NCP 20
SSCP 2
DLUS VTAM2
020
Attach(Receive_Encap_Msg_TP=X'22F0F0F6')
021
FID2 Encap((TH,RH,ACTLU(ERP)),LU CHAR,FQPCID2)
022
ACTLU(ERP)
023
RSP(ACTLU)
024
Attach(Receive_Encap_Msg_TP=X'22F0F0F6')
025
FID2 Encap((TH,RH,RSP(ACTLU)(SESS1)),FQPCID2)
026
RSP(ACTLU)(SESS1),FQPCID2

1. The current LU-LU session between LUa and LUb routes through PU2, ENa, NNa, NNb, NCP10, NNc, NNd, and ENb.
2. The SSCP-NCP session between SSCP1 and NCP10 ends due to a host outage.
3. SSCP1 reacquires ownership of NCP10 and its attached resources.
4.
5.
6.
7.
8. When SSCP1 has acquired ownership of the link station through which the LU-LU session is routed, NCP10 builds and sends to SSCP1 a BFSESSINFO. This message indicates that a session exists between LUa (which it believes to be an independent LU) and LUb.
9. The existing CP-SVR pipe is taken down due to the loss of VTAM1.
10. Once VTAM1 is reactivated, a new CP-SVR pipe is established.
11. SSCP1 initiates restart of its session with PU2 by building an ACTPU and forwarding it to VTAM1.
12. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
13. The DLUS VTAM1 encapsulates the ACTPU and forwards it to the DLUR ENa.
14. ENa de-encapsulates the ACTPU and forwards it to PU2.
15. PU2 sends a RSP(ACTPU) to SSCP1 through ENa.
16. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
17. ENa encapsulates the RSP(ACTPU) and forwards it to VTAM1.
18. VTAM1 de-encapsulates the RSP(ACTPU) and forwards it on to SSCP1.
19. SSCP1 initiates restart of its session with LUa by building an ACTLU and forwarding it to VTAM1.
20. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
21. VTAM1 encapsulates the ACTLU and forwards it to ENa.
22. ENa de-encapsulates the ACTLU and forwards it to PU2 and on to LUa.
23. LUa sends a RSP(ACTLU) to SSCP1 through PU2, which forwards it to ENa.
24. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
25. ENa builds a Session Information (X′2A′) control vector, describing the LUa-LUb session, and appends the control vector to the RSP(ACTLU). ENa then encapsulates the RSP(ACTLU) and forwards it to VTAM1.
26. VTAM1 de-encapsulates the RSP(ACTLU) and forwards it on to SSCP1. SSCP1 correlates the RSP(ACTLU) with the BFSESSINFO by the FQPCID contained in each RU.

10.2.2.2 SSCP-LU, LU-LU Sessions Routed Through Same BF-NCP (Takeover and Giveback)

In Figure 10-6 on page 10-17, SSCP-PU and SSCP-LU sessions have been established from SSCP1 to PU2 and LUa, respectively, over the CP-SVR pipe between ENa and VTAM1. This pipe routes through the BF in NCP10, as does the LU-LU session between LUa and LUb. NCP10 is owned by SSCP1.

An outage occurs in the SSCP1/VTAM1 host, and a new pipe is established between ENa and VTAM2. SSCP2 receives all session awareness RUs concerning the active LU-LU session.

Later SSCP1 gets control back from SSCP2 and receives session awareness RUs about the LU-LU session.

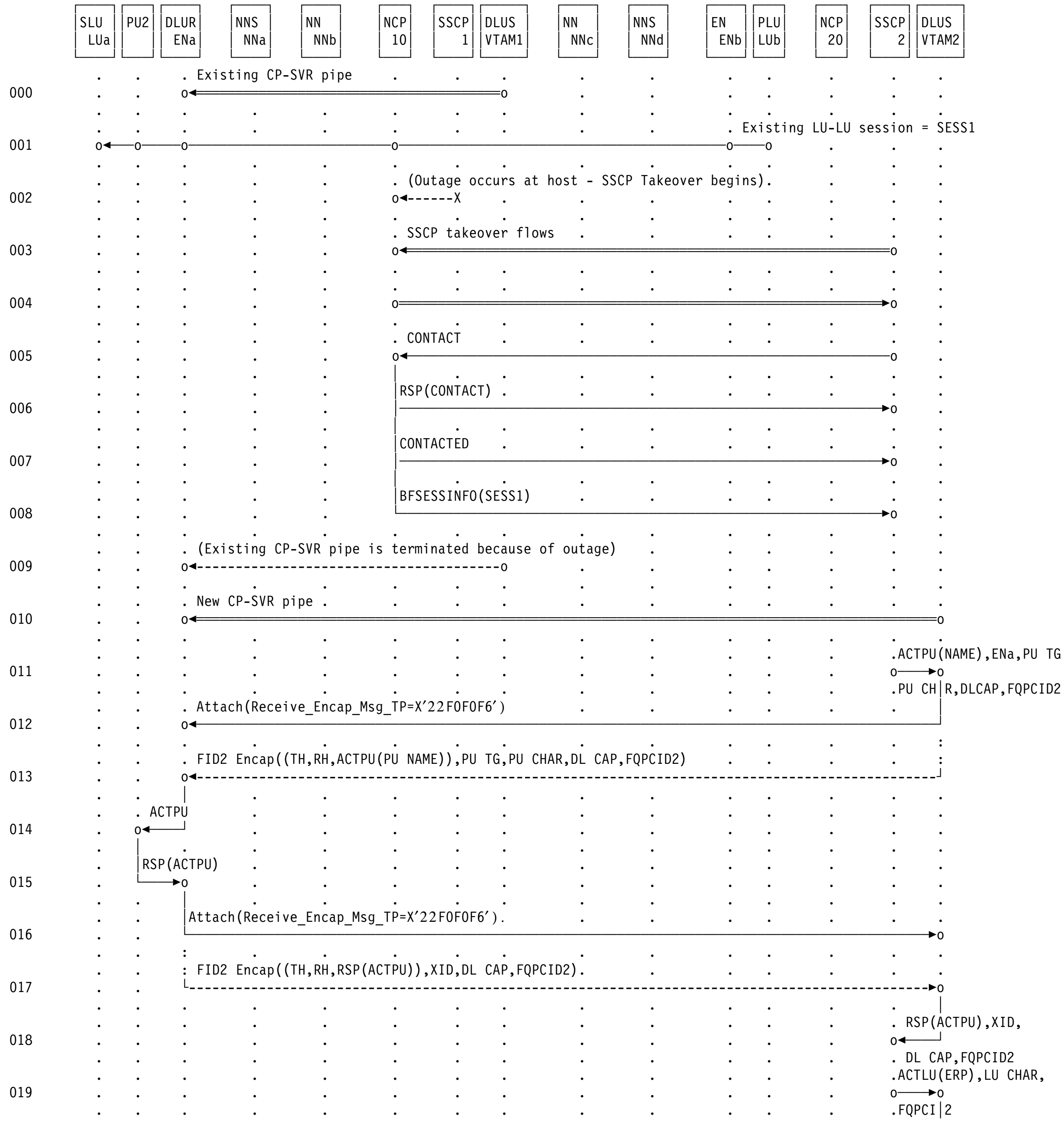


Figure 10-6. SSCP takeover & giveback - SSCP-LU, LU-LU sessions routed through same BF-NCP

```
       SLU  PU2  DLUR  NNS  NN   NCP  SSCP  DLUS   NN   NNS  EN   PLU  NCP  SSCP  DLUS
       LUa       ENa   NNa  NNb  10   1     VTAM1  NNc  NNd  ENb  LUb  20   2     VTAM2

           . Attach(Receive_Encap_Msg_TP=X'22F0F0F6')
020        o<----------------------------------------------------------------------------- DLUS VTAM2

           . FID2 Encap((TH,RH,ACTLU(ERP)),LU CHAR,FQPCID2)
021        o<- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - DLUS VTAM2

           . ACTLU(ERP)
022   o<---o<---- DLUR ENa

      RSP(ACTLU).
023   SLU LUa ---->o---->o DLUR ENa

           Attach(Receive_Encap_Msg_TP=X'22F0F0F6').
024        DLUR ENa -------------------------------------------------------------------------->o DLUS VTAM2

           : FID2 Encap((TH,RH,RSP(ACTLU)(SESS1)),FQPCID2)
025        DLUR ENa - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - ->o DLUS VTAM2

                                                                             . RSP(ACTLU)(SESS1),
026                                                           SSCP 2 o<---- DLUS VTAM2
                                                                             . FQPCID2

                                  .SSCP giveback flows
027                               SSCP 1 o=================================================>o SSCP 2

                              . CONTACT
028                     NCP 10 o<----o SSCP 1

                              RSP(CONTACT)
029                     NCP 10 ----->o SSCP 1

                              CONTACTED
030                     NCP 10 ----->o SSCP 1

                              BFSESSINFO(SESS1)
031                     NCP 10 ----->o SSCP 1

                                                                      .DACTPU(Giveback),
032                                                           SSCP 2 o---->o DLUS VTAM2
                                                                      .FQPCID2
           . Attach(Receive_Encap_Msg_TP=X'22F0F0F6')
033        o<----------------------------------------------------------------------------- DLUS VTAM2

           . FID2 Encap((TH,RH,DACTPU(Giveback)),FQPCID2)
034        o<- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - DLUS VTAM2

           :Attach(Receive_Encap_Msg_TP=X'22F0F0F6').
035        DLUR ENa -------------------------------------------------------------------------->o DLUS VTAM2

           : FID2 Encap((TH,RH,RSP(DACTPU)),FQPCID2).
036        DLUR ENa - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - ->o DLUS VTAM2

                                                                      .  RSP(DACTPU),
037                                                           SSCP 2 o<---- DLUS VTAM2
                                                                      . FQPCID2
           . UNBIND
038        o<----------------------------------------------------------------------------- DLUS VTAM2

           UNBIND
039        o<----------------------------------------------------------------------------- DLUS VTAM2

           Remaining CP-SVR pipe deactivation flows.
040        DLUR ENa ==========================================================================>o DLUS VTAM2
```

```
     SLU   PU2  DLUR    NNS     NN       NCP     SSCP  DLUS     NN      NNS      EN    PLU    NCP    SSCP  DLUS
      LUa        ENa     NNa     NNb      10        1  VTAM1     NNc     NNd      ENb   LUb     20       2  VTAM2

                 |Locate(Rq,Find(VTAM1),Cdinit(mode=CPSVRMGR))
041              └───────►o

                          |Locate(Rq,Find(VTAM1))
042                       └──────────────────────────────►o

                          . Locate(Rp,Found(VTAM1))       |
043                       o◄──────────────────────────────┘

                 . Locate(Rp,Found(VTAM2,Cdinit(RSCV))
044              o◄───────┘

                                                   . (Remaining CP-SVR pipe activation, REQACTPU, and ACTPU flows)
045         o◄══o═════════════════════════════════o════o

                                                   .ACTLU(ERP),LU CHAR,FQPCID3
046                                                o───►o

                 . Attach(Receive_Encap_Msg_TP=X'22F0F0F6')
047              o◄────────────────────────────────────┘

                 . FID2 Encap((TH,RH,ACTLU(ERP)),LU CHAR,FQPCID3)
048              o◄- - - - - - - - - - - - - - - - - - ┘
                 |
            . ACTLU(ERP)
049   o◄──o──────┘
      |
      |RSP(ACTLU).
050   └───o─────►o

                 :Attach(Receive_Encap_Msg_TP=X'22F0F0F6').
051              :─────────────────────────────────────►o
                 :
                 : FID2 Encap((TH,RH,RSP(ACTLU)(SESS1)),FQPCID3)
052              └- - - - - - - - - - - - - - - - - - -►o
                                                         |
                                                   . RSP(ACTLU)(SESS1),FQPCID3
053                                                o◄───┘
```

1. The current LU-LU session between LUa and LUb routes through PU2, ENa, NNa, NNb, NCP10, NNc, NNd, and ENb.
2. The SSCP-NCP session between SSCP1 and NCP10 ends due to a host outage.
3. SSCP2 acquires ownership of NCP10 and its attached resources.
4.
5.
6.
7.
8. When SSCP2 has acquired ownership of the link station through which the LU-LU session is routed, NCP10 builds and sends to SSCP2 a BFSESSINFO. This message indicates that a session exists between LUa (which it believes to be an independent LU) and LUb.
9. The existing CP-SVR pipe is taken down due to the loss of VTAM1.
10. A new CP-SVR pipe is established between ENa and VTAM2.
11. SSCP2 initiates restart of its session with PU2 by building an ACTPU and forwarding it to VTAM2.
12. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
13. The DLUS VTAM2 encapsulates the ACTPU and forwards it to the DLUR ENa.
14. ENa de-encapsulates the ACTPU and forwards it to PU2.
15. PU2 sends a RSP(ACTPU) to SSCP2 through ENa.
16. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
17. ENa encapsulates the RSP(ACTPU) and forwards it to VTAM2.
18. VTAM2 de-encapsulates the RSP(ACTPU) and forwards it on to SSCP2.
19. SSCP2 initiates restart of its session with LUa by building an ACTLU and forwarding it to VTAM2.
20. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
21. VTAM2 encapsulates the ACTLU and forwards it to ENa.
22. ENa de-encapsulates the ACTLU and forwards it to PU2 and on to LUa.
23. LUa sends a RSP(ACTLU) to SSCP2 through PU2, which forwards it to ENa.
24. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
25. ENa builds a Session Information (X′2A′) control vector, describing the LUa-LUb session, and appends the control vector to the RSP(ACTLU). ENA then encapsulates the RSP(ACTLU) and forwards it to VTAM2.
26. VTAM2 de-encapsulates the RSP(ACTLU) and forwards it on to SSCP2. SSCP2 correlates the RSP(ACTLU) with the BFSESSINFO by the FQPCID contained in each RU.
27. SSCP1 signals to SSCP2 that it is ready to reacquire ownership of the resources it had owned at the time of the outage. SSCP1 acquires ownership of NCP10 and its attached resources.
28.
29.
30.

31. When SSCP1 has acquired ownership of the link station through which the LU-LU session is routed, NCP10 builds and sends to SSCP1 a BFSESSINFO. This message indicates that a session exists between LUa (which it believes to be an independent LU) and LUb.

32. SSCP2 initiates deactivation of its session with PU2 by building a DACTPU(giveback) and forwarding it to VTAM2.

33. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.

34. The DLUS VTAM2 encapsulates the DACTPU and forwards it to the DLUR ENa.

35. ENa de-encapsulates the DACTPU and converts it to a RSP(DACTPU). The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.

36. The DLUR encapsulates the RSP(DACTPU) and forwards it to VTAM2.

37. VTAM2 de-encapsulates the RSP(DACTPU) and forwards it on to SSCP2.

38. VTAM2 determines that there are now no active or pending SSCP-PU sessions on its CP-SVR pipe to ENa, so it sends an UNBIND to ENa on VTAM2′s conwinner session to begin deactivation of the pipe.

39. VTAM2 also sends an UNBIND on its conloser session to ENa.

40. The DLUR node sends a RSP(UNBIND) to the DLUS for the conwinner session and another RSP(UNBIND) for the conloser session. For more details on CP-SVR pipe deactivation see 5.6, “CP-SVR Pipe Deactivation” on page 5-20.

41. ENa determines that an attached LU has an active LU-LU session, and it restarts activation of the CP-SVR pipe to VTAM1 (it will not attempt activation of the CP-SVR pipe to VTAM2, since it received a DACTPU(giveback) from VTAM2′s SSCP). ENa sends a locate for VTAM1.

42. The NNS(ENa) issues either a broadcast or directed search for the CP_LU of the DLUS=VTAM1.

43. The DLUS node responds to the search request with a Locate Found reply.

44. The NNS(ENa) uses the information in the Locate reply to calculate a route to the DLUS node and forwards this back to the DLUR in the form of an RSCV.

45. The CP-SVR pipe activation completes successfully; ENa sends a REQACTPU to SSCP1, and SSCP1 sends an ACTPU to restart its SSCP-PU session with PU2.

46. SSCP1 initiates restart of its session with LUa by building an ACTLU and forwarding it to VTAM1.

47. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.

48. VTAM1 encapsulates the ACTLU and forwards it to ENa.

49. ENa de-encapsulates the ACTLU and forwards it to PU2 and on to LUa.

50. LUa sends a RSP(ACTLU) to SSCP1 through PU2, which forwards it to ENa.

51. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.

52. ENa builds a Session Information (X′2A′) control vector, describing the LUa-LUb session, and appends the control vector to the RSP(ACTLU). ENa then encapsulates the RSP(ACTLU) and forwards it to VTAM1.

53. VTAM1 de-encapsulates the RSP(ACTLU) and forwards it on to SSCP1. SSCP1 correlates the RSP(ACTLU) with the BFSESSINFO by the FQPCID contained in each RU.

10.2.2.3 SSCP-LU, LU-LU Sessions Routed Through Different BF-NCPs (Same Takeover SSCP)

In Figure 10-7 on page 10-23, SSCP-PU and SSCP-LU sessions have been established from SSCP1 to PU2 and LUa, respectively, over the CP-SVR pipe between ENa and VTAM1. This pipe routes through the BF in NCP10, which is owned by SSCP1. The LU-LU session between LUa and LUb routes through the BF in NCP20, which is also owned by SSCP1.

An outage occurs in the SSCP1/VTAM1 host, and SSCP2 takes over ownership of the resources previously owned by SSCP1. This includes taking over ownership of both NCP10 and NCP20 as well as establishing a new pipe between ENa and VTAM2. SSCP2 receives all session awareness RUs concerning the active LU-LU session.

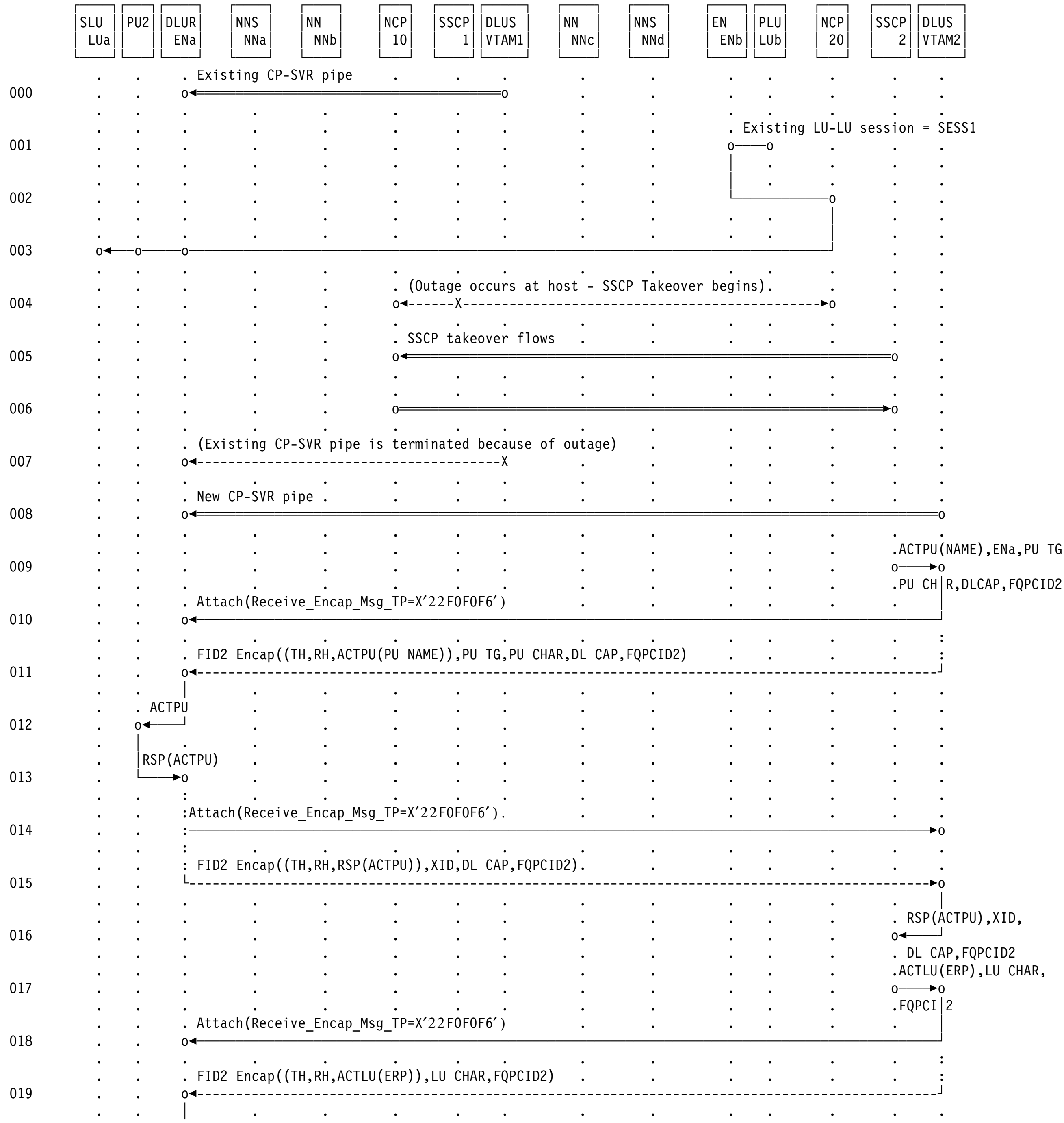
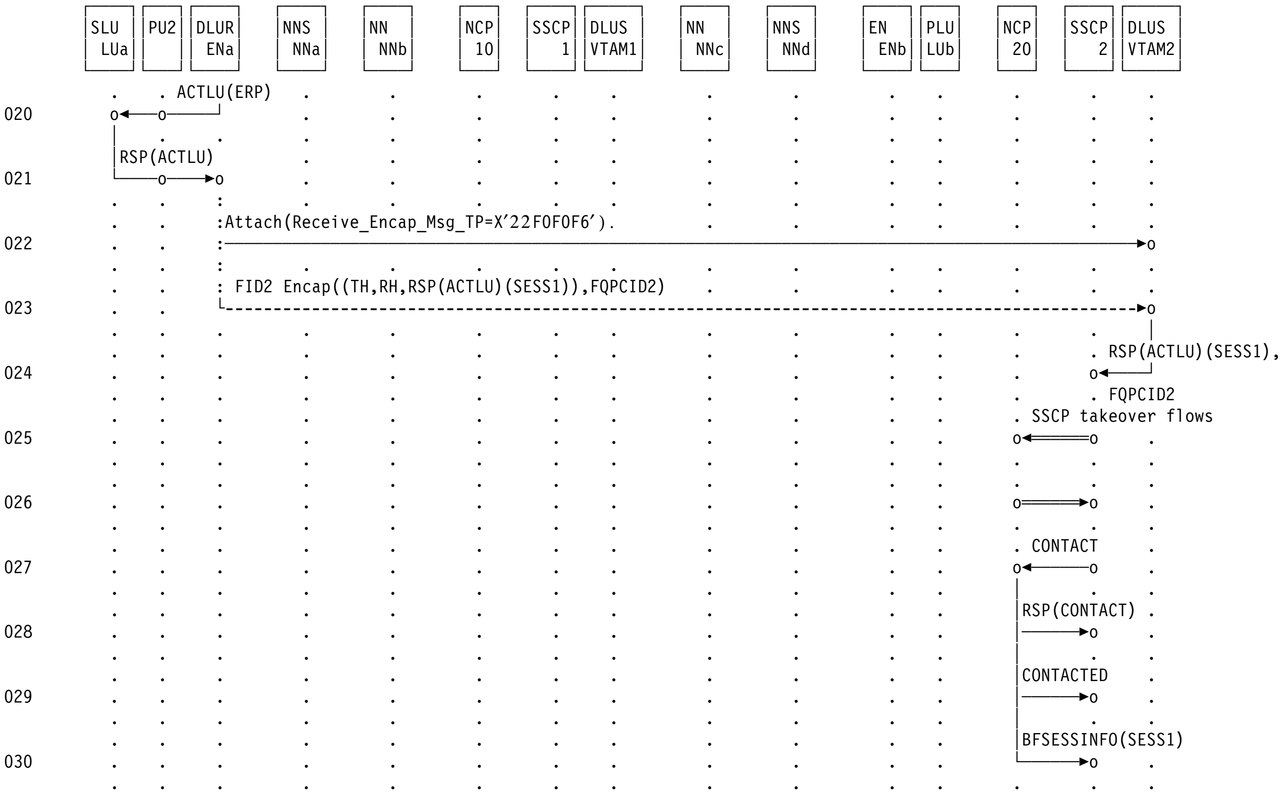


Figure 10-7. SSCP takeover - SSCP-LU, LU-LU sessions routed through different BF-NCPs / same takeover SSCP


SLU LUa
PU2
DLUR ENa
NNS NNa
NN NNb
NCP 10
SSCP 1
DLUS VTAM1
NN NNc
NNS NNd
EN ENb
PLU LUb
NCP 20
SSCP 2
DLUS VTAM2
020
ACTLU(ERP)
021
RSP(ACTLU)
022
:Attach(Receive_Encap_Msg_TP=X'22F0F0F6').
023
FID2 Encap((TH,RH,RSP(ACTLU)(SESS1)),FQPCID2)
024
RSP(ACTLU)(SESS1),
FQPCID2
025
SSCP takeover flows
026
027
CONTACT
028
RSP(CONTACT)
029
CONTACTED
030
BFSESSINFO(SESS1)

1. The current LU-LU session between LUa and LUb routes through PU2, ENa, NNa, NNb, NCP20, NNc, NNd, and ENb.
2.
3.
4. The SSCP-NCP sessions between SSCP1 and NCP10 and between SSCP1 and NCP20 end due to a host outage.
5. SSCP2 acquires ownership of NCP10 and its attached resources previously owned by SSCP1.
6.
7. The existing CP-SVR pipe is taken down due to the loss of VTAM1.
8. A new CP-SVR pipe is established between ENa and VTAM2.
9. SSCP2 initiates normal activation of its session with PU2 by building an ACTPU and forwarding it to VTAM2.
10. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
11. The DLUS VTAM2 encapsulates the ACTPU and forwards it to the DLUR ENa.
12. ENa de-encapsulates the ACTPU and forwards it to PU2.
13. PU2 sends a RSP(ACTPU) to SSCP2 through ENa.
14. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
15. ENa encapsulates the RSP(ACTPU) and forwards it to VTAM2.
16. VTAM2 de-encapsulates the RSP(ACTPU) and forwards it on to SSCP2.
17. SSCP2 initiates normal activation of its session with LUa by building an ACTLU and forwarding it to VTAM2.
18. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
19. VTAM2 encapsulates the ACTLU and forwards it to ENa.
20. ENa de-encapsulates the ACTLU and forwards it to PU2 and on to LUa.
21. LUa sends a RSP(ACTLU) to SSCP2 through PU2, which forwards it to ENa.
22. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
23. ENa builds a Session Information (X′2A′) control vector, describing the LUa-LUb session, and appends the control vector to the RSP(ACTLU). ENa then encapsulates the RSP(ACTLU) and forwards it to VTAM2.
24. VTAM2 de-encapsulates the RSP(ACTLU) and forwards it on to SSCP2.
25. SSCP2 acquires ownership of NCP20 and its attached resources.
26.
27.
28.
29.
30. When SSCP2 has acquired ownership of the link station through which the LU-LU session is routed, NCP20 builds and sends to SSCP2 a BFSESSINFO. This message indicates that a session exists between LUa (which it believes to be an independent LU) and LUb. SSCP2 correlates the BFSESSINFO with the RSP(ACTLU) by the FQPCID contained in each RU.

10.2.2.4 SSCP-LU, LU-LU Sessions Routed Through Different BF-NCPs (Different Takeover SSCPs)

In Figure 10-8 on page 10-27, SSCP-PU and SSCP-LU sessions have been established from SSCP1 to PU2 and LUa, respectively, over the CP-SVR pipe between ENa and VTAM1. This pipe routes through the BF in NCP10, which is owned by SSCP1. The LU-LU session between LUa and LUb routes through the BF in NCP20, which is also owned by SSCP1.

An outage occurs in the SSCP1/VTAM1 host, and SSCP2 and SSCP3 take over ownership of the resources previously owned by SSCP1. This includes SSCP2 taking over ownership of NCP10, SSCP3 taking over ownership of NCP20, and ENa and VTAM2 establishing a new CP-SVR pipe. The dependent LU session awareness RUs concerning the active LU-LU session will be sent to SSCP2, while the session′s independent LU session awareness RUs will be sent to SSCP3.

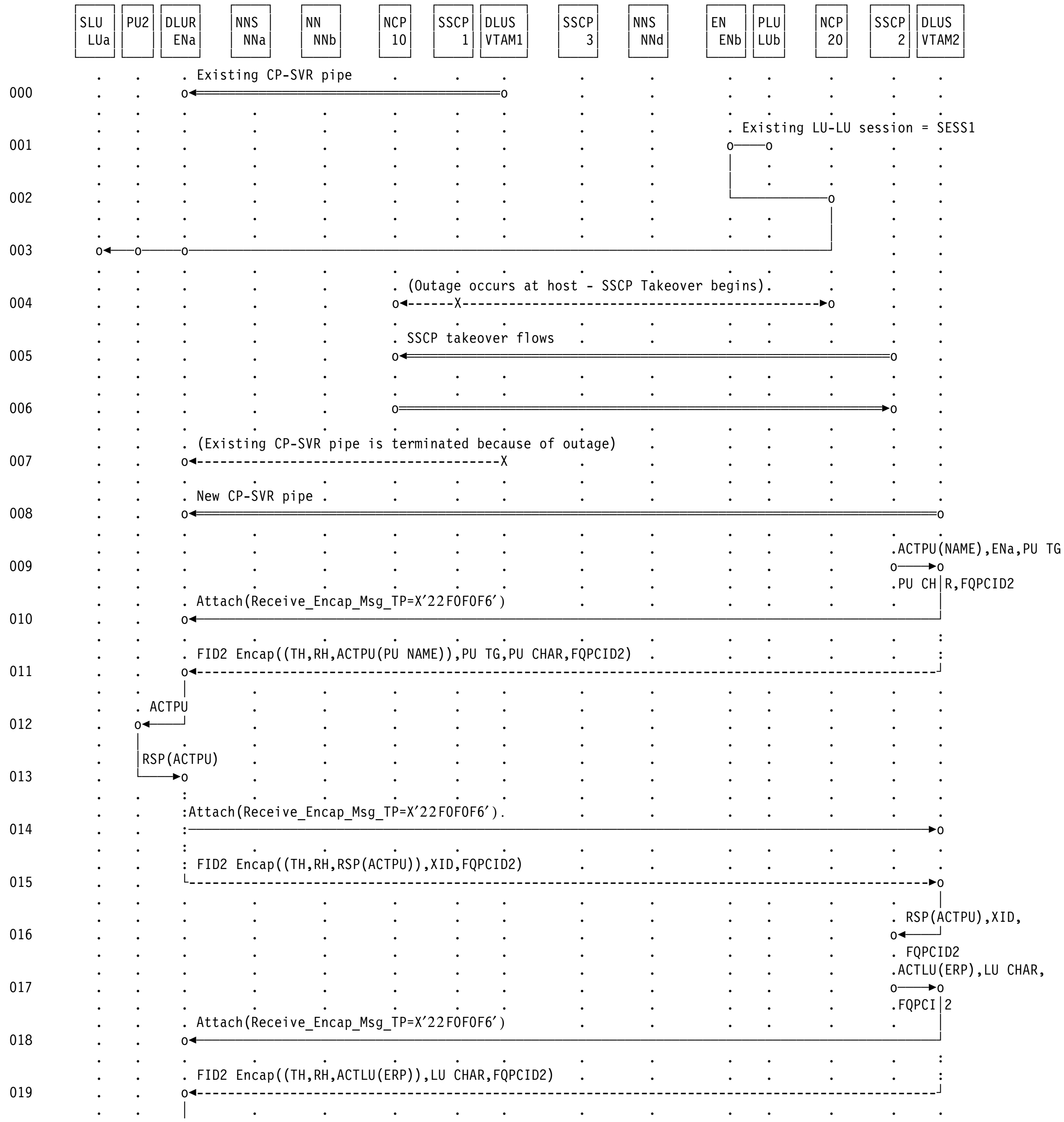
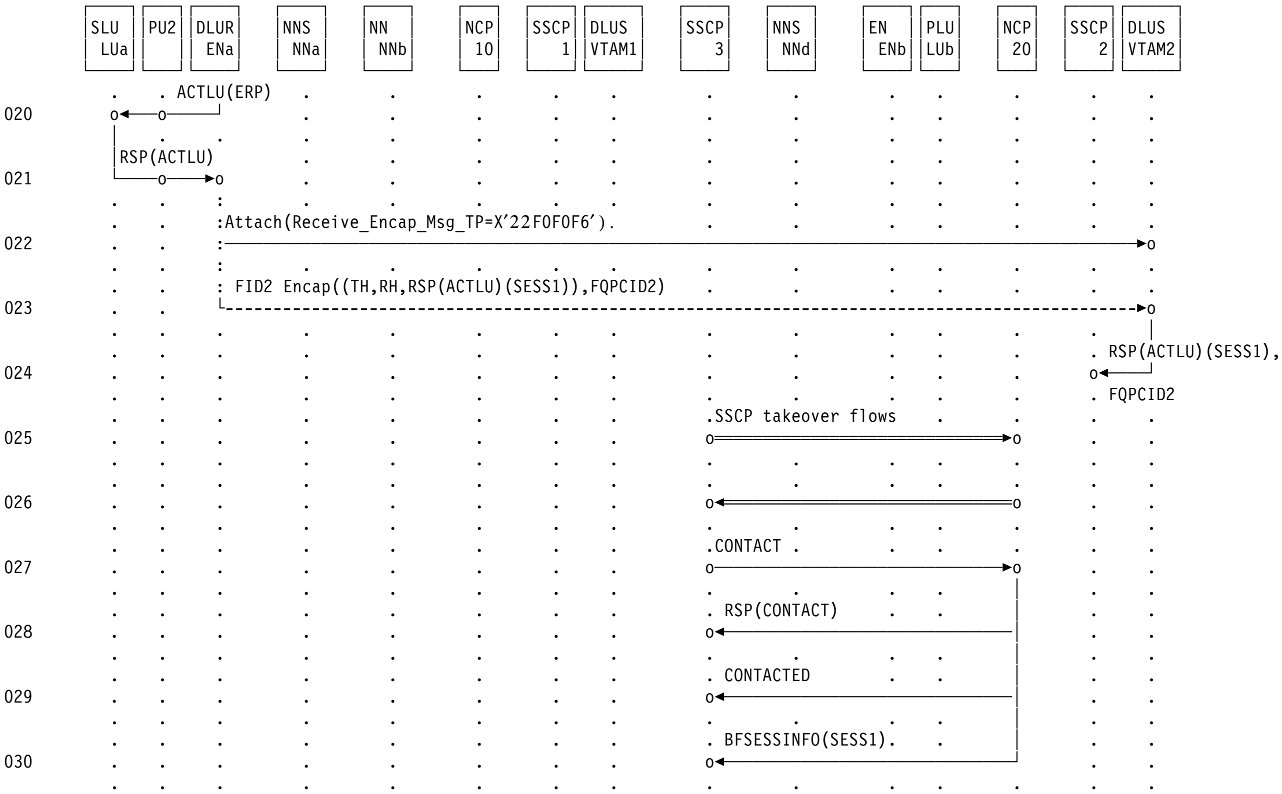


Figure 10-8. SSCP takeover - SSCP-LU, LU-LU sessions routed through different BF-NCPs / different takeover SSCPs


SLU LUa
PU2
DLUR ENa
NNS NNa
NN NNb
NCP 10
SSCP 1
DLUS VTAM1
SSCP 3
NNS NNd
EN ENb
PLU LUb
NCP 20
SSCP 2
DLUS VTAM2
020
ACTLU(ERP)
021
RSP(ACTLU)
022
Attach(Receive_Encap_Msg_TP=X'22F0F0F6').
023
FID2 Encap((TH,RH,RSP(ACTLU)(SESS1)),FQPCID2)
024
RSP(ACTLU)(SESS1),
FQPCID2
025
SSCP takeover flows
026
027
CONTACT
028
RSP(CONTACT)
029
CONTACTED
030
BFSESSINFO(SESS1).

1. The current LU-LU session between LUa and LUb routes through PU2, ENa, NNa, NNb, NCP20, NNc, NNd, and ENb.
2.
3.
4. The SSCP-NCP sessions between SSCP1 and NCP10 and between SSCP1 and NCP20 end due to a host outage.
5. SSCP2 acquires ownership of NCP10 and its attached resources previously owned by SSCP1.
6.
7. The existing CP-SVR pipe is taken down due to the loss of VTAM1.
8. A new CP-SVR pipe is established between ENa and VTAM2.
9. SSCP2 initiates normal activation of its session with PU2 by building an ACTPU and forwarding it to VTAM2.
10. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
11. The DLUS VTAM2 encapsulates the ACTPU and forwards it to the DLUR ENa.
12. ENa de-encapsulates the ACTPU and forwards it to PU2.
13. PU2 sends a RSP(ACTPU) to SSCP2 through ENa.
14. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
15. ENa encapsulates the RSP(ACTPU) and forwards it to VTAM2.
16. VTAM2 de-encapsulates the RSP(ACTPU) and forwards it on to SSCP2.
17. SSCP2 initiates normal activation of its session with LUa by building an ACTLU and forwarding it to VTAM2.
18. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
19. VTAM2 encapsulates the ACTLU and forwards it to ENa.
20. ENa de-encapsulates the ACTLU and forwards it to PU2 and on to LUa.
21. LUa sends a RSP(ACTLU) to SSCP2 through PU2, which forwards it to ENa.
22. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
23. ENa builds a Session Information (X′2A′) control vector, describing the LUa-LUb session, and appends the control vector to the RSP(ACTLU). ENa then encapsulates the RSP(ACTLU) and forwards it to VTAM2.
24. VTAM2 de-encapsulates the RSP(ACTLU) and forwards it on to SSCP2.
25. SSCP3 acquires ownership of NCP20 and its attached resources.
26.
27.
28.
29.
30. When SSCP3 has acquired ownership of the link station through which the LU-LU session is routed, NCP20 builds and sends to SSCP3 a BFSESSINFO. This message indicates that a session exists between LUa (which it believes to be an independent LU) and LUb.

10.2.3 Normal LU-LU Session Deactivation Flows

For LU-LU session deactivation, there will be the following different dependent LU session awareness flow scenarios in the DLUS environment:

1. the dependent LU′s SSCP-LU and LU-LU sessions route through the same BF-NCP
   - the SESSEND and BFSESSEND are received by the DLUS′s SSCP
2. the dependent LU′s SSCP-LU and LU-LU sessions route through different BF-NCPs owned by the same SSCP (the DLUS′s SSCP)
   - the SESSEND and BFSESSEND are received by the DLUS′s SSCP
3. the dependent LU′s SSCP-LU and LU-LU sessions route through different BF-NCPs owned by different SSCPs
   - only the SESSEND is received by the DLUS′s SSCP; the BFSESSEND is received by a different SSCP
4. the dependent LU′s LU-LU session does not route through a BF-NCP
   - no BFSESSEND is built; only the SESSEND is received by the DLUS′s SSCP

The DLUR will build a new format of SESSEND, Format 3; it differs from the current Format 2 in that it does not carry all of the same control vectors (see 13.3, “Dependent LU Session Awareness Format Changes” on page 13-7 for the new SESSEND format).

10.2.3.1 SSCP-LU, LU-LU Sessions Routed Through Same BF-NCP

In Figure 10-9, SSCP-PU and SSCP-LU sessions have been established from SSCP1 to PU2 and LUa, respectively, over the CP-SVR pipe between ENa and VTAM1. This pipe routes through the BF in NCP10, as does the LU-LU session between LUa and LUb. NCP10 is owned by SSCP1, which will receive all of the session awareness RUs concerning LUa .

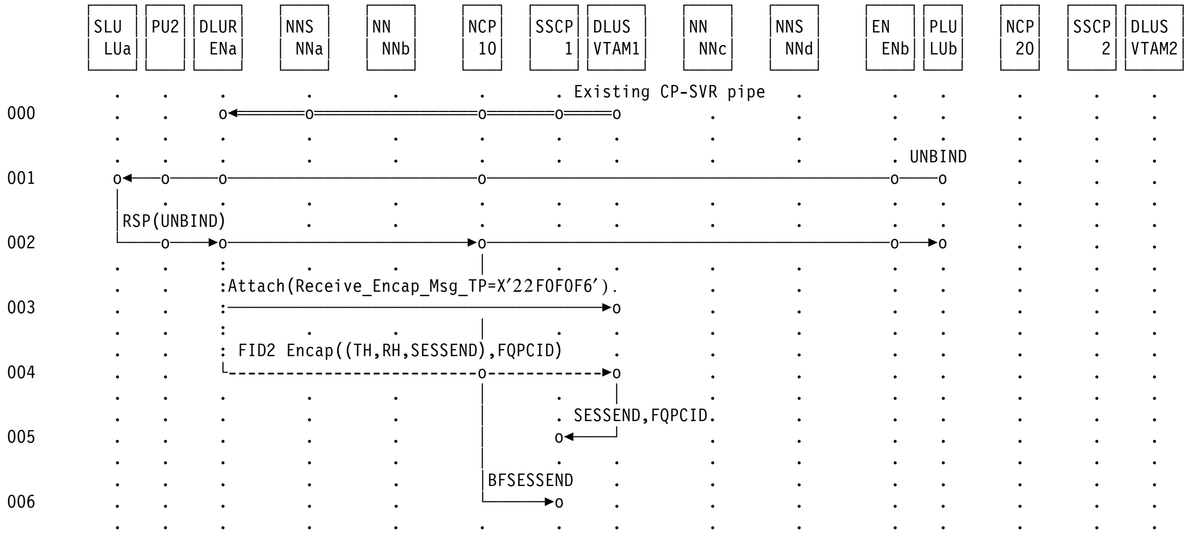


Figure 10-9. LU-LU session deactivation - SSCP-LU, LU-LU sessions routed through same BF-NCP

1. UNBIND is sent by LUb to LUa, routing through ENb, NNd, NNc, NCP10, NNb, NNa, ENa, and PU2.
2. RSP(UNBIND) is sent by LUa to LUb over the same route.
3. Upon receipt of RSP(UNBIND), the DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
4. The DLUR in ENa creates a Format 3 SESSEND and encapsulates it, sending it on the CP-SVR pipe to the DLUS in VTAM1.
5. DLUS de-encapsulates the SESSEND and forwards it to SSCP1.
6. Upon receipt of RSP(UNBIND), NCP10 creates a BFSESSEND and sends it to SSCP1. SSCP1 correlates the BFSESSEND with the SESSEND by the FQPCID contained in each RU.

10.2.3.2 SSCP-LU, LU-LU Sessions Routed Through Different BF-NCPs (Same SSCP)

In Figure 10-10, SSCP-PU and SSCP-LU sessions have been established from SSCP1 to PU2 and LUa, respectively, over the CP-SVR pipe between ENa and VTAM1. This pipe routes through the BF in NCP10. The LU-LU session between LUa and LUb routes, however, through NCP20. Both NCPs are owned by SSCP1, so all session awareness RUs concerning LUa will be sent to SSCP1.

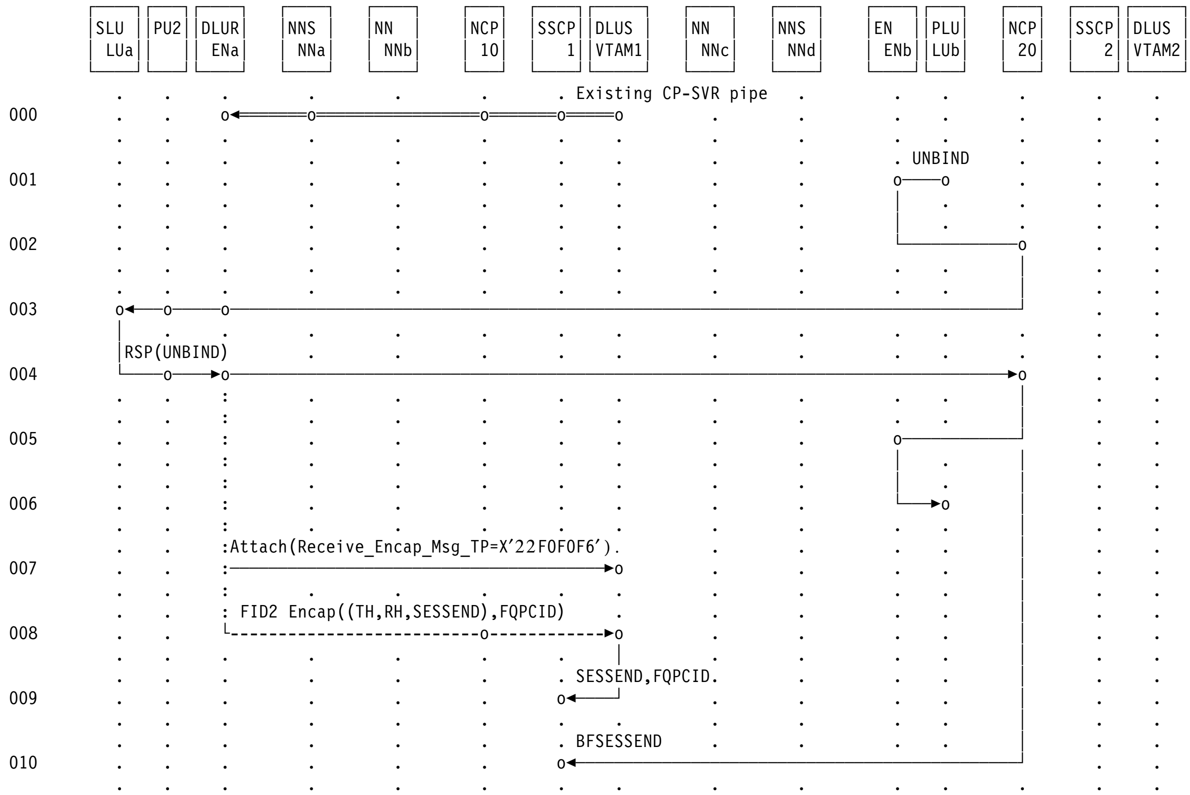


Figure 10-10. LU-LU session deactivation - SSCP-LU, LU-LU sessions routed through different BF-NCPs (same SSCP)

1. UNBIND is sent by LUb to LUa, routing through ENb, NNd, NNc, NCP20, NNb, NNa, ENa, and PU2.
2.
3.
4. RSP(UNBIND) is sent by LUa to LUb over the same route.
5.
6.
7. Upon receipt of RSP(UNBIND), the DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
8. The DLUR in ENa creates a Format 3 SESSEND and encapsulates it, sending it on the CP-SVR pipe to the DLUS in VTAM1.
9. DLUS de-encapsulates the SESSEND and forwards it to SSCP1.
10. Upon receipt of RSP(UNBIND), NCP20 creates a BFSESSEND and sends it to SSCP1. SSCP1 correlates the BFSESSEND with the SESSEND by the FQPCID contained in each RU.

10.2.3.3 SSCP-LU, LU-LU Sessions Routed Through Different BF-NCPs (Different SSCPs)

In Figure 10-11, SSCP-PU and SSCP-LU sessions have been established from SSCP1 to PU2 and LUa, respectively, over the CP-SVR pipe between ENa and VTAM1. This pipe routes through the BF in NCP10. The LU-LU session between LUa and LUb routes, however, through NCP20. NCP10 is owned by SSCP1, and NCP20 is owned by SSCP2. Therefore, the dependent LU session awareness RUs concerning LUa will be sent to SSCP1, while LUa's independent LU session awareness RUs will be sent to SSCP2.

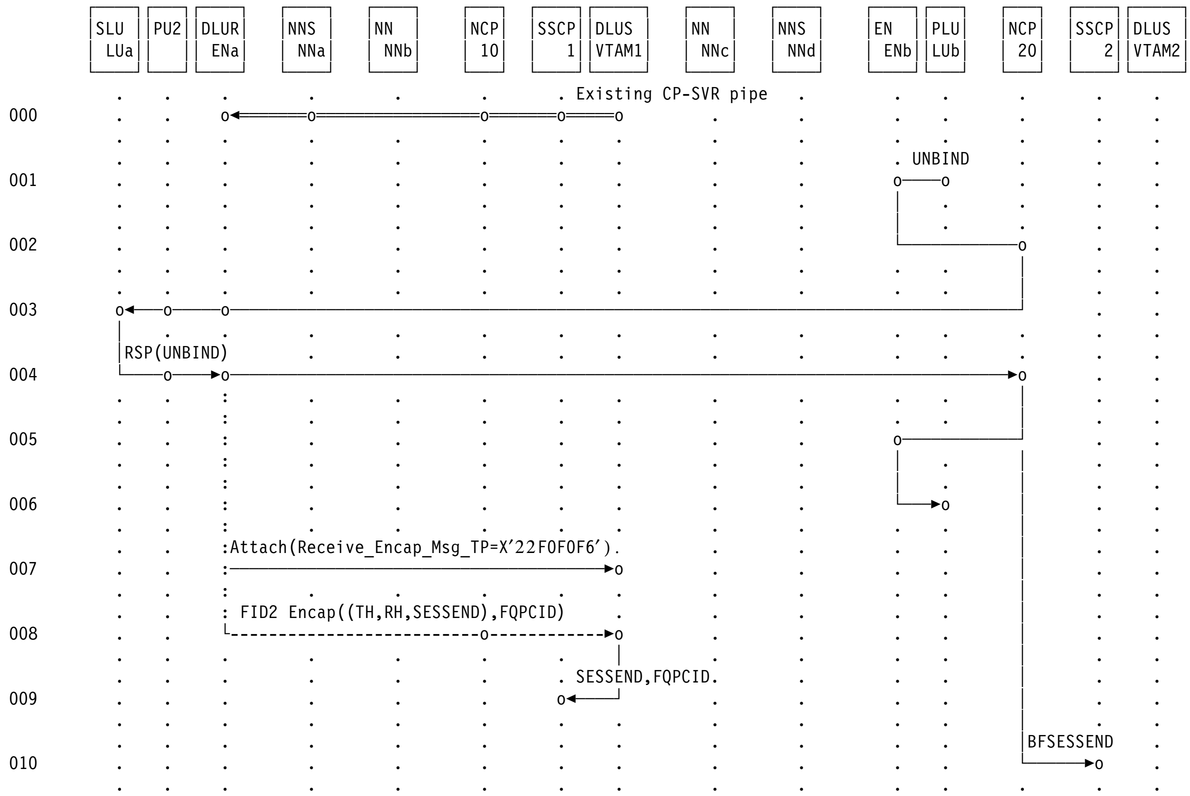


Figure 10-11. LU-LU session deactivation - SSCP-LU, LU-LU sessions routed through different BF-NCPs (different SSCPs)

1. UNBIND is sent by LUb to LUa, routing through ENb, NNd, NNc, NCP20, NNb, NNa, ENa, and PU2.
2.
3.
4. RSP(UNBIND) is sent by LUa to LUb over the same route.
5.
6.
7. Upon receipt of RSP(UNBIND), the DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
8. The DLUR in ENa creates a Format 3 SESSEND and encapsulates it, sending it on the CP-SVR to the DLUS in VTAM1.
9. DLUS de-encapsulates the SESSEND and forwards it to SSCP1.
10. Upon receipt of RSP(UNBIND), NCP20 creates a BFSESSEND and sends it to SSCP2.

10.2.3.4 LU-LU Session Not Routed Through A BF-NCP

In Figure 10-12, SSCP-PU and SSCP-LU sessions have been established from SSCP1 to PU2 and LUa, respectively, over the CP-SVR pipe between ENa and VTAM1. This pipe routes through the BF in NCP10. The LU-LU session between LUa and LUb does not route, however, through any NCP. Therefore, SSCP1 will only receive dependent LU session awareness RUs concerning LUa.

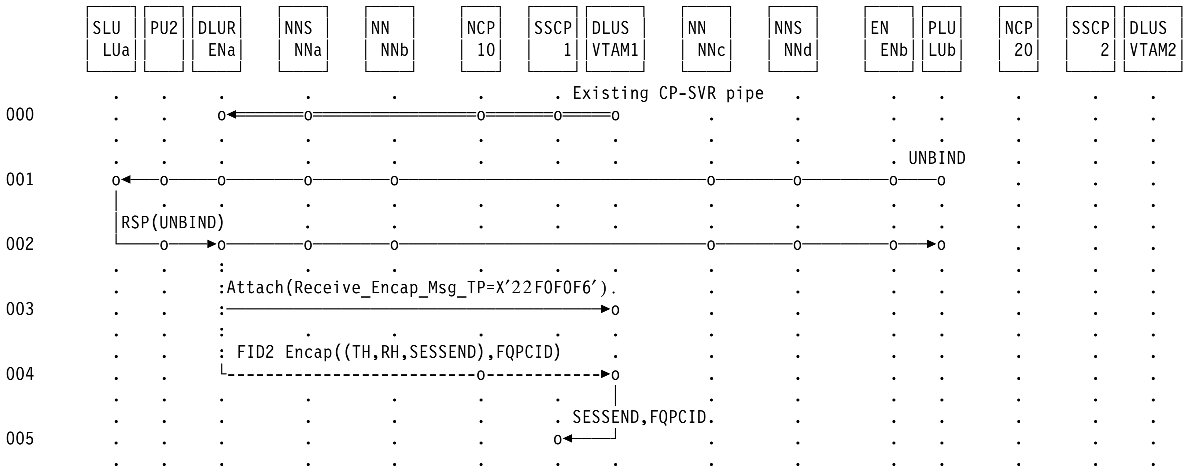


Figure 10-12. LU-LU session deactivation - LU-LU session not routed through a BF-NCP

1. UNBIND is sent by LUb to LUa, routing through ENb, NNd, NNc, NNb, NNa, ENa, and PU2.
2. RSP(UNBIND) is sent by LUa to LUb over the same route.
3. Upon receipt of RSP(UNBIND), the DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
4. The DLUR in ENa creates a Format 3 SESSEND and encapsulates it, sending it on the CP-SVR pipe to the DLUS in VTAM1.
5. DLUS de-encapsulates the SESSEND and forwards it to SSCP1.

10.2.4 Abnormal LU-LU Session Deactivation Flows

The same session deactivation flow scenarios apply to abnormal deactivation situations. The new Format 3 SESSEND will also be used in these cases.

10.2.4.1 Abnormal LU-LU Session Deactivation

In Figure 10-13 on page 10-38, SSCP-PU and SSCP-LU sessions have been established from SSCP1 to PU2 and LUa, respectively, over the CP-SVR pipe between ENa and VTAM1. This pipe routes through the BF in NCP10, as does the LU-LU session between LUa and LUb. NCP10 is owned by SSCP1, which will receive all of the session awareness RUs concerning LUa.

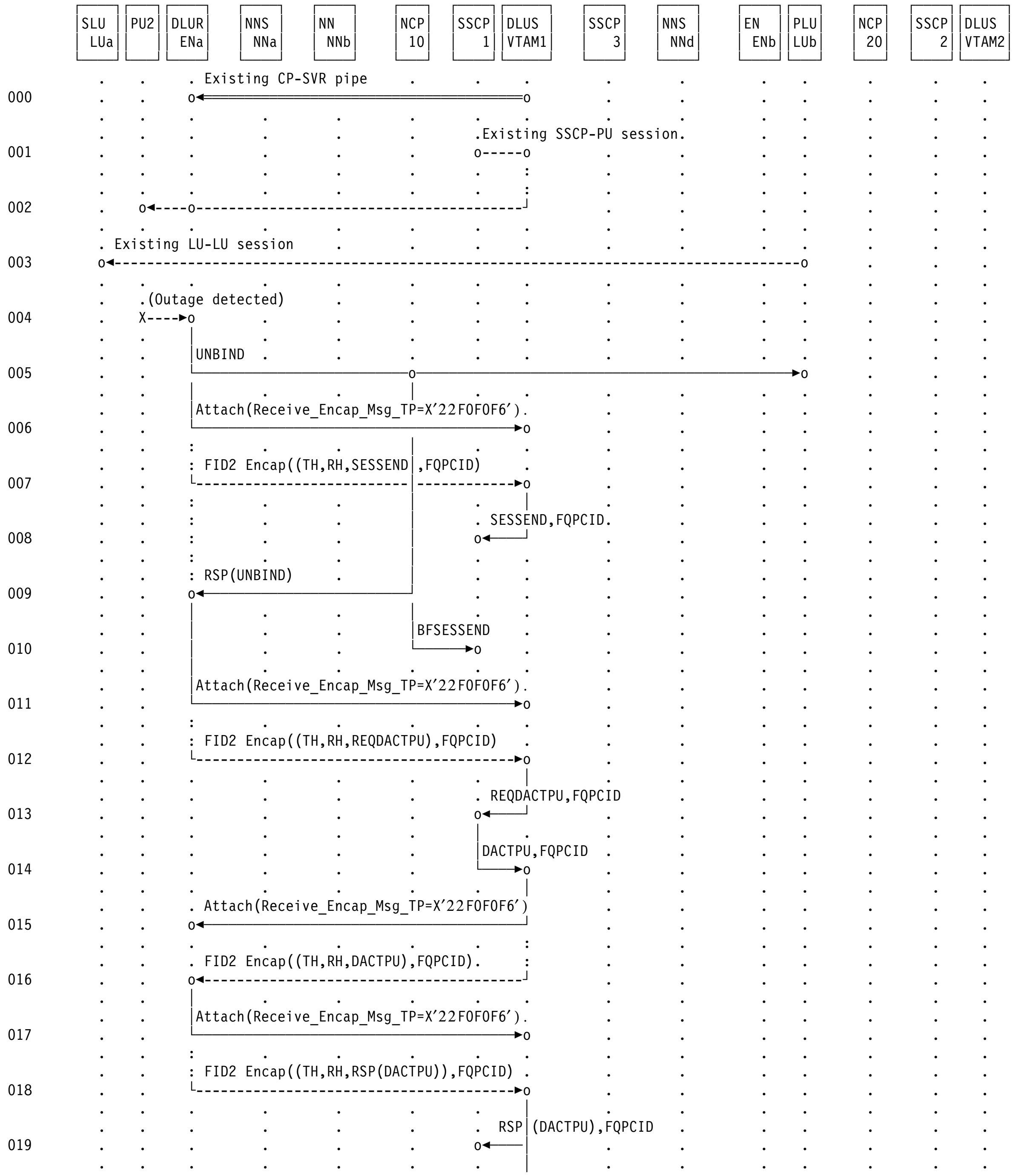


Figure 10-13. LU-LU session deactivation (abnormal)

```
        ┌─────┐ ┌───┐ ┌────┐   ┌────┐   ┌────┐    ┌───┐  ┌────┐ ┌─────┐   ┌────┐   ┌────┐    ┌────┐ ┌───┐   ┌───┐  ┌────┐ ┌─────┐
        │SLU  │ │PU2│ │DLUR│   │NNS │   │NN  │    │NCP│  │SSCP│ │DLUS │   │SSCP│   │NNS │    │EN  │ │PLU│   │NCP│  │SSCP│ │DLUS │
        │ LUa │ │   │ │ ENa│   │ NNa│   │ NNb│    │ 10│  │   1│ │VTAM1│   │   3│   │ NNd│    │ ENb│ │LUb│   │ 20│  │   2│ │VTAM2│
        └─────┘ └───┘ └────┘   └────┘   └────┘    └───┘  └────┘ └─────┘   └────┘   └────┘    └────┘ └───┘   └───┘  └────┘ └─────┘
           .      .     . UNBIND .         .         .       .      │         .        .          .     .       .       .      .
020        .      .     o◄──────────────────────────────────────────┘         .        .          .     .       .       .      .
           .      .     ║        .         .         .       .      │         .        .          .     .       .       .      .
           .      .     ║ UNBIND .         .         .       .      │         .        .          .     .       .       .      .
021        .      .     o◄──────────────────────────────────────────┘         .        .          .     .       .       .      .
           .      .     ║        .         .         .       .      .         .        .          .     .       .       .      .
           .      .     ║Remaining CP-SVR pipe deactivation flows.            .        .          .     .       .       .      .
022        .      .     ╚══════════════════════════════════════════►o         .        .          .     .       .       .      .
           .      .     .        .         .         .       .      .         .        .          .     .       .       .      .
```

1. There is an active session between SSCP1 and PU2. It is the only active SSCP-PU session using the CP-SVR pipe between VTAM1 and ENa.
2.
3. There is an active session between LUa and LUb.
4. ENa detects an outage on its connection to PU2.
5. UNBIND is sent by the DLUR in ENa to LUb, routing through NNa, NNb, NCP10, NNd, and ENb. The UNBIND type is X′08′ (route extension inoperative).
6. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
7. The DLUR in ENa also creates a Format 3 SESSEND and encapsulates it, sending it on the CP-SVR pipe to the DLUS in VTAM1.
8. DLUS de-encapsulates the SESSEND and forwards it to SSCP1.
9. Upon receipt of UNBIND, NCP10 creates a RSP(UNBIND) and sends it to ENa while forwarding the UNBIND on to LUb.
10. NCP10 also creates a BFSESSEND and sends it to SSCP1. SSCP1 correlates the BFSESSEND with the SESSEND by the FQPCID contained in each RU.
11. The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
12. To signal to SSCP1 that its SSCP-PU session with PU2 is broken, ENa builds a REQDACTPU, encapsulates it, and forwards it to VTAM1 on its CPSVRMGR pipe.
13. VTAM1 de-encapsulates the REQDACTPU and forwards it to SSCP1.
14. SSCP1 responds to the REQDACTPU by building a DACTPU and passing it on to VTAM1.
15. The DLUS node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUR node.
16. VTAM1 encapsulates the DACTPU and forwards it to ENa.
17. ENa de-encapsulates the DACTPU, and since it no longer has a connection to PU2, converts the DACTPU into a positive RSP(DACTPU). The DLUR node sends an FMH5 Attach to initiate the Receive_Encap_Msg_TP at the DLUS node.
18. ENa then encapsulates the RSP(DACTPU) and sends it to VTAM1.
19. VTAM1 de-encapsulates the RSP(DACTPU) and forwards it to SSCP1.
20. VTAM1 also determines that there are now no active or pending SSCP-PU sessions on its CP-SVR pipe to ENa, so it sends an UNBIND to ENa on VTAM1′s conwinner session to begin deactivation of the pipe.
21. VTAM1 also sends an UNBIND on its conloser session to ENa.
22. The DLUR node sends a RSP(UNBIND) to the DLUS for the conwinner session and another RSP(UNBIND) for the conloser session. For more details on CP-SVR pipe deactivation see 5.6, “CP-SVR Pipe Deactivation” on page 5-20.

10.3 DLUR HPR Session Awareness

Down to this point, the discussion in Chapter 10, "Dependent LU Session Awareness Management" on page 10-1 has dealt with the various SAW-related functions that all DLUR implementations must support. In this section we describe an optional extension to a DLUR′s session awareness capabilities, called DLUR HPR Session Awareness (DLUR HPR SAW). Since DLUR HPR SAW support is optional, a DLUR is free not to support it. If a DLUR does support it, however, it must support it in its entirety.

DLUR HPR SAW is designed to solve a very specific problem, in a very specific configuration. If a DLUR node supports APPN HPR′s RTP Endpoint function, and if some of its dependent LUs have sessions that use an RTP connection that has an endpoint in the DLUR node, then the session path information that the DLUR provided at session establishment or at SSCP takeover may become incorrect because of an HPR path switch. The DLUR HPR SAW function gives the DLUR a way to provide the DLUS with an updated path for an RTP connection when the connection performs a path switch.

Such an update is possible only because the DLUR node is also the RTP endpoint for the RTP connection. If a dependent LU has a session that uses an RTP connection which doesn′t end in the DLUR node, then the DLUR has no way of knowing when an HPR path switch occurs, and no way of knowing the new path that results from the path switch. Thus the DLUR HPR SAW function applies only to RTP connections that end in the DLUR node.

The DLUR HPR SAW function itself is very straightforward. When the DLUR provides its initial SAW data for an LU-LU session (on either the SESSST RU or the RSP(ACTLU)), it determines whether the session traverses an RTP connection that ends in the DLUR node. If the session does traverse such an RTP connection, the DLUR provides three additional pieces of SAW data for it:

- the network address of the RTP connection′s local endpoint, including its Network Connection Endpoint Identifier (NCE ID);
- the RTP connection′s local transport connection identifier (TCID);
- the RTP connection′s current path.

The first two of these items uniquely identify the RTP connection.

If the RTP connection subsequently performs a path switch, then the DLUR sends an update containing the same three items. The first two items identify the particular RTP connection whose path has switched, and the third provides the new path. If a SAW focal point application has saved the network address and TCID that the DLUR sent originally, then it can easily identify the LU-LU sessions affected by the path switch, and update their paths to reflect it.

The DLUR sends the update message as an SNA/MS major vector flowing in an SNA/MS Multiple Domain Support Message Unit (MDS-MU). This MDS-MU uses the same MDS infrastructure used by other SNA/MS flows, such as Alerts.

The SNA/MS focal point functions (MS Capabilities exchange) play no role in the DLUR HPR SAW function. The purpose of an MS Capabilities exchange is to provide an entry point with the name of the focal point to which it should send its unsolicited messages (such as Alerts). With DLUR HPR SAW, however, the destination of the update message is already known: it is the same DLUS to which the DLUR sent the original SAW data. Thus the MDS-MU that carries the update message has the following destination information:

- destination NETID: NETID of the DLUS;
- destination NAU Name: control point name of the DLUS;
- destination application ID: always X′23F0F5F3′, DLUR_HPR_SAW_NETOP.

For the origin information in the MDS-MU, the DLUR provides its own NETID and control point name, and the origin application ID value X′23F0F5F2′, EP_DLUR_HPR_SAW.

If the MDS-MU carrying an update message experiences some problem on its way to the DLUS, then the MDS infrastructure may return an MDS Error Message to the DLUR. The DLUR is not required to perform any actions (for example, retaining the update message and resending it later) to recover from the error. Its only responsibility is to insure that the EP_DLUR_HPR_SAW MDS application program accepts the MDS Error Message from MDS.

10.3.1 Signalling Support for DLUR HPR SAW in DLUR/S Capabilities

The DLUR/S Capabilities (X′51′) control vector contains a new flag (byte 9, bit 7) which both DLURs and DLUSs can use to signal their support for DLUR HPR SAW. This flag doesn′t simply represent whether an implementation is capable of sending (DLUR) or receiving (DLUS) the formats for DLUR HPR SAW. Rather, it indicates that the sender is prepared to supply / consume DLUR HPR SAW data to / from this particular partner. An implementation that supports DLUR HPR SAW may include a configuration option that allows an operator to disable the function, either globally or with respect to a single partner.

DLUR HPR Session Awareness is enabled between a DLUR and a DLUS if and only if both of them set the DLUR HPR SAW flag to 1 when they exchange their DLUR/S Capabilities.

10.3.2 Formats for the Initial DLUR HPR SAW Data

A DLUR identifies an RTP connection used by an LU-LU session, and provides its current path, in three additional control vectors that it includes in SESSST and RSP(ACTLU). These three control vectors are always included in the following order:

- Network Address (X′05′)
- Transport Connection Identifier (X′4B′)
- Route Selection (X′2B′).

The DLUR may include other SAW control vectors in the same SESSST or RSP(ACTLU). If it includes a COS/TPF (X′2C′) control vector and/or another Route Selection (X′2B′) control vector reporting the complete session path, then the DLUR must place these other control vectors before the three DLUR HPR SAW control vectors in the SESSST or RSP(ACTLU).

It is worthwhile making explicit something that is implicit in the preceding paragraph. Looking at the SESSST and RSP(ACTLU) from a receiver′s point of view, there may be one or two Route Selection (X′2B′) control vectors present in the request or response unit:

- If there is one Route Selection control vector and no Network Address or Transport Connection Identifier control vectors, then the SESSST or RSP(ACTLU) contains no DLUR HPR SAW data: the route in the Route Selection control vector is thus the complete session path.
- If there is one Route Selection control vector following a Network Address and a Transport Connection Identifier control vector, then the Route Selection control vector is part of the DLUR HPR SAW data: the route in the Route Selection control vector is thus the path of the RTP connection, and not the complete session path.
- If there is one Route Selection control vector preceding the Network Address and Transport Connection Identifier control vectors, and a second one following them, then the first Route Selection control vector contains the complete session path, and the second one contains the path of the RTP connection.

10.3.3 Formats for the DLUR HPR SAW Update Message

The DLUR HPR SAW update message is encoded as a DLUR HPR Session Awareness Data (X′0099′) major vector, inside a CP-MSU. The CP-MSU is transported in an MDS-MU, in exactly the same way that a CP-MSU containing an Alert (X′0000′) major vector is transported. The DLUR HPR Session Awareness major vector contains three subvectors, corresponding to the three control vectors identified in 10.3.2, "Formats for the Initial DLUR HPR SAW Data." These subvectors are almost identical to the three control vectors; the only differences are in the vector headers. Their order in the DLUR HPR Session Awareness major vector matches the order of the corresponding control vectors in SESSST and RSP(ACTLU).

Chapter 11. Multi-Subnet DLUR/S

11.1 Overview

So far this document has described DLUR/S protocols when the DLUS, DLUR, and PLU have all resided in the same APPN subnet. This chapter will describe the additional functions necessary to allow one or more of these nodes (DLUS, DLUR, PLU) to reside in different APPN subnets. These functions will impact:

- DLUS
- DLUR
- Extended Border Node (EBN)
- Central Directory Server (CDS)
- Network Node Server (NNS) of an EN DLUR

For DLURs unable to support full multi-subnet function, there is a limited DLUR multi-subnet support capability. A DLUR with limited multi-subnet support must reside in the same subnet as its DLUS, but a limited multi-subnet DLUR and the PLU can reside in different subnets. A DLUR with limited multi-subnet support has all searches for its DLUS-supported dependent LUs handled by its DLUS.

11.2 Multiple Subnet DLUR/S Configurations

There are five possible configurations to consider (there may or may not be intermediate subnets also, but the same considerations apply):

Table 11-1. Multiple Subnet DLUR/S Configurations

Configuration	DLUS	DLUR	PLU	Case 1	Case 2	Case 3
1	Subnet A	Subnet A	Subnet A			
2	Subnet A	Subnet B	Subnet B	X	X	
3	Subnet A	Subnet B	Subnet A	X		X
4	Subnet A	Subnet A	Subnet B		X	X
5	Subnet A	Subnet B	Subnet C	X	X	X

These configurations can fall into three cases, each with its respective requirements.

11.2.1 Case 1: DLUS And DLUR In Different Subnets

- Requirements/Solutions
  1. CP-SVR pipe activation rules require that the pipe not cross through a subarea network. In multiple subnet paths, this cannot be currently enforced.
     – A new indicator will be needed to suppress subarea searching in every subnet when a multi-subnet CP-SVR pipe is established.

2. Tail vector registration when the DLUR is in a different subnet cannot be used by the DLUS to provide route information to the PLU.

   – The DLUS will have to indicate that the tail vectors are not usable and that another Locate must be sent to the DLUR to acquire correct tail vectors. Also that Locate will need a special indicator so that the DLUR will return a positive reply (normally the DLUR always responds negatively to Locates for a DLUR-attached dependent LU).

11.2.2 Case 2: DLUS And PLU In Different Subnets

- Requirements/Solutions

  1. BNs need to send Locates for the SLU to the DLUS. However, they should not send BINDs for the SLU to the DLUS, since this would set up a suboptimal LU-LU session path.

     – Responsibility for acquiring tail vectors will depend on whether or not the DLUS and DLUR reside in the same subnet:

       - If the DLUS and DLUR are in the same subnet, the tail vectors returned by the DLUS, along with specifying ENCP=DLUR and NNS=DLUS, will provide enough information to calculate an LU-LU session route which does not include the DLUS (see 8.2, "DLUR EN Tail Vector Registration" on page 8-6 for more information - note tail vector registration only occurs when the DLUR is an EN).

       - If the DLUS and DLUR are in different subnets, the EBN nearest the PLU will need to send a Locate to the DLUR to acquire the correct tail vectors. The EBN also will modify the response to include NNS=BN. This will provide enough information to calculate an LU-LU session route which does not include the DLUS or even the DLUS′s subnet.

  2. DLUS needs to send Locates for the PLU through a BN. However, the BIND from the PLU should not necessarily be sent back through the same BN (especially when PLU and DLUR are in the same subnet).

     – Responsibility for acquiring tail vectors and modifying the Locate to appear as if it came from the DLUR will depend on whether or not the DLUS and DLUR reside in the same subnet:

       - If the DLUS and DLUR are in the same subnet, the tail vectors will be returned to the PLU by the DLUS. The DLUS will modify the Locate, specifying ENCP=DLUR and NNS=DLUS. The LU-LU session route will go through the same BNs as the Locate from the DLUS to the PLU.

       - If the DLUS and DLUR are in different subnets, the EBN nearest the PLU will need to send a Locate to the DLUR to acquire the correct tail vectors. The EBN also will use this information to modify the the Locate for the PLU to appear as if it came from the DLUR. This will provide enough information to calculate an LU-LU session route which does not include the DLUS or even the DLUS′s subnet.

11.2.3 Case 3: PLU And DLUR In Different Subnets

- Requirements/Solutions

  1. BNs need to send Locates for the SLU to the DLUS. However, they should not send BINDs for the SLU to the DLUS, since this would set up a suboptimal LU-LU session path.

     – Responsibility for acquiring tail vectors will depend on whether or not the DLUS and PLU reside in the same subnet:

- If the DLUS and PLU are in the same subnet, it is the DLUS′s responsibility to provide the correct tail vectors to the PLU. The DLUS will need to send a Locate to the DLUR to acquire the correct tail vectors. This will provide enough information to calculate an LU-LU session route which does not include the DLUS.

- If the DLUS and PLU are in different subnets, see 11.2.2, "Case 2: DLUS And PLU In Different Subnets" on page 11-2.

2. DLUS needs to send Locates for the PLU through a BN. However, the BIND from the PLU should not necessarily be sent back through the same BN.

   – Responsibility for acquiring tail vectors and modifying the Locate to appear as if it came from the DLUR will depend on whether or not the DLUS and PLU reside in the same subnet:

     - If the DLUS and PLU are in the same subnet, it is the DLUS′s responsibility to provide the correct tail vectors to the PLU and to modify the Locate to appear as if it came from the DLUR. The DLUS will need to send a Locate to the DLUR to acquire the correct tail vectors. This will provide enough information to calculate an LU-LU session route which does not include the DLUS.

     - If the DLUS and PLU are in different subnets, see 11.2.2, "Case 2: DLUS And PLU In Different Subnets" on page 11-2.

3. Blind BINDs (BINDs sent without a prior Locate) for DLUR-attached dependent LUs need to be intercepted by a BN to allow the DLUR to be involved in LU-LU session activation.

   – Normally an EBN would generate a Locate for the SLU and the DLUR would respond not found. The EBN needs to include an indicator in that Locate so that the DLUR will return a positive reply.

11.2.4 Configuration Restrictions

Complications arise in the Locate/Search procedures for a dependent LU served by DLUR/S nodes across multiple subnets. Rather than complicate the DLUR by requiring that it or its NNS handle Session Services Extensions Locate functions, it was decided to have the DLUS handle those functions. To do this, the DLUS must initiate and respond to Locate Searches on behalf of the SLU even though BINDs to the SLU must go through the DLUR. This means that a Locate search initiated by a PLU for a SLU must (crossing intersubnetwork TGs if necessary) reach the DLUS of the SLU which handles Session Services Extensions complications to the search. The DLUR must also be located (either through a search or by information passed back by the DLUS) since it will be the recipient of the BIND. The need to have two apparent locations for the SLU is the root cause of the complications for multiple subnet DLUR/S. These separate locations must be cached by BNs, DLUS, and central directory servers.

The following restrictions must be adhered to when configuring multiple subnets to provide DLUR/S support:

1. The CP-SVR pipe from the DLUS to the DLUR must not pass through a subarea network unless it uses a virtual route transmission group (VR TG).

2. A path through EBNs must exist between the PLU and DLUS in different subnets to support Session Services Extensions type flows. This includes a requirement for an EBN in the PLU′s subnet as well as a requirement that if the PLU is in a subarea network, the connection to the subnet containing the ICN (to that subarea) must be an EBN.

3. Within its subnet, the DLUR may have either NNs or peripheral border nodes (PBNs) which must connect to an EBN across an intersubnetwork TG along the path to a PLU in another subnet. The EBN must handle blind BINDs to the SLU. Notice that there are no Session Services Extensions type flows between the DLUR and the PLU.

4. Within its subnet, the DLUR may have either NNs or PBNs, which must connect to an EBN across an intersubnetwork TG along the path to the DLUS in another subnet.

To summarize,

- traffic entering and leaving the DLUS subnet must go through an EBN at the DLUS subnet boundary
- traffic entering and leaving the PLU subnet must go through an EBN at the PLU subnet boundary
- traffic entering and leaving the DLUR subnet must go through an EBN, a PBN, or a NN at the DLUR subnet boundary

11.3 Functional Description

In the previous section various solutions to requirements for multi-subnet DLUR/S support were outlined. What follows are the details of this support.

Added function must be put in EBNs, DLUS nodes, DLUR nodes, and CDSs. The function required differs depending on the distribution of the PLU, DLUR, and DLUS among subnets. If the DLUR and DLUS are both in the same subnet, no redirected Locate Searches to Locate the DLUR are required. In all other cases, either the DLUS or an EBN initiates a second search. Which node conducts the extra search and when it does so depends on the configuration of subnets and where EBNs are located between subnets. CDS must cache two locations for each SLU, one being the location of the DLUS and the second being the location of the NNS(DLUR).

11.3.1 Border Node Terminology

- Origin Border Node (OBN)
  - The OBN is the nearest EBN to the OLU along a Locate path. This BN need not be in the same subnet as the OLU. If a BN receives a search and the Intersubnetwork TGs Crossed count is zero, then the BN can assume that it must serve as the OBN. The OBN may be required to perform searches to Locate the DLUR of a SLU in the case of a PLU-initiated search.
- Destination Border Node (DBN)
  - The DBN is the nearest EBN to the DLU along a Locate path. This BN need not be in the same subnet as the DLU, however, in general, the BN will not be able to determine that it is a DBN unless it is in the same subnet or across an intersubnetwork TG from the subnet containing the DLU. The DBN may be required to initiate a search for the DLUR of the SLU in the case of a SLU-initiated search.

11.3.2 Multi-Subnet DLUR/S Indicators

- DSR - DLUR Search Required
  - one bit field carried in Locate
  - indicates that the tail vectors in the CDINIT are not usable in the BIND and a follow-up search to locate the resource on the DLUR is required. This bit always applies to the SLU. This bit is turned on by the DLUS when it generates a Found if the DLUS and DLUR are in different

subnets (this information is available when the CP-SVR pipe is set up). The DSL bit (see below) is always set when the DSR is set.

- DSL - DLUS-Served LU
  - one bit field carried in the Find and Found GDS variables
  - indicates that the SLU is a DLUS-served LU. The DSL bit is set in a SLUinit in the Locate Find by the DLUS node initiating the search. The DSL bit is also set by the DLUS or DLUR in the Found in the Locate reply if the found resource is a DLUS-served LU.
- OCR - Owning CP Respond
  - one bit field carried in the Find and Found GDS variables
  - If the OCR is set in the Find, the DLUR owning the resource will respond with a Found. If the OCR is not set, a DLUR will respond not found to the resource if it is a DLUS-served LU owned by the DLUR.
- PSP - Prevent Subarea Path
  - one bit field carried in the Locate GDS variable
  - When this bit is set, the BN does not reset the Suppress Subarea Search bit. This bit is set when the DLUS or DLUR is sending a Locate as part of the CP-SVR pipe activation since the pipe may not cross a subarea network.
- Limited DLUR Multi-Subnet Support
  - one bit field carried in the DLUR/S Capabilities control vector
  - When this bit is set, the DLUR will not respond to searches with the OCR bit set; the DLUS will respond to these searches.

11.3.3 Multi-Subnet CP-SVR Pipe Activation

When the DLUS wants to set up the CP-SVR pipe, the DLUS must initiate a Locate for the DLUR. When the DLUR wants to set up the CP-SVR pipe, the DLUR must initiate a Locate for the DLUS. In either case the Locate must not cross a subarea. To insure this, the node initiating the Locate must set both the Suppress Subarea Search (SSS) bit in the Locate and the Prevent Subarea Path (PSP) bit in the Locate. Once the partner (DLUS or DLUR) is located without crossing a subarea, the CP-SVR pipe may be established just like in the single subnet case.

During CP-SVR pipe activation, if a DLUS detects:

- the CP-SVR pipe crosses a subnet boundary, and
- the DLUR only supports limited multi-subnet function in the DLUR/S Capabilities control vector

then the DLUS will deactivate the CP-SVR pipe, using sense data X′088E 000F′ in the CV X′35′ of the UNBIND(s) (see 5.6.2.2, “DLUR/S Capabilities Mismatches” on page 5-32 for more information on DLUR/S capabilities mismatch processing).

11.3.4 Multi-Subnet EN DLUR Capabilities and Resource Registration

An EN DLUR should be able to register its LUs to allow its NNS to handle Locates for these LUs without having to forward the Locates to the DLUR. For DLUS-served LUs, this would require the NNS to be able to identify this type of LU and know how to handle Locates for it. This function is called DLUS-served LU registration, and its NNS impacts are identified as option set 1116.

The DLUR impacts are in the base DLUR architecture, namely that since the DLUR cannot be guaranteed that its NNS is uplevel, it must be able to identify the capabilities of its NNS and handle its resource registration accordingly.

11.3.4.1 NNS Capability For DLUS-Served LU Registration (Option Set 1116)

A new bit, the DLUS-served LU registration indicator, will be defined in format 3 of the XID I-field. This bit will only be used by a NN. When it is used, setting the indicator will signify that the NN supports DLUS-served LU registration (option set 1116). Resetting the indicator will signify that the NN does not support DLUS-served LU registration.

11.3.4.2 Limited Multi-Subnet Support EN DLUR Processing

An EN DLUR with limited multi-subnet support will always perform the following processing - it will not examine the DLUS-served LU registration indicator:

> An EN DLUR with limited multi-subnet support may choose to reset the Search Status Indicator in the End Node Control Point (ENCP) Search Control CV (X′33′) during the CP Capabilities exchange with its network node server if it registers all non-DLUS-served LUs with its NNS (this significantly cuts down domain broadcast traffic). Such an EN need not register its DLUS-served LUs, since the DLUR will always respond not Found to any Locate for its DLUS-served LUs.

11.3.4.3 Full Multi-Subnet Support EN DLUR Processing - NNS Without Option Set 1116

An EN DLUR with full multi-subnet support will perform the following processing when it receives an XID3 from its NNS that has the DLUS-served LU registration indicator reset:

> An EN DLUR with full multi-subnet support must set the Search Status Indicator in the ENCP Search Control CV (X′33′) during the CP Capabilities exchange with its network node server.
>
> An EN may choose not to set this indicator if it registers all LUs with its NNS (this significantly cuts down domain broadcast traffic). However, an EN cannot register its DLUS-served LUs with a down-level NNS, since the NNS will delete the directory entry as soon as the EN rejects a Locate (which it must do if the OCR bit is not set). Even if a DLUR-capable EN registers all its non-DLUS-served LUs, it must still set the Search Status Indicator.

11.3.4.4 Full Multi-Subnet Support EN DLUR Processing - NNS With Option Set 1116

An EN DLUR with full multi-subnet support will perform the following processing when it receives an XID3 from its NNS that has the DLUS-served LU registration indicator set:

> An EN DLUR with full multi-subnet support may choose to reset the Search Status Indicator in the ENCP Search Control CV (X′33′) during the CP Capabilities exchange with its network node server if it registers all LUs with its NNS (this significantly cuts down domain broadcast traffic). It must register its DLUS-served LUs and non-DLUS-served LUs separately.

To register its DLUS-served LUs with its uplevel NNS, the DLUR will send a Register Resource (X′12C3′) GDS variable with a new DLUS-Served LU Indicator set to specify that the resources being registered are DLUS-served LUs.

When the DLUR sets the DLUS-Served LU Indicator on in the Register Resource (X′12C3′) GDS variable, it must also reset the Central Resource Registration Request Indicator (both indicators are defined in the same byte).

11.3.4.5 NNS Processing For Option Set 1116

A NN with DLUS-served LU registration support will perform the following processing:

The NN will send an XID3 with the DLUS-served LU registration indicator set.

When the NNS receives a Register Resource (X′12C3′) GDS variable with the DLUS-served LU indicator set, it will mark the resources being registered as DLUS-served LUs.

If the corresponding EN DLUR′s Search Status indicator is reset,

- when the NNS receives a Locate with the OCR bit off for a registered DLUS-served LU,
  - the NNS will not forward the Locate
  - the NNS will not delete the registered resource from the directory
  - the NNS will handle the Locate as if it had received a negative reply to it from the DLUR
- when the NNS receives a Locate with the OCR bit on for a registered DLUS-served LU,
  - the NNS will forward the Locate
  - the NNS will delete/retain the registered resource pending the results in the Locate reply

If the corresponding EN DLUR′s Search Status indicator is set,

- when the NNS receives a Locate with the OCR bit off for a registered DLUS-served LU,
  - the NNS will forward the Locate
  - the NNS will not delete the registered resource from the directory
- when the NNS receives a Locate with the OCR bit on for a registered DLUS-served LU,
  - the NNS will forward the Locate
  - the NNS will delete/retain the registered resource pending the results in the Locate reply

11.3.5 Multi-Subnet DLUR/S Locate Searches

The DLUS must respond to PLU-initiated Locates for DLUS-served LUs. The DLUS must initiate searches on behalf of the SLUs it serves. In both the SLU-initiated and PLU-initiated searches, the ensuing BIND must go from the PLU to the DLUR (not the DLUS) before reaching the SLU. The mismatch between the endpoints of the BIND and the Locate must be accounted for. The DLUS and EBN redirected searches required to handle this are described below.

11.3.5.1 PLU-initiated searches

For PLU-initiated searches, either the DLUS or OBN (if one exists) making the Locate response to the PLU must indicate that the BIND should be sent to the DLUR. If the DLUS does not already know the location of the DLUR (relative to the PLU), either the DLUS or the OBN must find it for the PLU.

If the DLUS and DLUR are in the same subnet, the DLUS will know the TGs for the DLUR and can return them in the Locate Found. In this case no additional search for the DLUR is required. The only modification the DLUS will need make to the Locate reply is return the DLUR TGs with the DLUR listed as ENCP in the reply. Both the DLUS and DLUR for the SLU must be located before the session is established. The DLUS(SLU) must be located first. In the case where the DLUR and DLUS are in the same subnet, only the Locate to the DLUS is necessary since the DLUS knows the TGs to the DLUR which can be used to determine the best route to the DLUR.

When the DLUS is in a different subnet than the DLUR, a separate search to the DLUR must be conducted to find a multisubnet route between the PLU and DLUR. This search will be initiated by the OBN when one is available. If there is no OBN, then the PLU is in the same subnet as the DLUS. In this case the DLUS must initiate the search (which verifies a COS-acceptable route across subnets). When the DLUR is located by the OBN or DLUS in this manner, the Locate Found returned to the PLU is modified so that it will be possible for the PLU to establish a session with the SLU which goes through the DLUR but need not go through the DLUS or even the subnet containing the DLUS.

What follows is the specific role of the DLUR node in a multi-subnet DLUR/S PLU-initiated search:

11.3.5.1.1 DLUR / PLU-initiated search: The function of the DLUR is to serve as the intermediary between the dependent LU and the SSCP on the DLUS. The session between a PLU and a DLUS-served SLU must pass through the DLUR to reach the SLU.

A DLUR will return a positive reply to a search for one of its dependent resources only if the Find has the OCR bit set and the DLUR supports full multi-subnet function. Only an EBN or DLUS should initiate a search with the OCR bit set. Except for when the EBN generates a search in response to a blind BIND, no searches for a dependent LU with the OCR bit set should be initiated unless the DLUS has first received a search without the bit set. Searches initiated by downlevel nodes will never have this bit set and will always locate the DLUS rather than the DLUR.

The DLUR Found response will have the DSL bit set, indicating that the SLU is a dependent LU. On a DLUR search reply, the DLUR should set the NNS(DLUR) as the NNS(SLU), the DLUR as ENCP (not required if DLUR=NNS(DLUR)) and the DLUS is carried in the CV X′40′.

11.3.5.2 SLU-initiated searches

When a SLU wishes to initiate a search, the DLUS must begin the search on behalf of the SLU to find the PLU. The DSL bit is set to indicate that the SLU is a DLUS-served LU. The Locate is initiated at the DLUS. However, when the PLU receives the Locate, it must send a BIND to the SLU by way of the DLUR (and not back to the DLUS). The complication of the SLU-initiated search comes about because the multi-subnet route that the BIND takes from the PLU to the SLU by way of the DLUR may pass through a different set of BNs than the Locate did in its path from the DLUS to the PLU. The SLU-initiated Locate sent to the PLU must contain the appropriate node or TGs to get the BIND to the correct destination. If the PLU is local to the DLUR subnet, the TGs to the DLUR (if an EN) or an RSCV containing the DLUR as last entry (if a NN) are carried by the Locate to the PLU. When the PLU is not local to the DLUR subnet, the TGs will be those appropriate to leave the PLU subnet along a path to the DLUR. The DLUS will be carried as a CV X′40′ when an EBN has been crossed in the path between the DLUS and the PLU.

11.3.5.2.1 DLUR / SLU-initiated search: The function of the DLUR is to serve as the intermediary between the dependent LU and the SSCP on the DLUS. The session between a PLU and a DLUS-served SLU must pass through the DLUR to reach the SLU. A DLUR will return a positive reply to a search for one of its dependent resources only if the Find has the OCR bit set and the DLUR supports full multi-subnet function. Only an EBN or DLUS should initiate a search with the OCR bit set. Searches initiated by downlevel nodes will never have this bit set and will always locate the DLUS rather than the DLUR. The DLUR Found response will have the DSL bit set, indicating that the SLU is a dependent LU. On a DLUR search reply, the DLUR should set the NNS(DLUR) as the NNS(SLU), the DLUR as ENCP (not required if DLUR=NNS(DLUR)) and the DLUS is carried in the CV X′40′.

11.3.6 Multi-Subnet Functional Impact Summary

The following DLUR functions are required for DLUR/S support in a multiple subnet environment:

11.3.6.1 DLUR / Multi-Subnet Impacts

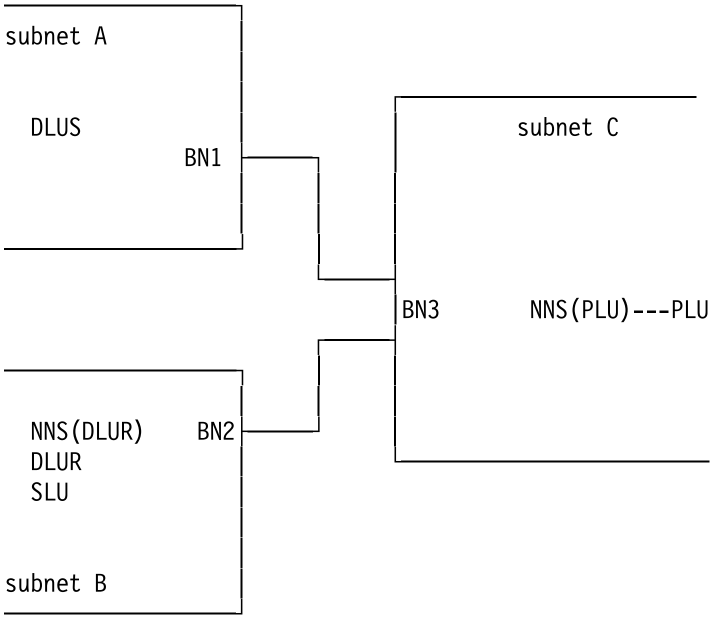
- The DLUR returns not found for Locates for its DLUS-served resources unless the OCR bit is set and the DLUR supports full multi-subnet function.
- The DLUR returns Found (with the DSL bit set) if the OCR bit is set in the Find and the DLUR owns the resource (identifies the SLU as served by a DLUS) and the DLUR supports full multi-subnet function.
- The DLUR returns Found with CV X′40′ indicating the DLUS name for use by management services.
- The DLUR sets the PSP bit when initiating a Locate for the DLUS for the purpose of setting up the CP-SVR pipe. This prevents setting up a pipe across a subarea.
- Wildcards do not apply to DLUS-served LUs. All DLUS-served LUs must be explicitly defined on the DLUR and on the DLUS. No wildcard responses are made with the DSL bit set.
- A DLUR-capable EN must set the Search Status Indicator in the ENCP Search Control CV (X′33′) during the CP Capabilities exchange with its network node server when the EN has full multi-subnet support but its NNS does not support DLUS-served LU registration. When an EN DLUR has full multi-subnet support and its NNS supports DLUS-served LU registration, the Search Status indicator can be reset and the DLUS-served LUs can be registered with the NNS (with the Central Resource Registration Request Indicator reset). When an EN DLUR has limited multi-subnet support, the Search Status Indicator can be reset and the DLUS-served LUs need not be registered with the NNS.

11.4 Multi-Subnet DLUR/S Flows

11.4.1 Multi-Subnet PLU-Initiated Searches

11.4.1.1 DLUS, DLUR, PLU In Different Subnets

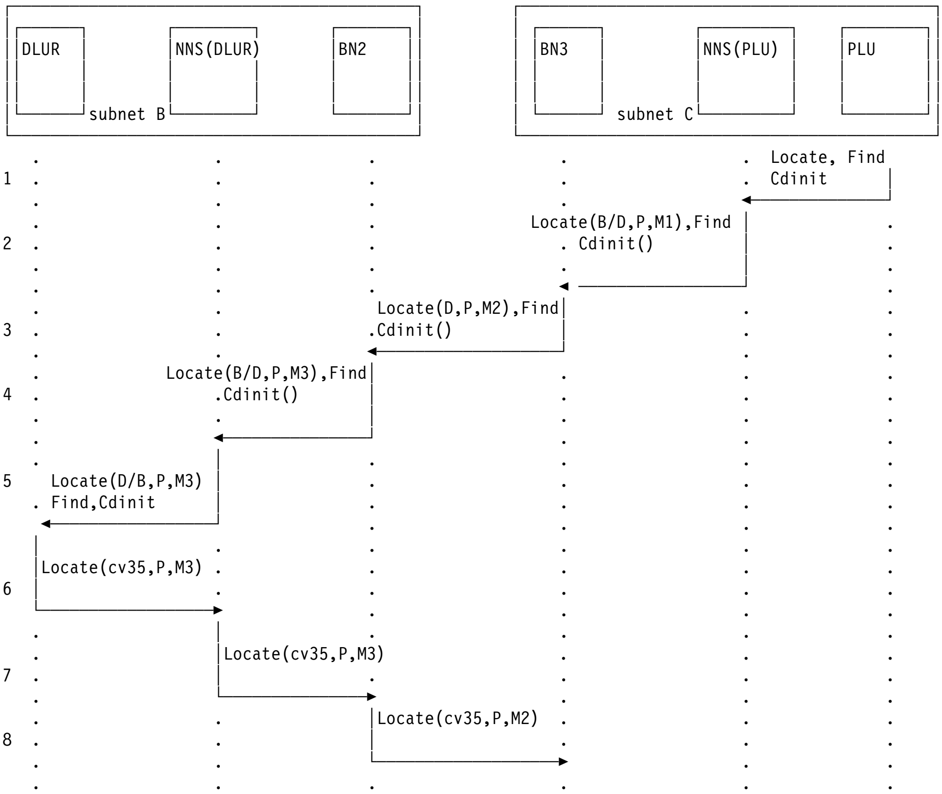
Note: Refer to Table 3-1 on page 3-2 for an explanation of the notations used in the diagrams in this chapter.



SUBNET lists

BN1 Search for net	at BN	BN2 search for net	at BN	BN3 search for net	at BN
NETC(all)	BN3	NETC(all)	BN3	NETC(all)	BN3
NETB(OCR off)	BN1,BN3	NETB(OCR off)	BN2,BN3	NETB(OCR off)	BN2,BN1
NETB(OCR on)	BN3	NETB(OCR on)	BN2	NETB(OCR on)	BN2
NETA(all)	BN1	NETA(all)	BN3	NETA(all)	BN1

Figure 11-1. DLUS, DLUR, PLU in different subnets / one BN per subnet



D - directed search
B - broadcast search
Mi - PCID modifiers
P - PCID
INS - Internet Search
SS - Subnet Search
DSL - DLUS-Served LU
DSR - DLUR Search Required
OCR - Owning CP Respond

Figure 11-2. PLU-initiated search for dependent LU when DLUS, DLUR and PLU are all in different subnets.

Notes:

1. The search for NETB.SLU is originated by the PLU in subnet C which forwards the search to its Network Node Server.
2. The NNS(PLU) does not have the location of SLU cached and sends a broadcast which reaches a Border Node (BN3). The “DLUS-Served LU” (DSL) bit is off.
3. BN3 checks its subnet list and finds two entries for NETB with the DSL bit off. The search is forwarded from BN3 to BN2 since BN2 is the first entry in the list. BN3 caches the PCID and remembers that the DSL bit was off.
4. BN2 checks its subnet list for NETB and finds entries for its local net and for BN3. BN2 first searches its local net by sending out a broadcast with the internet search bit on (to prevent exiting the subnet). The DSL bit is off. BN2 caches the PCID for the search along with the information that the DSL bit was off.
5. NNS(DLUR) forwards the search to DLUR.

6. DLUR returns a not Found since it does not respond to a search for the dependent LU (SLU) unless the search comes in with the OCR bit on which indicates that the DLUS has properly set up the pipe between the DLUS and the DLUR.
7. NNS(DLUR) returns a not Found to BN2.

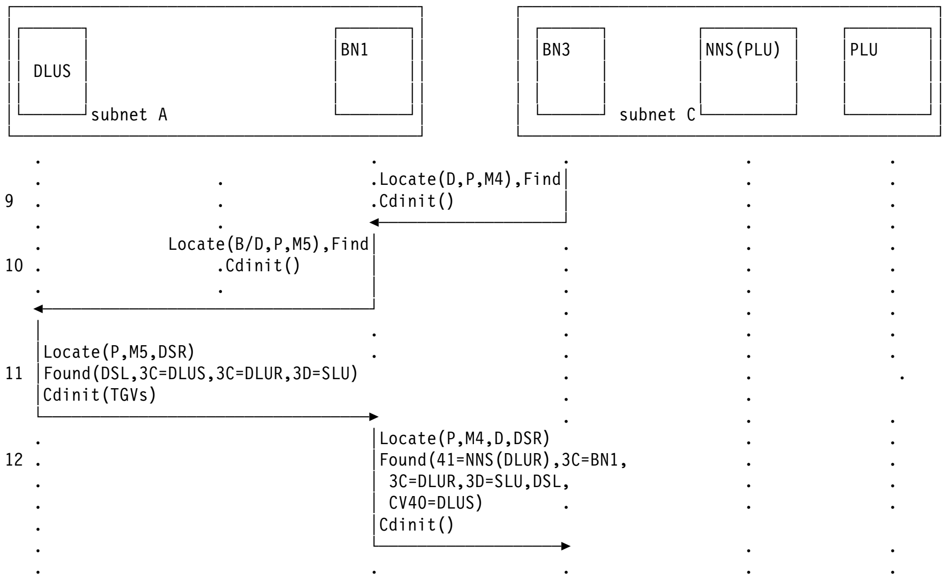


Figure 11-3. PLU-initiated search for dependent LU support when DLUS, DLUR and PLU are all in different subnets (cont.).

8. BN2 has discovered that the SLU is not in subnet B so BN2 checks its subnet list for the next place to search for SLU. Based on the subnet list, BN2 sends a not Found to BN3.
9. BN3 checks its subnet list and finds that a resource with NETID NETB may be in BN1 and forwards a search to BN1.
10. BN1 initiates a search of its local subnet (NETA) by sending out a broadcast.
11. DLUS receives the search for SLU, and realizes that it supports that particular resource through DLUR. DLUS assumes the role of network node server of SLU with DLUR as the ENCP (even if DLUR had been a NN, it is treated as an ENCP in the hierarchy). The Found returned to BN1 contains these control vectors indicating the hierarchy: X′3C′(DLUS, X′F6′), X′3C′(DLUR, X′F4′), and X′3D′(SLU, X′F3′). The DSL bit is set since it is known that SLU is a DLUS-served LU. The DSR bit is set since the TGs returned by DLUS are not appropriate to reach the DLUR. The Found is returned to BN1.
12. BN1 replaces the name in the first control vector (X′3C′) with its own name and returns it to BN3. The DLUS name which was initially in this CV X′3C′ is put in a CV X′40′. BN1 caches the location of DLUS(SLU) for future searches.

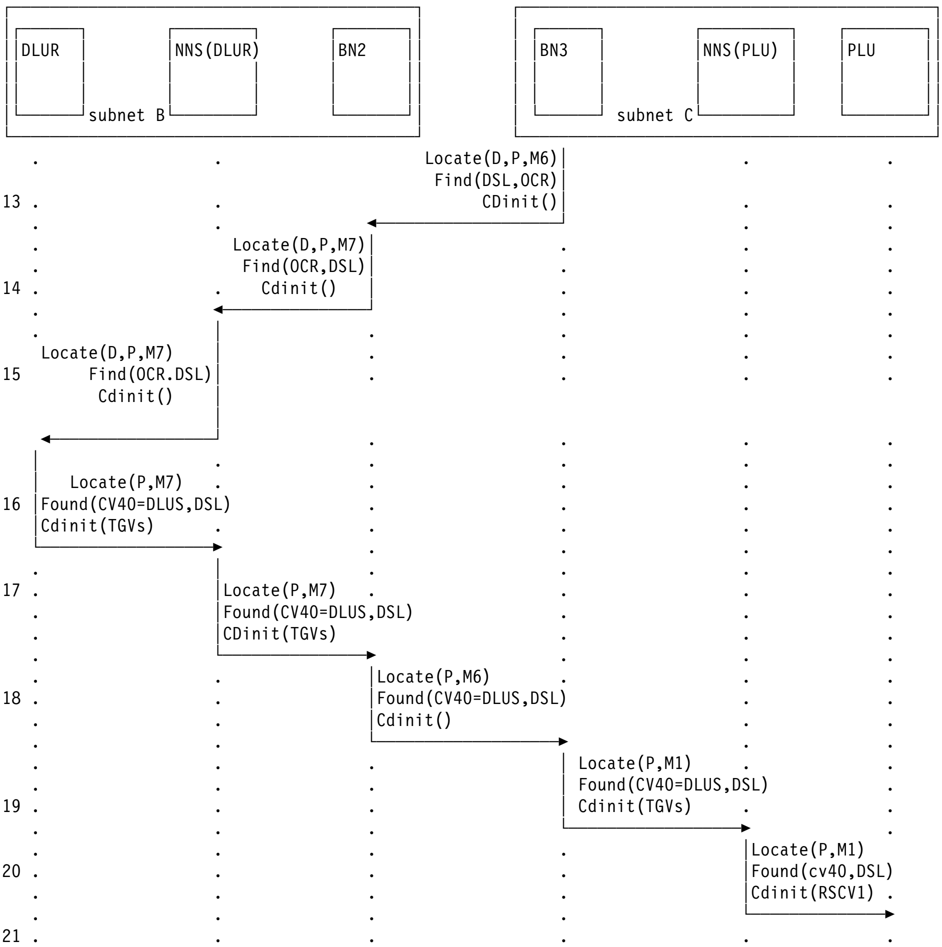


Figure 11-4. PLU-initiated search for dependent LU support when DLUS, DLUR and PLU are all in different subnets (cont.).

13. BN3 replaces the name in the first control vector (X′3C′) with its name. It caches the location of SLU by way of the DLUS as BN1. Since BN3 is the originating BN for this search it has extra responsibilities. The DLUS has been located. Now BN3 must send a Locate Find (with the OCR bit set and an incremented Procedure Resubmit Number (PRN)) to locate the SLU by way of the DLUR. BN3 sees from its subnet list (and the OCR bit set) that the appropriate BN to send the search to is BN2.

14. BN2 had previously cached the PCID for this search but with a different PRN. Since this search has a new PRN, BN2 will accept this search (and cache the fact that this PCID with its PRN was received). BN2 sees from its subnet list that NETB.SLU might be in its local subnet. BN2 checks its cache, and if the SLU by way of the DLUR is in its cache, BN2 sends a directed search to NNS(DLUR) for SLU. Otherwise BN2 sends a broadcast search for the SLU with the Internet Search (INS) and Subnet Search (SS) bits set to prevent exiting the subnet.

15. When NNS(DLUR) receives the search it is sent to DLUR.

16. DLUR sees that the OCR bit is on, therefore it responds with a Found. The DSL bit is set and CV X′40′=DLUS is carried on the Found.

17. NNS(DLUR) returns the Found to BN2 with CV X′3C′=NNS(DLUR), CV X′3C′=DLUR, and CV X′3D′=SLU.

18. BN2 replaces the first CV X′3C′ with its own name but does not change the CV X′40′. BN2 caches the location of the SLU by way of the DLUR as being at NNS(DLUR). The Found is sent to BN3.

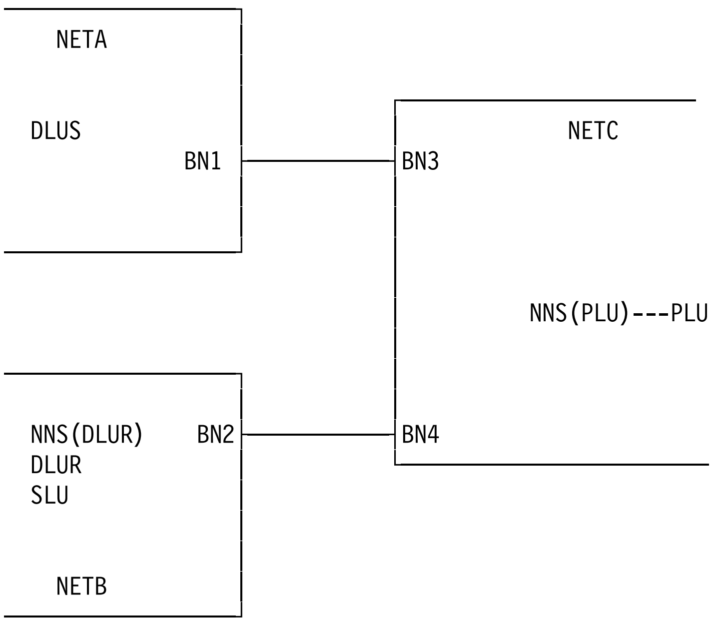
19. BN3 caches the location of SLU by way of the DLUR as BN2. BN3 inserts its own name as the CV X′3C′ indicating NNS and returns the Found to NNS(PLU).

20. NNS(PLU) caches BN3 as the NNS of SLU and sends the Found back to PLU.

21. PLU sends a BIND to BN3. BN3 checks its cache (associates the Locate with the BIND PCID and the highest PRN) and sends the BIND on to BN2. BN2 checks the cache and forwards the BIND on to NNS(DLUR) and to the DLUR from there. The BIND response is sent back through BN2 and BN3 to PLU.

11.4.2 Multi-Subnet SLU-Initiated Searches

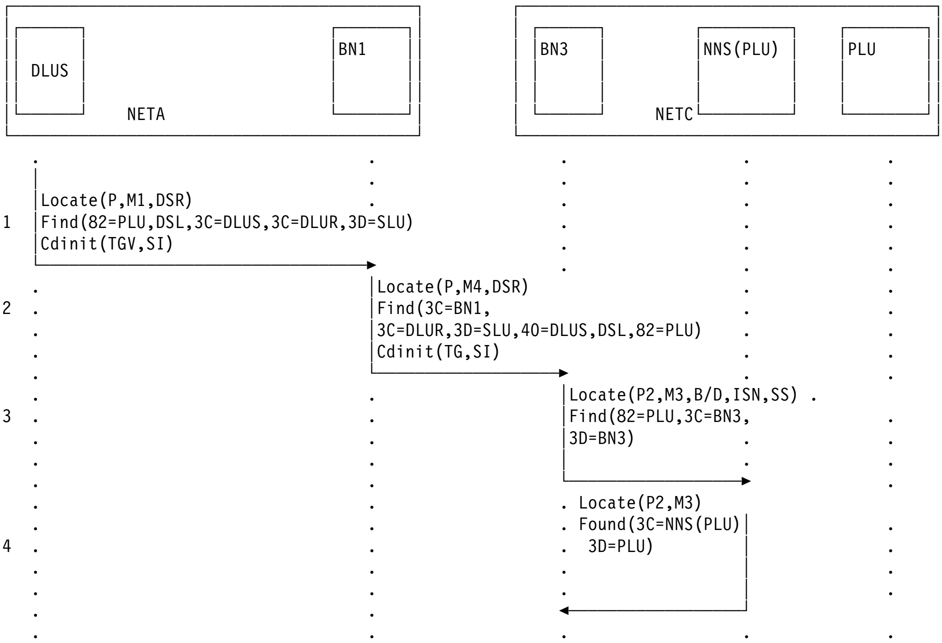
11.4.2.1 DLUS, DLUR, PLU In Different Subnets



SUBNET lists

BN1		BN2		BN3		BN4	
Search for net	at BN	search for net	at BN	search for net	at BN	search for net	at BN
NETC(all)	BN3	NETC(all)	BN4	NETC(all)	BN3	NETC(all)	BN4
NETB(OCR off)	BN1,BN3	NETB(OCR off)	BN2,BN4	NETB(OCR off)	BN4,BN1	NETB(OCR on)	BN2
NETB(OCR on)	BN3	NETB(OCR on)	BN2	NETB(OCR on)	BN4	NETB(OCR off)	BN2,BN3
NETA(all)	BN1	NETA(all)	BN4	NETA(all)	BN1	NETA(OCR on)	BN3

Figure 11-5. DLUS, DLUR, PLU in different subnets / more than one BN per subnet



D - directed search
B - broadcast search
Mi - PCID modifiers
P - PCID
DSL - DLUS-Served LU
SI - SLU Init
DSR - DLUR Search Required
PRN - Procedure Resubmit Number
INS - Internet Search
SS - Subnet Search
OCR - Owning CP Respond

Figure 11-6. SLU-initiated search when DLUS, DLUR and PLU are all in different subnets.

1. DLUS generates a SLU-initiated search on the behalf of SLU to Locate NETC.PLU. The Find is sent to BN1 with the DSL bit set indicating that the SLU is a DLUS-served LU. The Locate indicates (with the DSR bit) that the TG carried in the CDINIT is not appropriate for a PLU outside this subnet to use in a BIND. The TG carried in the CDINIT is one between BN1 and BN3 over which the CP-SVR pipe is set up.

2. BN1 checks its subnet list and discovers that to reach NETC it must go through BN3. BN1 removes the RSCV from NETA, inserts its name as the NNS in the hierarchy, and puts DLUS in CV X'40'. The search is sent to BN3.

3. BN3 checks its subnet list and realizes that it may be the entry BN into the destination subnet of a DLUS-served SLU-initiated search. Therefore BN3 may optionally verify this. If BN3 chooses, it initiates a search within NETC to determine if the PLU actually is in its local subnet. This is not really a redirection of the original search; rather, it is treated as a search originating at BN3. BN3 inserts its name as the originator of the search. A new PCID is chosen and the INS and SS bits are set to prevent the search from exiting the subnet.

4. When the NNS(PLU) (also the CP in this case) receives the search, it responds with a Found.

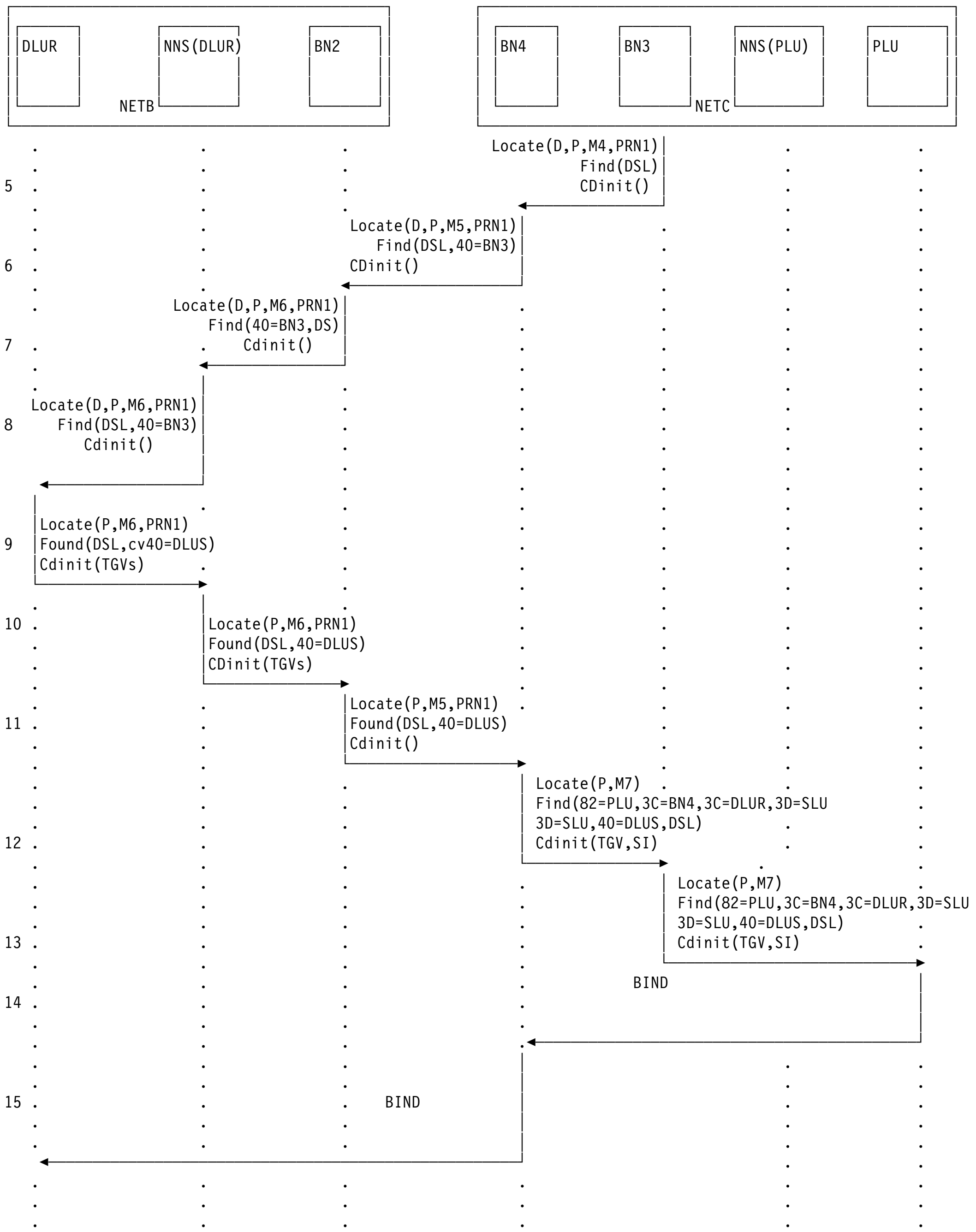


Figure 11-7. SLU-initiated search when DLUS, DLUR and PLU are all in different subnets (continued).

5. BN3 replaces the name in the first (NNS) control vector X′3C′ with its name. It caches the location of SLU by way of the DLUS as BN1. Since BN3 is the originating BN for this search and the Find has the DSR bit set, the BN has extra responsibilities. Since the DLUS has been located, its name is in the CV X′40′. This information is saved and will later be sent back to the PLU. Now BN3 must send a Locate Find (with the OCR bit set and an incremented PRN) to locate SLU. BN3 sees from its subnet list that the appropriate BN to send the search to is BN4.

6. BN4 receives the search from BN3 and indicates itself to be the NNS and puts BN3 in the CV X′40′. BN4 sees from its subnet list that it must send the search to BN2.

7. BN2 will accept this search (and cache the fact that this PCID/PRN was received). BN2 sees from its subnet list that NETB.SLU might be in its local subnet. BN2 will check its cache for the location of SLU by way of the DLUR. If SLU is found, BN2 sends a directed search to NNS(DLUR) for SLU; otherwise a broadcast search restricted to BN2′s local subnet (NETB) will be initiated.

8. When NNS(DLUR) receives the search it sends it on to DLUR.

9. DLUR sees that the OCR bit is on, therefore it responds with a Found (DSL bit on). A CV X′40′ is appended with the name of DLUS.

10. NNS(DLUR) returns Found to BN2 with CV X′3C′=NNS(DLUR), CV X′3C′=DLUR, CV X′3D′=SLU and CV X′40′=DLUS.

11. BN2 replaces the first CV X′3C′ with its own name. BN2 caches the location of the SLU by way of the DLUR as being at NNS(DLUR). The Found is sent to BN4.

12. BN4 accepts the Found, indicates itself as being the NNS(SLU), caches the location of SLU by way of the DLUR as being at BN2, and sends the search back to BN3 with a TG from BN4 to BN2.

13. BN3 caches the location of SLU by way of the DLUR as BN4. BN3 now has the information it needs to modify the original SLU-initiated search and send it on to NNS(PLU). The search must indicate that BN4 is the NNS since it is the BN on the route to the DLUR. The PCID, modifier, and PRN are set to indicate that the original SLU-initiated search is continuing. The TG from BN4 to BN2 is put in the CDINIT. BN3 keeps the name of BN4 as the NNS(SLU). The CV X′40′=DLUS is still included in the Find (for management services at the PLU). The modified Find is sent on to NNS(PLU).

14. NNS(PLU) caches BN4 as the NNS of SLU. It uses the TG in the CDINIT, computes a RSCV to BN4, and sends PLU instructions to initiate a BIND.

15. PLU sends BIND to BN4. BN4 checks its cache and sends the BIND on to BN2. BN2 checks the cache and forwards the BIND on to NNS(DLUR) and to the DLUR from there. The BIND response is sent back through BN2 and BN4 to the PLU.

Chapter 12. DLUR Option Set Content

12.1 DLUR Tower Requirements

Any product that implements the DLUR tower must implement the following functions as described in this document:

12.1.1 CP-SVR Pipe

The DLUR node will have to be able to establish the CP-SVR pipes in conjunction with the activation or reactivation of a given SSCP-PU session as described in Chapter 5, "CP-SVR Pipe" on page 5-1. This will specifically include the ability to:

1. Support the new CPSVRMGR mode and COS.
2. Know the DLUS node to be contacted when a PU requires ACTPU.
3. De-activate CP-SVR pipes when idle.
4. Bring down a conwinner session when the conloser fails.

12.1.2 SSCP-PU & SSCP-LU Session Activation/Encapsulation

The DLUR node will also be required to activate SSCP-PU and SSCP-LU sessions over the CP-SVR pipes. This will include both logic to establish the sessions as well as logic to encapsulate and de-encapsulate the associated flows. Specific requirements include the ability to:

1. Generate an FQPCID for a PU requiring activation.
2. Correlate SSCP-PU and SSCP-LU sessions to PU images on the basis of the FQPCID.
3. Generate the REQACTPU RU and understand related responses and sense codes.
4. Interrogate ACTLU commands to build a table that relates SLU names to the PU image that supports them.
5. Maintain an active session count for each dependent LU.
6. Examine +RSP(ACTLU) for extended BIND and network-qualified name support information.
7. Examine INIT-SELF to acquire PLU name and URC information, and include a URC in the RU if one is not already present.
8. Optionally provide SDDLU NMVT support to allow for dynamic dependent LU definition in the DLUS (required only if dynamic definition is supported).
9. Detect ACTPU race conditions and resolve them appropriately.
10. Generate the REQDACTPU RU and understand related responses and sense codes.
11. Encapsulate and de-encapsulate SSCP-PU and SSCP-LU FID2 PIUs.
12. Send and receive multiple FID2 Encapsulation (X′1500′) GDS variables in a single BIU.
13. Support network name forwarding.

12.1.3 EN TGV Reporting

DLUR ENs will be required to report their TGVs up to the DLUS node only if the DLUR and the DLUS are in the same subnet. These TDUs will be sent as events that effect them occur. In the case where the DLUR and the DLUS are non-adjacent (or no CPSVCMG session exists between them), these TDUs will flow over the CP-SVR pipe. In the case where the DLUR and DLUS are also connected via a CPSVCMG session, these TDUs may optionally flow over the CP-CP session or over the CP-SVR pipe.

12.1.4 LU-LU BIND/UNBIND Processing

1. Provide reformating services:
   - Unextend extended BINDs destined for nodes that do not support receipt of such BINDs.
   - Reextend unextended +RSP(BIND)s when the corresponding BIND request was unextended.
   - Convert short +RSP(BIND)s into extended short form +RSP(BIND)s.
   - Unextend extended UNBINDs destined for nodes that do not support receipt of extended BINDs - no reextending of the +RSP(UNBIND) is required.
2. Remove NETIDs from LU names in BINDs destined for nodes that do not support network-qualified names.
3. Replace PLU names in BINDs destined for nodes which specified a PLU name in the INIT-SELF different from the PLU name received in the BIND, removing DLUR-generated URCs from the BIND.
4. Reject BINDs received for already active dependent LUs.
5. Properly set the Whole BIUs Required Indicator (WBRI) in the RSP(BIND) dependent on the segment reassembly capabilities of the DLUR BF.
6. Properly set the adaptive session pacing indicator in the BIND and +RSP(BIND) for the REX and APPN stages and set the REX stage pacing window size in the BIND when required.
7. Route BINDs to the destination PU/LU based upon the SLU name specified in the NS field of the BIND and information maintained in the SLU name table.
8. Perform intermediate routing functions when the LU is located on a downstream PU. This includes:
   - Intermediate routing function
   - Session outage notification
   - Maintaining pacing counts/indicators in both directions

12.1.5 Session Awareness Reporting

In order to assist the DLUS node in providing session limit management, the DLUR node must provide the DLUS node with session awareness data. The DLUR node will be required to:

1. Upon receipt of a positive RSP(ACTLU) from the attached dependent LU, if the dependent LU has an active LU-LU session, the DLUR must build a Session Information (X′2A′) control vector, append it to the RSP(ACTLU), and send the RU to the SSCP by encapsulating the RU and sending it on the CP-SVR pipe to the DLUS.
2. Upon receipt of a positive RSP(BIND) from the attached dependent LU, the DLUR must build a SESSST RU and send it to the SSCP by encapsulating the RU and sending it on the CP-SVR pipe to the DLUS.

3. Upon receipt of a RSP(UNBIND) from the attached dependent LU, the DLUR must build a SESSEND RU and send it to the SSCP by encapsulating the RU and sending it on the CP-SVR pipe to the DLUS.

12.1.6 Multi-Subnet Support

1. The DLUR returns not found for Locates for its DLUS-served resources unless the OCR bit is set and the DLUR supports full multi-subnet function.
2. The DLUR returns Found (with the DSL bit set) if the OCR bit is set in the Find and the DLUR owns the resource (identifies the SLU as served by a DLUS) and the DLUR supports full multi-subnet function.
3. The DLUR returns Found with CV X′40′ indicating the DLUS name for use by management services.
4. The DLUR sets the PSP bit when initiating a Locate for the DLUS for the purpose of setting up the CP-SVR pipe. This prevents setting up a pipe across a subarea.
5. Wildcards do not apply to DLUS-served LUs. All DLUS-served LUs must be explicitly defined on the DLUR and on the DLUS. No wildcard responses are made with the DSL bit set.
6. A DLUR-capable EN must set the Search Status indicator in the ENCP Search Control CV (X′33′) during the CP Capabilities exchange with its network node server when the EN has full multi-subnet support but its NNS does not support DLUS-served LU registration. When an EN DLUR has full multi-subnet support and its NNS supports DLUS-served LU registration, the Search Status Indicator can be reset and the DLUS-served LUs can be registered with the NNS (with the Central Resource Registration Request Indicator reset). When an EN DLUR has limited multi-subnet support, the Search Status Indicator can be reset and the DLUS-served LUs need not be registered with the NNS.

12.2 DLUR HPR Session Awareness Option Set (APPN Option Set 1058)

A DLUR implementation may optionally include support for the DLUR HPR SAW function described in 10.3, “DLUR HPR Session Awareness” on page 10-41. This support is identified in the APPN architecture as option set 1058, DLUR HPR SAW.

Chapter 13. Format Changes

This section describes changes to RUs, control vectors, and GDS variables, for Dependent LU Requester/Server architecture. The actual changes can be found in *Systems Network Architecture Formats*.

13.1 CP-SVR Pipe Format Changes

13.1.1 FID2 Encapsulation (X′1500′) GDS Variable

This is a new GDS variable used to carry SSCP-PU and SSCP-LU sessions on the CP-SVR pipe. This variable includes:

- a field indicating the length of the GDS variable
- a GDS ID of X′1500′
- a field indicating the length of the FID2 PIU
- the entire FID2 PIU (TH, RH, RU (including optional control vectors))
- one or more control vectors
  - a Network Name (X′0E′) control vector (type X′F1′ - PU name)
  - a Security ID Control (X′25′) control vector
  - an Assign LU Characteristics (X′30′) control vector
  - an Extended SDLC Station (X′43′) control vector
  - a TG Descriptor (X′46′) control vector. See 13.1.7, "TG Descriptor (X′46′) Control Vector" on page 13-3 for details about the format changes for this control vector when it specifies the connection between the DLUR and the PU.
  - a new DLUR/S Capabilities (X′51′) control vector
  - a Fully Qualified Procedure Correlation Identifier (FQPCID) (X′60′) control vector

    An FQPCID includes a PCID, a network-qualified CP name, and an optional PCID modifier. The network-qualified CP name is defined as follows:

    - DLUS-generated - the network-qualified SSCP name
    - DLUR-generated - the network-qualified CP(DLUR) name
  - a new XID Image (X′81′) control vector

Table 13-1. GDS X′1500′ Control Vectors		
	term used in flows	**RUs CV appears with**
X′0E′	PU TG	REQACTPU
X′25′		RSP(ACTPU), REQACTPU
X′30′	LU CHAR	ACTLU
X′43′	PU CHAR	ACTPU
X′46′	PU TG	ACTPU, REQACTPU
X′51′	DL CAP	ACTPU, RSP(ACTPU), REQACTPU, RSP(REQACTPU)
X′60′	FQPCID	all RUs
X′81′	XID	REQACTPU, RSP(ACTPU)

13.1.1.1 XID Image (X′81′) FID2 Encapsulation Control Vector

This is a new control vector, locally defined to this GDS variable, which contains either:

- the I-field image of the XID0, XID1 or XID3 SDLC response of the external PU attached to the DLUR, or
- a DLUR-generated XID0 image for an internal PU or for an external PU which does not support XID.

13.1.2 ACTPU (Activate Physical Unit)

The Network Name (X′0E′) control vector (type X′F1′ - PU name) is included with the ACTPU when sent by a DLUS to a DLUR-serviced PU.

There is a network name forwarding request indicator in the PU Capabilities (X′80′) ACTPU control vector which, when set, requests the DLUR to leave the Network Name (X′0E′) control vector in the ACTPU RU when the DLUR forwards the RU to the PU. Also, if the indicator is set, the DLUR is requested to leave the Network Name (X′0E′) control vector in the ACTLU RU for all of the PU′s LUs when the DLUR forwards these RUs to the LUs (for more details about PU/LU network name forwarding, see 6.2.4, “PU/LU Network Name Forwarding” on page 6-25).

13.1.3 INIT-SELF Format 0 (INITIATE-SELF)

When the DLUR node receives an INIT-SELF Format 0 from a DLUR-attached dependent LU, it will create a User Request Correlation (URC) (X′0A′) control vector and include the control vector in the RU as part of name substitution processing (for more details about name substitution processing, see 8.1.3, “INIT-SELF” on page 8-3 and 8.1.4, “BIND” on page 8-3).

13.1.4 REQACTPU (Request ACTPU)

This is a new RU built by the DLUR and sent to the SSCP in the DLUS requesting that an ACTPU be sent to establish an SSCP-PU session with a DLUR-attached PU. This PU could be predefined or undefined to the SSCP ; it could be internal to the DLUR or downstream of the DLUR.

The REQACTPU has two formats:

- Format X′0′ - external PU

- Format X′1′ - internal PU

A positive response to REQACTPU indicates that the SSCP will send an ACTPU to the PU specified in the REQACTPU′s corresponding FID2 Encapsulation (X′1500′) GDS variable. A negative response to REQACTPU indicates that no ACTPU will be sent, and sense data will be included to specify the reason.

13.1.5 REQDACTPU (Request DACTPU)

This is a new RU built by the DLUR and sent to the SSCP in the DLUS requesting that a DACTPU be sent to take down an active SSCP-PU session with a DLUR-attached PU.

The REQDACTPU specifies which PU′s SSCP-PU session is to be deactivated as well as the cause for the request, e.g.:

- X′00′: loss of connectivity from the DLUR to the downstream PU
- X′01′: REQDISCONT(normal) received from the downstream PU
- X′02′: REQDISCONT(immediate) received from the downstream PU

A positive response to REQDACTPU indicates that the SSCP will send a DACTPU to the PU specified in the REQDACTPU′s corresponding FID2 Encapsulation (X′1500′) GDS variable, except for cause X′01′, where the DACTPU may be preceded by LU-LU and SSCP-LU session deactivation commands. A negative response to REQDACTPU indicates that no DACTPU will be sent, and sense data will be included to specify the reason.

13.1.6 Network Name (X′0E′) Control Vector

The Network Name (X′0E′) control vector (type X′F1′ - PU name) is optionally network-qualified. The DLUS includes a network-qualified PU name when the PU′s NETID is different from the SSCP′s NETID.

13.1.7 TG Descriptor (X′46′) Control Vector

When this control vector is included in a FID2 Encapsulation (X′1500′) GDS variable, the TG Identifier (X′80′) subfield is not included since the link being described by the CV in this case is not a TG but a boundary link.

A new subfield (X′91′) has been created to identify the DLC or physical interface type of a PU internal to or externally attached to the DLUR. Information carried in this subfield is currently defined for:

- Subfield X′91′ - DLC and physical interface types:
  - Internal PU
  - Token-Ring LAN
  - Ethernet (IEEE 802.3) LAN
  - Frame relay PVC
  - SDLC nonswitched
  - SDLC switched
  - V.25 switched physical interface
  - V.25bis switched physical interface

– X.25 PVC

– X.25 SVC

Additional DLC and physical interface types will be included at a later time.

New subfields in the range X′92′-X′F0′ are being defined to carry the signaling information, e.g., addresses and dial digits, that is appropriate for each DLC and physical interface type. At present, subfields X′92′ to X′96′ have been defined.

13.1.8 DLUR/S Capabilities (X′51′) Control Vector

This is a new control vector which describes the capabilities of the DLUS or DLUR to its partner.

For this, the initial level of DLUR/S support, there is:

- a one byte release field indicating Release 1 DLUR/S support (X′01′). Future releases will have their own values to aid in uplevel/downlevel support determination.
- a one byte indicator specifying whether the node is a NN (X′00′) or an EN (X′01′).
- a one byte indicator specifying whether the node is a DLUR (X′00′) or a DLUS (X′01′).
- a four byte Flow Reduction Sequence Number (FRSN)
- a one byte support indicator field, including:
  - an indicator for support of the RECEIVE_TDU_TP (X′22F0F0F4′) service transaction program - when set, the sending node supports receipt of TDUs on this session
  - an indicator for limited DLUR support of ANS - when set, the DLUR only supports ANS=STOP; when not set, the DLUR supports both ANS=CONT and ANS=STOP
  - an indicator for limited DLUR multi-subnet support - when set, the DLUR responds to dependent LU searches with the OCR bit set; when not set, the DLUR responds to dependent LU searches with the OCR bit set
  - an indicator for network name forwarding support - when set, the DLUR will, if requested by the DLUS, forward Network Name control vectors on ACTPUs and ACTLUs to DLUR-served PUs and their associated LUs; when not set, the DLUR will always remove Network Name control vectors from ACTPUs and ACTLUs for DLUR-served PUs and LUs (for more details about PU/LU network name forwarding, see 6.2.4, "PU/LU Network Name Forwarding" on page 6-25)
  - an indicator for nondisruptive DLUS-DLUR session deactivation type X′08A0 000B′ support - when set, the DLUR will process an UNBIND with X′08A0 000B′ as described in 5.6.2.1, "UNBIND-Initiated CP-SVR Pipe Deactivation" on page 5-30; when not set, the DLUR cannot process this type of UNBIND
  - an indicator for FID2 Encapsulation (X′1500′) GDS variable blocking support - when set, the DLUR can, if also supported by the DLUS, send and receive BIUs on the CP-SVR pipe which include more than one GDS X′1500′ (for more details about FID2 Encapsulation blocking, see 6.1.5, "Blocking Multiple FID2 Encapsulations" on page 6-7)
  - an indicator for support of CP-SVR pipe persistence - when set, the sender supports CP-SVR pipes which are active even without any associated pending or active SSCP-PU sessions; in addition, if the sender is a DLUR, the sender requests persistence for this particular CP-SVR pipe (for more information about persistent CP-SVR pipes, see 5.8, "Persistent CP-SVR Pipes" on page 5-75)
- four reserved bytes (for future use)

13.1.9 DLC XID Information Field

A new XID3 bit (byte 15, bit 1) has been defined to be the DLUR ACTPU Indicator. This bit will have significance when the ACTPU Suppression Indicator is reset (signifying an ACTPU is requested of the adjacent node). Setting the new bit will signify that the DLUR would prefer to receive the ACTPU over the CP-SVR pipe; resetting the new bit signifies no preference (received either on the TG dependent flow or encapsulated over the CP-SVR pipe).

A new bit, the DLUS-served LU registration indicator, is defined in format 3 of the XID I-field. This bit is only used by a NN. When it is used, setting the indicator signifies that the NN supports DLUS-served LU registration (option set 1116). Resetting the indicator signifies that NN does not support DLUS-served LU registration.

13.1.10 Register Resource (X′12C3′) GDS Variable

13.1.10.1 Command Parameters (X′80′) Register Control Vector Subfield

13.1.10.1.1 DLUS-Served LU Indicator: This is a new indicator to specify whether or not the resources being registered are DLUS-served LUs.

13.2 Multi-Subnet Session Format Changes

13.2.1 Locate (X′12C4′) GDS Variable

13.2.1.1 Intersubnetwork Search (X′82′) Locate Control Vector Subfield

13.2.1.1.1 DLUR Search Required (DSR) Indicator: This is a new indicator to specify whether or not a search to obtain the DLUR TGVs is required.

13.2.1.1.2 Prevent Subarea Path (PSP) Indicator: This is a new indicator to specify whether or not to prevent the EBN from resetting the Suppress Subarea Search indicator.

13.2.2 Find Resource (X′12CA′) GDS Variable

13.2.2.1 Command Parameters (X′80′) Find Control Vector Subfield

13.2.2.1.1 DLUS-Served LU (DSL) Indicator: This is a new indicator to specify whether or not the search origin LU is a dependent LU served by a Dependent LU Server.

13.2.2.1.2 Owning CP Respond (OCR) Indicator: This is a new indicator to specify whether the DLUS or the DLUR should respond to the Locate-Find.

13.2.3 Found Resource (X′12CB′) GDS Variable

13.2.3.1 Command Parameters (X′80′) Found Control Vector Subfield

13.2.3.1.1 DLUS-Served LU (DSL) Indicator: This is a new indicator to specify whether or not the search destination LU is a dependent LU served by a Dependent LU Server.

13.2.3.1.2 Owning CP Respond (OCR) Indicator: This is a new indicator to specify whether the DLUS or the DLUR has responded to the Locate-Find.

13.3 Dependent LU Session Awareness Format Changes

13.3.1 SESSST (Session Started)

There is a new format, X′03′, defined in byte 3 of the SESSST RU. Comparing format X′03′ (which is built by the DLUR) to format X′01′ (which is built by the BF-NCP),

- The DLUR is not aware of the network address of the dependent LU′s LU-LU session partner, so no session key is included in format X′03′.
- The DLUR is not aware of ER or VR information, so the VR-ER Mapping Data (X′1E′) control vector is not included in format X′03′.
- The DLUR is not aware of XRF information, so the Related Session Identifier (X′28′) and the XRF/Cryptography (X′68′) control vectors are not included in format X′03′.
- For network management, specifically problem determination, the DLUR includes the BIND Image (X′31′) control vector; it includes the original BIND image from the PLU, not including any modifications made by the DLUR.
- Also for problem determination, the DLUR includes the Route Selection (X′2B′) control vector; it is obtained from the BIND request from the PLU.
- To return name information, the Session Information (X′2A′) control vector is included in format X′03′, replacing the Local-Form Session Identifier (X′23′) and the Fully-Qualified PCID (X′60′) control vectors, which are already included in the X′2A′ control vector (see 13.3.4, “Session Information (X′2A′) Control Vector” on page 13-8).

Therefore, following the format byte, format X′03′ includes only the Session Information (X′2A′) , the Route Selection (X′2B′), and the BIND Image (X′31′) control vectors.

13.3.2 SESSEND (Session Ended)

There is a new format, X′3′, defined in byte 3, bits 0-3 of the SESSEND RU. Comparing format X′3′ (which is built by the DLUR) to format X′2′ (which is built by the BF-NCP),

- The DLUR is not aware of the network address of the dependent LU′s LU-LU session partner, so no session key is included in format X′3′.

Therefore, following the format byte, format X′3′ includes the cause byte, a retired byte, and, conditionally, the Extended Sense Data (X′35′) and the Fully-Qualified PCID (X′60′) control vectors.

13.3.3 RSP(ACTLU)

The RSP(ACTLU) RU, when built by a DLUR, has several changes to it:

- The DLUR is not aware of XRF information, so the Related Session Identifier (X′28′) and the XRF/Cryptography (X′68′) control vectors are not included in the RSP(ACTLU) processed by a DLUR.
- The Session Information (X′2A′) control vector built and included in the RSP(ACTLU) by the DLUR is different (see 13.3.4, “Session Information (X′2A′) Control Vector” on page 13-8 for details).

Therefore, following the FM profile, RSP(ACTLU), when processed by a DLUR, includes only the SSCP-LU Session Capabilities (X′00′), the LU-LU Session Services Capabilities (X′0C′), and the modified Session Information (X′2A′) control vectors.

13.3.4 Session Information (X′2A′) Control Vector

The Session Information (X′2A′) control vector, when built by a DLUR, has several changes to it:

- The DLUR is not aware of the network address of the dependent LU′s LU-LU session partner, so the Fully-Qualified Network Address Pair (X′15′) control vector is not included.
- The DLUR is not aware of ER or VR information, so the VR-ER Mapping Data (X′1E′) control vector is not included.

Therefore, when built by a DLUR, the Session Information (X′2A′) control vector includes only:

- the Network Name (X′0E′) control vector - it includes the original NS PLU name that appeared in the BIND from the PLU,
- the Local-Form Session Identifier (X′23′) control vector - it includes the LFSID which flows on the BIND from the DLUR to the SLU (stage 4 in Figure 6-2 on page 6-6),
- and the Fully-Qualified PCID (X′60′) control vector.

Appendix A. Alternate Solutions

A.1 LU-LU Session Route Calculation

As an alternative to the modifications of the associated resource hierarchy and TGVs indicated in Chapter 8, “LU-LU Session Routing” on page 8-1, the DLUS node could indicate the DLUR node to be an EN served by the DLUS node with ***dummy TGVs*** when the DLUR is, in fact, an NN. This would be done to avoid any incompatibilities with existing APPN PLU nodes that cannot accept a resource hierarchy of NNCP, ENCP with no TGVs. In this case, (when the DLUR node is an NN), SS at the CP(PLU) will send to TRS a Request_Route message indicating the origin node (CP(PLU)), the destination node (DLUR), and the a dummy tail vector (received on the associated resource hierarchy from the DLUS node) pointing back to the DLUR node. This dummy tail vector will specify a TG number of 1 and the partner CP name will be the DLUR node. TRS will then believe DLUR to be an EN connected to only one NN whose name happens to be the CP name of the DLUR node. This being the case, TRS will compute a route to the DLUR node (thinking it is an adjacent NN) and then use the TG number from the dummy tail vector and change the CP name to the destination node (which is also the DLUR node). So the resultant RSCV will specify a route from the origin to the DLUR node with an extra hop using TG number 1 to the DLUR node again. When the DLUR node receives such an RSCV, it will detect that the last hop represents a dummy TGV by a bit specified in the CV X′4680′. It will consequently ignore this hop and route the BIND to the appropriate LU on this node.

Appendix B. Sample Network Configurations

B.1 Sample Network Configuration Number 1

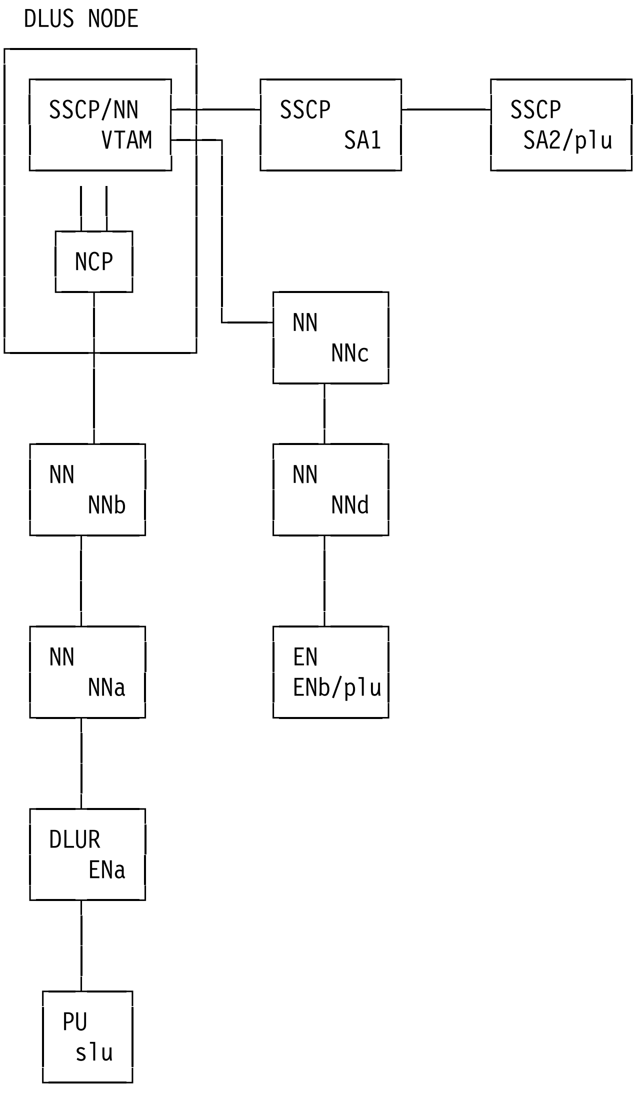


Figure B-1. Sample network configuration number 1

B.1.1 Sample Network Interpretation

In the illustration above, the node functions are listed in the upper left hand corner of the box. The node′s CP name is listed in the lower right hand corner. The VTAM and NCP boxes listed in the upper left hand corner of the diagram represent a composite node image. This composite node presents a single T2.1 NN image to the APPN portion of the network and a subarea image to the subarea portion of the network. This node is an interchange node and is also the Dependent LU Server (DLUS) of the network. Its CP name is VTAM.

The LUs in this network have been named SLU and PLU. The SLU is located on a PU which is physically downstream from the Dependent LU Requester (DLUR) End Node whose CP name is ENa. The PLU has been recorded in two places. The PLU is located in the local domain of SSCP known as SA2. PLU is also located on End Node ENb. Although it is not correct to have two LUs with the same name in a single network, this does not present a problem in the flows described throughout the document because both SA2 and ENb are never both reflected in the network at the same time.

B.2 Sample Network Configuration Number 2

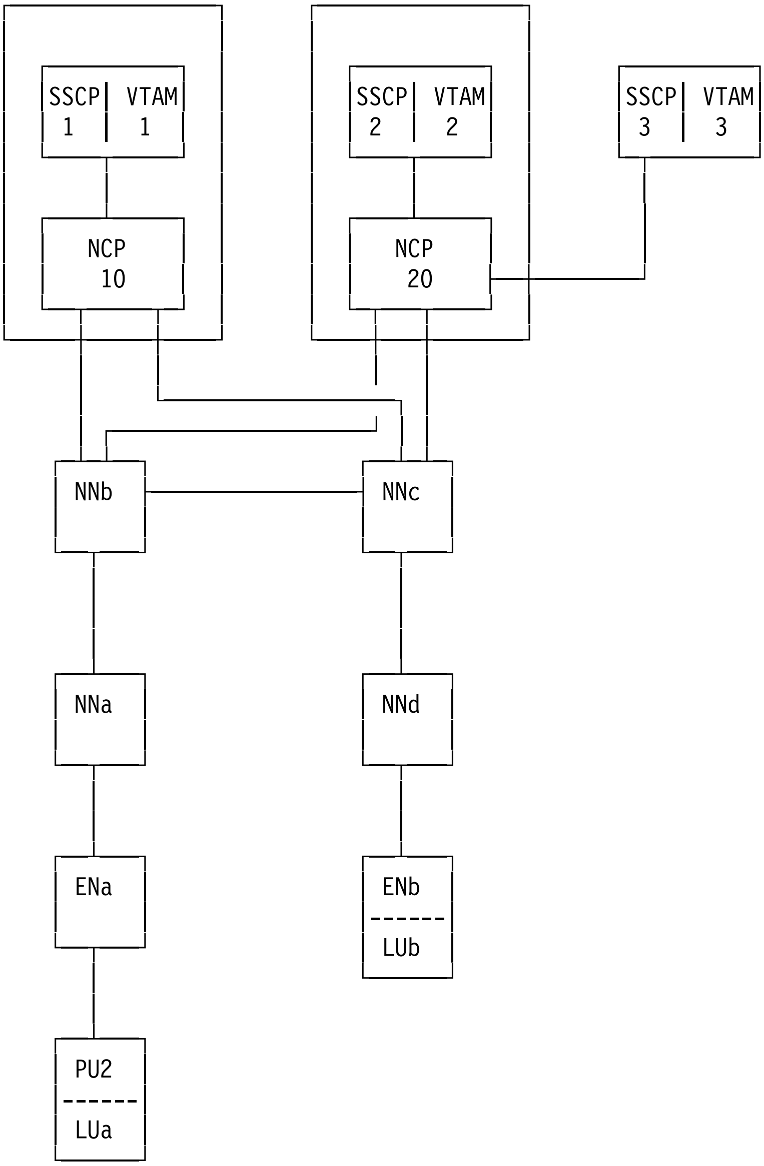


Figure B-2. Sample Network Configuration Number 2

Glossary

For a complete APPN glossary, refer to the *SNA APPN Architecture Reference*.

B

boundary function (BF). (1) In SNA, a capability of (a) a subarea node to provide protocol support for attached peripheral nodes, such as: (i) interconnecting subarea path control and peripheral path control elements, (ii) performing session sequence numbering for low-function peripheral nodes, and (iii) providing session-level pacing support; (b) a DLUR node to provide protocol support for attached type 2.0 nodes. (2) In SNA, the component that provides these capabilities.

C

CP-to-server (CP-SVR) pipe. The two LU 6.2 sessions between a DLUS node and a DLUR node used to carry SSCP-PU and SSCP-LU session flows.

D

dependent LU. See *SSCP-dependent LU*.

dependent logical unit requester (DLUR). An optional component in an APPN end node or network node that supports dependent secondary LUs local to its node or in adjacent type 2.0 or 2.1 nodes by obtaining SSCP services for the LUs over an APPN network from the appropriate dependent LU server (DLUS). The support is obtained using SSCP-PU and SSCP-LU sessions whose flows are encapsulated on LU 6.2 session flows between the DLUR node and the DLUS node. See also *CP-to-server (CP-SVR) pipe*.

dependent LU requester (DLUR) node. An APPN end node or network node that includes the DLUR component.

dependent logical unit server (DLUS). An optional component in an APPN network node that provides SSCP services over an APPN network to remote secondary dependent LUs by using SSCP-PU and SSCP-LU sessions whose flows are encapsulated on LU 6.2 session flows between the DLUS node and the appropriate dependent LU requester (DLUR) nodes. See also *CP-to-server (CP-SVR) pipe*.

dependent LU server (DLUS) node. An APPN network node that includes the DLUS component.

DLUR. Dependent LU requester.

DLUR/S. Dependent LU requester/server.

DLUS. Dependent LU server.

P

peripheral node. A type 1, 2.0, or 2.1 node connected to a subarea boundary node and that uses local addresses for routing and therefore is not affected by changes in network addresses. A peripheral node requires boundary-function assistance from the adjacent subarea node to perform the local-to-network address conversion.

S

SSCP-dependent logical unit. 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.

Index

Communicating Your Comments to IBM

Systems Network Architecture
Advanced Peer-to-Peer Networking
Dependent LU Requester Architecture Reference
Version 1.5

Publication No. SV40-1010-04

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 electronically, use this network ID:

  Internet: **appn@us.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
Advanced Peer-to-Peer Networking
Dependent LU Requester Architecture Reference
Version 1.5

Publication No. SV40-1010-04

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!
SV40-1010-04

IBM®

Cut or Fold Along Line

Fold and Tape | **Please do not staple** | Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation
APPN Architecture
Department BRQA/Building 502
PO BOX 12195
Research Triangle Park, North Carolina 27709-9990

Fold and Tape | **Please do not staple** | Fold and Tape

SV40-1010-04

Cut or Fold Along Line

Printed in U.S.A.

SV40-1010-04