APPN High Performance Routing
===> Draft document <===

Document Number HPR-1

September 24, 1993

Unclassified

Preface

This is the first draft of the APPN High Performance Routing Architecture document. Any questions or comments relative to the contents of this document should be sent to the following Internet address: (to be determined).

Trademarks

The following terms, denoted by an asterisk (*) on their first occurrence in this document, are trademarks of the IBM Corporation in the United States and/or other countries:

- APPN

Contents

Figures

Chapter 1. Introduction

This document describes the High Performance Routing (HPR) architecture. HPR is an enhancement to APPN* that improves its performance and reliability.

It is assumed that the reader is familiar with APPN architecture.

Chapter 2. Requirements

Requirements for new/enhanced functions in APPN:

- Improve APPN data routing

  Provide fast low-level intermediate node routing that minimizes intermediate node storage usage.

- Improve APPN reliability

  Provide greater reliability in case of link and node failures. It should be accomplished transparent to the end users (e.g. humans and applications).

General architecture requirements:

- Functional equivalence

  Provide at least the same level of functional capability in HPR as in APPN. For example, if APPN supports connection networks then HPR does too.

- Migration
  - with existing down-level nodes

    HPR must interoperate with all existing APPN-level nodes.

  - with future SNA nodes

    APPN architecture is migrating towards Gigabit APPN in order to adequately support high-speed networks of the future. HPR should facilitate moving in this direction.

- Minimize costs

  Keep HPR small and simple so as to reduce product development cycle time.

- Easy to manage

  Make it easy to manage combined APPN/HPR network.

Chapter 3. Overview

3.1 HPR Functions

3.1.1 General Description

Rapid Transport Protocol (RTP) connections are established within an HPR subnet[1] and are used to transport session traffic. Session traffic may originate from an APPN or HPR node and, likewise, may be destined for an APPN or HPR node. The physical path utilized by an RTP connection satisifies the Class-of-Service (COS) associated with the session traffic it is carrying. Traffic from many sessions may be carried by a single RTP connection provided they all use the same COS. An RTP connection provides two important advantages:

1. It transports data at very high speeds by using low-level intermediate routing and by minimizing the number of flows over the links for error recovery and flow control protocols. The flows are minimized by performing these functions at the RTP connection end points rather than at each hop (link) along the path.
2. An RTP connection's path may automatically be switched to reroute data around a failed node or link without disrupting the sessions.

HPR can run on existing hardware (thus protecting customer investment) and all HPR features can be provided by a software upgrade.

3.1.2 Functions to provide high-speed data routing:

- Automatic Network Routing (ANR) mode

  ANR routing mode is a low-level routing mechanism that minimizes cycles and storage requirments for routing packets through intermediate nodes. ANR routing is estimated to be 3 to 10 times faster than current APPN routing. No intermediate node storage is required (APPN requires 200-300 bytes per session) and no pre-committed buffers are necessary (APPN recommends pre-committed buffers). See 7.1, "ANR Routing Mode."

- End-to-End Error Recovery

  In the past, error recovery on each link (hop) of a network route has been necessary because of the high link error rates. However, improvements in link error rates have made it feasable and desireable to provide end-to-end error recovery in place of error recovery on each hop. HPR provides this capability by:

  – utilizing existing links and DLCs so that they bypass link-level error recovery

[1] An HPR subnet is a group of contiguously interconnected HPR nodes.

Low error rate links may be operated in non-error-recovery mode. That is, DLCs may be operated such that they do not perform error recovery. See Chapter 6, "HPR Link Support."

- doing end-to-end error recovery on RTP connections

  RTP performs efficient error recovery by retransmitting only those packets that have failed to reach the intended receiver (selective retransmission). See 7.2, "HPR Usage of Rapid Transport Protocol (RTP)."

- End-to-end Flow Control and Congestion Control

  The APPN hop-by-hop window-based flow control protocol (i.e. adaptive session-level pacing) is inadequate for high-speed data routing. HPR uses a protocol suitable for high-speed routing called **Adaptive Rate Based (ARB) Flow/Congestion Control** It regulates the flow of data over an RTP connection by adaptively changing the sender's rate based on feedback on the receiver's rate. This protocol allows for high link utilization and prevents congestion before it occurs, rather than recovering from congestion once it occurs. See 7.2.4.4, "Adaptive Rate-Based Flow/Congestion Control Algorithm."

3.1.3 Functions to improve reliability

- Non-disruptive Path Switch

  A path switch within the HPR portion of the network occurs automatically to bypass link and node failures if an acceptable alternate path is available. It occurs transparently to the sessions (i.e. the sessions will not be disrupted).

3.1.4 Functional Equivalence

Support exists in HPR for the following items in order to maintain functional equivalence with APPN.

- Priority routing

  HPR will provide the capability for higher priority traffic to pass lower priority traffic in intermediate nodes within the HPR portions of the network. The priorities supported are the same as those in APPN.

- Connection Networks

  HPR will provide the capability to do ANR routing across connection networks. The functional capability of connection networks in HPR is the same as in APPN.

3.1.5 Migration

The following features of HPR provide support for the basic architecture requirements.

- Interoperation with existing APPN nodes

  Transforming APPN protocols into HPR and vice versa is done by the **APPN/HPR boundary function** which is described in Chapter 10, "LU-LU Sessions involving APPN nodes (APPN/HPR Boundary Function)." This function provides all the necessary transformations to allow APPN-level nodes and HPR-level nodes to interoperate seamlessly.

- No Configuration Restrictions ("drop in" migration)

  A customer may add new HPR nodes or upgrade existing APPN nodes to HPR in any manner desired. There are no configuration restrictions whatsoever. However the benefits of HPR do not manifest themselves until contiguous clusters of HPR nodes (HPR subnets) are formed.

- Use existing APPN Control Point protocols and algorithms

  In order to preserve existing APPN code and thus minimize product development costs, HPR uses the existing Control Point protocols (directory, topology, CP capabilities, etc.). CP-CP sessions are employed, just as in APPN, to transport these protocols. The APPN route selection algorithm (with minor modification for path switch) is also used by HPR. Using existing APPN control flows significantly reduces the amount of code required for migration (i.e. to implement the APPN/HPR Boundary Function).

- Shared Topology

  All nodes and links are reflected in every APPN and HPR network node's topology to facilitate migration and network management.

3.1.6 Minimize Costs

The size of the HPR delta to APPN has been kept small in order to minimize product development costs.

3.1.7 Easy to Manage

HPR is integrated with APPN so that common network management protocols may be used.

3.2 General HPR/APPN network operation

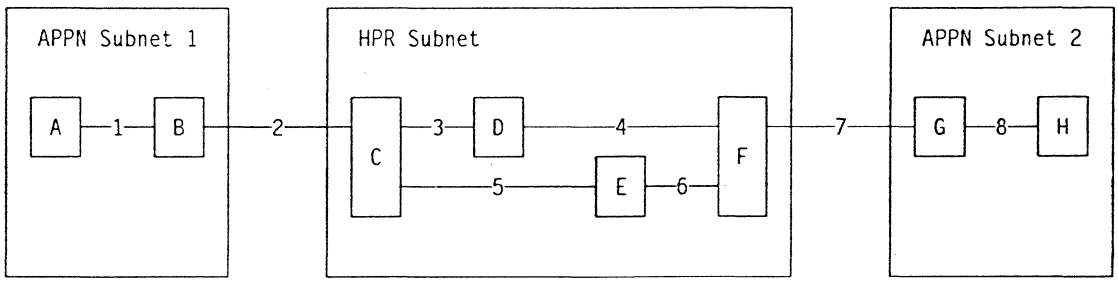


Figure 3-1. APPN/HPR Network

Figure 3-1 shows a network where an HPR subnet connects two APPN subnets. All nodes within the APPN subnets are APPN NN nodes and all nodes within the

HPR subnet are HPR NN nodes. The following sections describe at a high level how this network operates.

3.2.1 Topology

CP-CP sessions are established between adjacent node pairs (in this example pairs AB, BC, CD, CE, DF, EF, FG, and GH) and are used to for broadcasting the topology flows just like in APPN. When all the links and nodes are active, every node has the topology for the entire network stored in it's topology data base. Nodes in the APPN subnets think that nodes and links in the HPR subnet are APPN. Nodes in the HPR subnet can distinguish between the APPN and HPR links and nodes.

3.2.2 Session activation

If an LU in node A wishes to establish a session with an LU in node H the following events occur.

- Node A initiates a directory search to locate the target LU. The directory search protocols are exactly the same as in APPN and flow over the established CP-CP sessions.
- When the search completes, node A computes a route (1-2-3-4-7-8) to node H and sends the BIND over the first hop of the route (link 1). The BIND is sent using the APPN FID2 format.
- When node B receives the BIND it creates a session connector that is used for intermediate session routing (ISR) and adaptive pacing flow control.
- Node B forwards the BIND over link 2 to node C which contains an APPN/HPR Boundary Function (BF). The BF creates a session connector for running APPN protocols over link 2. The BF obtains ANR routing information for path 3-4 and establishes an RTP connection to node F over path 3-4. The BIND is sent over the RTP connection in a Network Layer Packet (NLP)[2] and sent over the RTP connection.
- Since data sent on RTP connections is ANR routed through intermediate nodes, node D simply forwards the NLP to node F over link 4 using fast, low-level ANR routing. Node D maintains no memory for ANR routed NLPs (even when one contains a BIND).
- Node F also contains a BF and when it receives the NLP containing the BIND it creates a session connector for running APPN protocols over link 7. The BIND is sent over link 7 to node G as a FID2 PIU.
- Node G performs normal APPN intermediate processing (just like in node B) and forwards the BIND to node H, the final destination.
- Node H sends back a BIND response which flows using APPN/FID2 protocols over links 8, 7, 2, and 1 and HPR/NLP protocols over the RTP connection that was established over links 3 and 4.

---

[2] All traffic flowing over RTP connections is carried in an NLP. An NLP contains headers for ANR routing and RTP protocols.

3.2.3 Session traffic

After the session has been established, the session traffic flows over links 1, 2, 7, and 8 using APPN/FID2 protocols. It flows over the RTP connection for links 3 and 4 where HPR/NLP protocols are used. Over the RTP connection, error recovery and flow control are performed end-to-end (i.e. between nodes C and F). Intermediate node D does not become involved in these protocols; it only does ANR routing.

3.2.4 Path switching

If a link fails along the HPR portion of the path it may be possible to switch paths. For example, if link 4 fails, the path for the RTP connection will be switched from path 3-4 to path 5-6 (assuming path 5-6 satisfies the COS associated with the sessions). Switching the path involves recalculating a new path and obtaining ANR information about it. Once this is done, traffic on the RTP connection is ANR routed over the new path 5-6 through node E. Node E doesn't know anything about RTP connections or sessions, it just performs ANR routing. Any data that might have been lost due to the link failure will be recovered by the RTP connection end points (in nodes C and F).

Chapter 4. HPR Base and Towers

In order to facilitate implementation across a wide range of products, certain portions of HPR have been designated as optional towers. The towers are shown in Figure 4-1. Many of the functions referred to in this chapter are described in detail in later chapters so don't be concerned if everything in this chapter is not immediately understood.

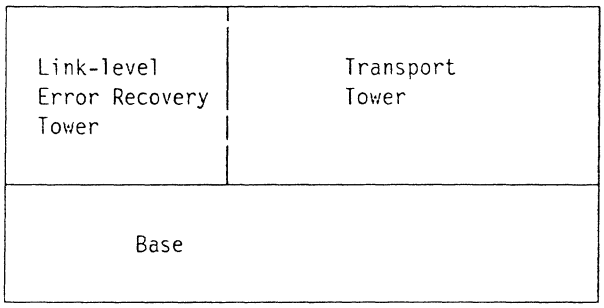


Figure 4-1. HPR Base and Towers

4.1 Base Functions

The primary function of the HPR base is to provide ANR routing. Products that only implement the base can participate as intermediate nodes (that only perform ANR routing) for HPR transport connections. Nodes that do not support the Transport Tower cannot be the end points of RTP connections. The following list summarizes the base functions:

- Intermediate ANR Routing for NLPs

  HPR Network Layer Packets (NLPs) may be efficiently routed, using ANR routing, through the node. The traffic that is ANR routed is that which flows over RTP connections. See 7.1, "ANR Routing Mode" on page 7-1.

- FID2 PIUs used for CP-CP and ISR LU-LU Sessions (i.e. those using APPN session connectors)

  All CP-CP session traffic flows as in APPN using FID2 PIUs. APPN LU-LU session traffic not flowing over RTP connections also uses FID2 PIUs.

- FID2 PIUs and NLPs share link (TG)

  Both FID2 PIUs and NLPs may flow over a single link. They are distinguished by the first 3 or 4 bits in the packet (FID2 packet B'0010', NLP B'110'). See Chapter 5, "HPR Format Overview" on page 5-1.

- No link-level error recovery for NLPs

  Link-level error recovery (i.e. LLC elements of procedure) is not performed for NLPs because the links are reliable and RTP is doing end-to-end error recovery.

However, link-level error recovery is always done for FID2 PIUs to insure reliable transmission. See Chapter 6, "HPR Link Support" on page 6-1.

- Link and node TDUs indicate level of HPR capability

  Link and node TDUs are sent indicating the appropriate level of HPR support. See 8.3.2, "Topology" on page 8-4.

- FID2 Route Setup

  Prior to establishing an RTP connection, a Route Setup protocol is executed in order to obtain the necessary ANR information associated with each link along the desired path. Every node along the path, including base-level nodes, participates by adding the appropriate ANR information. When the Route Setup messages are exchanged between two nodes where one or both are base-level nodes, it flows within a FID2 PIU. See 9.2, "Route Setup protocol" on page 9-2 and A.1.14, "FID2 Route Setup" on page A-30.

- Minimum link size 768 bytes

  The smallest "maximum link size" allowed on any link that supports HPR is 768. This information is exchanged in XID3 just as in today's APPN. See Chapter 6, "HPR Link Support" on page 6-1.

- HPR capability exchanged via XID3

  A new control vector on XID3 indicates the HPR support level. See A.1.9, "HPR capabilities control vector (CV61)" on page A-25.

- HPR only routes

  NN's understand how to calculate HPR only routes. See 9.6.10, "Obtaining a New Path" on page 9-18.

4.2 HPR Transport Tower

Nodes that support the HPR Transport tower are able to transport session traffic across HPR networks over RTP connections thus enabling the use of HPR's high-speed ANR routing and non-disruptive path switch functions. An RTP connection can only be made between nodes that support the HPR Transport Tower so it is essential that there be such nodes in the network. If all the HPR nodes in the network support only the base, there will be no advantages over APPN (in fact, pure APPN protocols will be used). All data flowing over an RTP connection is carried in a Network Layer Packet (NLP). The following functions are included in the HPR Transport tower.

- Rapid Transport Protocol (RTP)

  This is the transport protocol used in HPR for transporting data accross HPR subnets. All RTP functions are supported; there are no optional towers within RTP. See 7.2, "HPR Usage of Rapid Transport Protocol (RTP)" on page 7-3.

- Non-disruptive Path switch

  If the current path being used by an RTP connection fails, it may be switched to a new path automatically. Sessions that are being transported by the RTP connection are not disrupted. See 9.6, "LU-LU Path Switch" on page 9-10.

- RTP connection for CP-CP sessions

  CP-CP sessions established between two nodes where both nodes support the HPR Transport tower use an RTP connection to transport the session traffic. This allows running CP-CP sessions over links that do not do link-level error recovery. See Chapter 8, "CP-CP Sessions" on page 8-1.

- Directory Reply with LU's Network Connection Endpoint (NCE)

  An NCE is part of ANR routing that allows an NLP to be routed to a specific component within a node. The component is uniquely identified by the NCE. (See 7.1, "ANR Routing Mode" on page 7-1). A search reply for an LU contains the NCE associated with the LU. See 8.3.1, "Directory Searches" on page 8-3.

- RTP connection for Route Setup

  A long-lived RTP connection, established over each link when it is activated, is used to transport the Route Setup protocol NLPs. This allows running Route Setup protocols over links that do not do link-level error recovery. See 9.2, "Route Setup protocol" on page 9-2.

- APPN/HPR Boundary Function Support

  APPN (FID2 PIU) traffic is mapped to HPR (NLP) traffic and vice versa. See Chapter 10, "LU-LU Sessions involving APPN nodes (APPN/HPR Boundary Function)" on page 10-1.

4.3 Link-level Error Recovery Tower

Supporting link-level error recovery for NLPs is an optional tower with the purpose of providing error recovery for links with high error rates. Very low error rate links may never require this support. See Chapter 6, "HPR Link Support" on page 6-1.

Chapter 5. HPR Format Overview

There are various types of packets that can flow between HPR nodes. A packet for HPR may contain either an XID3 I-frame, a FID2 PIU, or a Network Layer Packet (NLP). See Figure 5-1.

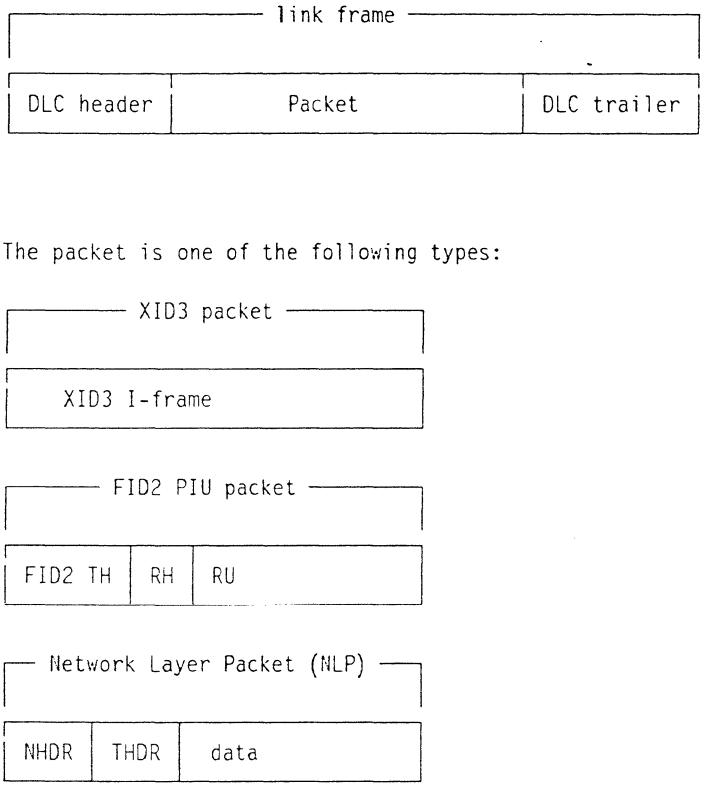


Figure 5-1. HPR formats

- **DLC header**

  The term DLC is used here to mean not only the traditional DLCs (e.g. LAN LLC) but also the DLCs comprising whole networks (e.g. Frame relay).

- **XID3 Packet**

  HPR uses XID3 (as defined in APPN) with the addition of a new CV (CV61).

- **FID2 PIU Packet**

  FID2 PIUs contain a FID2 TH where the first 4 bits of the TH indicate the FID type (i.e. B'0010' indicates FID2). These 4 bits are used to distinguish FID2 PIUs from NLPs. FID2 PIUs are used by HPR to carry the following:

  - CP-CP session traffic between nodes that do not both support the HPR Transport Tower.
  - LU-LU session traffic that is not carried over an RTP connection (i.e. it is transported using today's APPN ISR).

- Route Setup messages between nodes that do not both support the HPR Transport Tower.

- **NLP**

  The first 4 bits of an NLP is never B'0010'. This is so that NLPs and FID2 PIUs can be distinguished. NLPs are used by HPR to carry the following:

  - CP-CP session traffic between nodes that both support the HPR Transport Tower.
  - LU-LU session traffic that is carried over an RTP connection.
  - Route Setup messages between nodes that both support the HPR Transport Tower.

  The NLP consist of the following:

  - **NHDR - Network Layer Header**

    Contains ANR routing information.

  - **THDR - RTP Transport Header**

    Contains information used by RTP.

  - **data**

    For LU-LU or CP-CP session traffic this field contains a FID5 PIU. FID5 is a new TH type developed for HPR that contains an enhanced session address field. For non-session traffic (i.e. Route Setup) it contains the appropriate HPR GDS variables.

- **DLC trailer**

  Contains the frame CRC field which is always present and provides data integrity checking exactly as in APPN. This is the only integrity checking done in HPR so CRC is always required.

Chapter 6. HPR Link Support

Since HPR is an enhancement to APPN it will operate over links used by today's APPN as well as any new high-speed links that may be developed. That means existing hardware adapters and DLCs currently being used for APPN can also be used for HPR.

6.1 DLC Properties Required for HPR

All APPN DLCs (LLCs) have the following properties:

- they send and receive packets (as opposed to stream mode)
- they provide a frame CRC
- they guarantee FIFO delivery

These same properties are required for use with HPR.

6.2 Link activation

For link activation, DLCs are used by HPR in the same manner as APPN. That is, XID3 is exchanged and the appropriate set mode signals are sent when the exchange is complete. For HPR, a new control vector is carried by the negotiation-proceeding XID3 that contains additional HPR related information (see A.1.9, "HPR capabilities control vector (CV61)" on page A-25). CV 61 indicates the level of HPR support (e.g. what towers are supported). If both nodes indicate support for the HPR Transport tower, then HPR Transport Tower protocols are used; otherwise, HPR Base protocols are used (see Chapter 4, "HPR Base and Towers" on page 4-1). All of the XID3 and set mode protocols (including XID3 error recovery) remain the same as in today's APPN.

An HPR capable node may activate links in an APPN way (i.e. not include the HPR CV 61 on the XID) and use them exactly as in today's APPN. Although this is not generally recommended, some customers may wish to continue to run very slow speed links the APPN way because of constraints on link buffer sizes or link speeds.

6.2.1 CP-CP Session Activation

CP-CP session activation is triggered by link activation in the same manner as APPN.

6.2.2 Route Setup Long-lived RTP Connection

Immediately after an HPR link is activated where both sides support the HPR Transport Tower, an RTP connection is activated in order to carry Route Setup messages. This RTP connection remains active as long as the link remains active. See 9.2, "Route Setup protocol" on page 9-2 and 9.2.6.1, "Long-lived RTP Connections" on page 9-4 for further details.

6.2.3 HPR Minimum Packet size

(Note: this discussion is placed here in the Link Support chapter because the information pertains to links. However, many of the concepts have not yet been introduced. Therefore, it is suggested that the reader re-read this section after finishing the rest of the document.)

The minimum "maximum" packet size for an HPR link frame is 768. The NHDR and THDR cannot be segmented, therefore the minimum packet size must accomodate the largest possible (within reason) NHDR/THDR combination. The size of the NHDR depends on the number of hops (links) and the size of the ANR labels (one label represents each hop). The largest possible THDR is the one used when activating an RTP connection.

Assuming a 10-hop route with each label 4 bytes long, the NHDR/THDR maximum size is 481 bytes. A 10-hop with 2-byte labels is 441 bytes. See Figure 6-1 on page 6-3 for details of how these numbers were derived.

A minimum packet size of 512 would satisfy the above but would not leave much room for future expansion. To summarize, HPR will use a mimimum packet size of 768 bytes because of the following.

- It is reasonable and affordable, given memory costs, for HPR-level products to provide this link buffer size.
- 768 allows room for significant expansion.

```
10 Hops, 4-byte labels:

NHDR = 44
THDR = 166
   Base = 16
   Connection Qualifier Field = 32
   RTP Connection Setup Segment = 44
   RTP Adaptive Rate-Based Segment = 20
   RTP Routing Information Segment:
      HPR Routing Information CV = 70 (10 hops, 4-byte labels)
      HPR Return Route TG Descriptor CV = 255

NHDR + THDR = 481
= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
10 Hops, 2-byte labels:

NHDR = 24
THDR = 146
   Base = 16
   Connection Qualifier Field = 32
   Connection Setup Segment = 44
   Adaptive Rate-Based Segment = 20
   RTP Routing Information Segment:
      HPR Routing Information CV = 50 (10 hops, 2-byte labels)
      HPR Return Route TG Descriptor CV = 255

NHDR + THDR = 441

Note: assumes maximum length values for all subfields within the
     Connection Qualifier Field and Segments are assumed.
```

Figure 6-1. Computations for packet size: 10 hops, 4-byte labels

6.3 Link data traffic

Network Layer Packets (NLPs) may be sent over an HPR link using link-level error recovery procedures or not. (Note that when using HPR base protocols, FID2 PIUs always use the link-level error recovery procedures, just as in today's APPN). Whether or not error recovery is used on NLPs is determined during the XID3 exchange via the error recovery indicators in the new HPR CV 61.

- using link-level error recovery (tower)

  NLPs are transmitted in the same manner as in today's APPN where the LLC elements of procedure provide full error recovery. For HPR, this is recommended for links with high error rates.

- not using link-level error recovery (base)

  NLPs are transmitted in a manner such that the LLC will not perform any error recovery for it. This mode of operation is recommended for low error rate reli-

able links. The method for bypassing the link-level error recovery mechanisms varies depending on the link type (see descriptions of each link type below).

Not using link-level error recovery is an HPR base function; using it is a tower function (Link Level Error Recovery Tower). See Chapter 4, "HPR Base and Towers" on page 4-1.

The following sections describe the link types currently supported by HPR. Additional types will be added as necessary.

6.3.1 Frame Relay

The frame relay formats for HPR are based on those described in the "Protocol Encapsulation over Frame Relay Implementation Agreements - Frame Relay Forum Technical Committee TRF 92.32R2" document that is edited by Rao Cherikuri. There are two basic choices as to how to bypass link-level error recovery. The first is to use the frame relay header as currently defined for APPN FID2 and use the Unnumbered Information (UI) frame (also referred to as Type I data) in the 802.2 header. The second is to not include the 802.2 header at all. The second method has been chosen based on product performance. Not invoking 802.2 LLC at all was better than invoking it and doing UI processing. See the following frame relay formats for details.

- A.2.1, "Frame Relay format for XID3 and FID2 Route Setup" on page A-34
- A.2.2, "Frame Relay format for HPR NLPs when not using link level error recovery" on page A-35
- A.2.3, "Frame Relay format for HPR NLPs when using link level error recovery" on page A-36

6.3.2 LANs

HPR needs to be able to bypass error recovery for NLPs on LANs using 802.2. Many options were considered and the one chosen (primarily based on product performance) was to have a new SAP that is used to transmit NLPs with no error recovery. NLPs requiring error recovery would use the current APPN SAP. See the formats in the appendix for details.

- A.2.4, "Token Ring LLC 802.2 format for HPR NLPs when not using link level error recovery" on page A-37
- A.2.5, "Token Ring LLC 802.2 format for XID, FID2 PIUs, and HPR NLPs when using link level error recovery" on page A-37

6.3.3 SDLC

The Unumbered Information (UI) capability in SDLC is used to send data without performing link-level error recovery on it. Therefore, HPR NLPs that do not want link-level error recovery are sent as UI frames. See "Synchronous Data Link Control Concepts" - GA27-3093 for description of this capability. HPR NLPs that do want link-level error recovery are sent as normal SDLC information frames.

6.3.4 X.25

The QLLC option of X.25 is used when running HPR traffic over an X.25 link (circuit).

All the links through an X.25 network as well as the access links (DTE-DCE) provide link-level error recovery (this includes CRC checking). There is no way to "turn off" this error recovery since it's totally under the control of the X.25 network. The X.25 DTE's have a choice of running either QLLC or ELLC. ELLC provides an additional layer of error recovery which is done end-to-end (between the DTEs) and operates on top the already existing link-level error recovery. QLLC does not provide any additional error recovery; it relies on the underlying X.25 link-level error recovery.

All products supporting X.25 support the QLLC option (it is base level function) but ELLC is optional and, in fact, is not widely supported.

6.4 Link failure detection when not using error recovery

Even when link-level error recovery is not being done, it is still necessary to detect link outages. This is done using the link "inactivity" timer. The time for outage detection must be faster than the end-to-end RTP connection timeouts (in order for path switching to work properly) so the link "inactivity" timer value be adjusted accordingly so that TDUs are sent quickly. There are actually 3 link parameters that govern how long it will take to determine that the link has failed.

1. "inactivity" timer
2. "send" timer

   The timer used while waiting for acknowledgment to sent data.

3. number of retries

   The number of times data will be resent.

It is recommended that these link parameters be set as low as possible for HPR since the need for sending TDUs and doing path switches is time critical. These parameters must be exposed to the customer so they can be tuned appropriately based on the network environment. When a new path is calculated it is assumed that the topology database closely reflects the current states of the links. The topology database is kept current by the prompt sending of TDUs indicating the status of the links. Therefore, TDUs indicating link outages must be sent on a timely basis.

6.5 Limited Resource Link Deactivation

In today's APPN, a limited resource link is deactivated automatically when no sessions are using it. This is possible because all nodes (end and intermediate) have session awareness.

HPR end nodes have session awareness and can deactivate links based on session usage. However, HPR intermediate nodes have no session awareness so cannot

deactivate links automatically based on session usage. Of course, a system or node operator can always deactivate any link but this may require manual intervention. In HPR a limited resource link is automatically deactivated when no known sessions are using it (there could be FID2 sessions running over the link that have session connectors or half-sessions) AND no traffic has traversed the link for a specified period of time which is governed by the "limited resource link deactivation timer". The value used for this timer is coordinated with the value of RTP ALIVE timer as described in 7.2.4.1.3, "Timers" on page 7-14.

Chapter 7. Common Session Support (CP-CP and LU-LU)

7.1 ANR Routing Mode

HPR uses the ANR routing mode to route session traffic (including BINDs and UNBINDs) through an HPR network between nodes supporting the HPR Transport tower. Every packet using the ANR routing mode contains a network header (NHDR); the NHDR contains an ANR routing field.

The ANR routing field is composed of a series of ANR labels identifying the packet's path through the network. The last ANR label identifies the internal component in the destination node that is to receive the packet. In this document, the component receiving the packet is called the network connection endpoint (NCE). The other (preceding) ANR labels identify the intermediate links (TGs). As a packet flows through the network, the ANR labels are removed from the packet's ANR routing field as they are used. When an intermediate node receives a packet it examines the *first* ANR label to determine where to route the packet and removes this label prior to forwarding the packet. Because an ANR label is removed at each intermediate node, the size of the NHDR decreases as the packet flows through the network.

See A.1.4, "Network Layer Header (NHDR)" on page A-5 for more information on ANR routing.

7.1.1 Ending Delimiter

The end of the ANR routing field is indicated with a delimiter of X'FF'. This means no ANR label may contain a X'FF'.

7.1.2 ANR Label Assignment

There are two ANR labels associated with each link. When a link is activated, an ANR label is assigned by each node for its outbound direction. In Figure 7-1, node A assigns label X'D2' for the direction from A to B, and node B assigns label X'94' for the direction from B to A. Each ANR label assigned within a node (either for a link or an internal component) is unique. For example, if a node activates 25 links and has 5 NCEs, it assigns 30 unique ANR labels. The high-order (leftmost) bit of each ANR label is reserved for future use and always set to 1. For example, the label X'8103' is valid, but the label X'0103' is not.

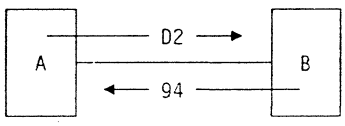


Figure 7-1. ANR Label Assignment

7.1.3 ANR Label Size

The size of ANR labels can vary from one to eight bytes, but they should be kept as small as possible because of the space they use in the NHDR. A typical size is one or two bytes. Sizes may vary even for labels assigned within a single node; however, when a node receives a packet and examines the first label in its ANR field (which was assigned by receiving node), it must be able to unambiguously determine which resource (link or internal component) that label identifies. For example, a node cannot have one link with a label of X'83' and another link with a label of X'8345'.

7.1.4 NCE Labels

The last label in the ANR routing field identifies an NCE within the destination node. NCEs include control points (CPs), LUs, boundary functions (BFs), and Route Setup (RS) components.

7.1.4.1 CP NCEs

Each node assigns a label for its CP. Adjacent nodes exchange their labels (referred to as NCE_CPs) during link activation (on XID3). All CP-CP session traffic is sent with an ANR routing field containing the NCE_CP of the destination node. Any packet received with the NCE_CP as the only label in the ANR field is internally routed to the CP. See Chapter 8, "CP-CP Sessions" on page 8-1 for more information.

7.1.4.2 LU NCEs

When the destination LU is located in a node that supports the HPR Transport tower, LU-LU session traffic is sent with an ANR routing field whose last label identifies the LU; such a label is called an NCE_LU. Any packet received containing an NCE_LU as the only label in the ANR routing field is internally routed to the appropriate LU. The NCE_LU identifies the component within the node that processes all packets received for that LU.

See Chapter 9, "LU-LU Sessions (HPR-HPR)" on page 9-1 and Chapter 10, "LU-LU Sessions involving APPN nodes (APPN/HPR Boundary Function)" on page 10-1 for more information about NCE_LU usage.

7.1.4.3 Boundary Function NCE (NCE_BF)

An NCE_BF is used to identify a component within a node that performs the HPR to APPN boundary function transformation. Such function is required for links to APPN nodes or HPR nodes without the Transport tower.

See Chapter 10, "LU-LU Sessions involving APPN nodes (APPN/HPR Boundary Function)" on page 10-1 for more information about NCE_BF usage.

7.1.4.4 Route Setup NCEs

HPR employs a route setup protocol in order to obtain ANR and RTP connection information. The component within the node that processes Route Setup messages is identified by a label called the NCE_SR. NCE_SRs are exchanged when links are activated. Chapter 9, "LU-LU Sessions (HPR-HPR)" on page 9-1 contains a complete description of the route setup protocol.

7.1.5 ANR Routing Example

Figure 7-2 shows an example of a Network Layer Packet being routed through a series of nodes. The packet consist of three parts; NHDR, THDR, and DATA. The NHDR contains the ANR routing field (ANRF). The ANR routing field is shown as a series of ANR labels separated by "-"s. The X'FF' delimiter is present but not explicitly shown in the sequences. The example shows a packet being sent between two LUs. The LU in node A has an NCE_LU of X'A2'. The LU in node D has an NCE_LU of X'94'. Note that the ANRF becomes shorter as ANR labels are removed at each intermediate node.

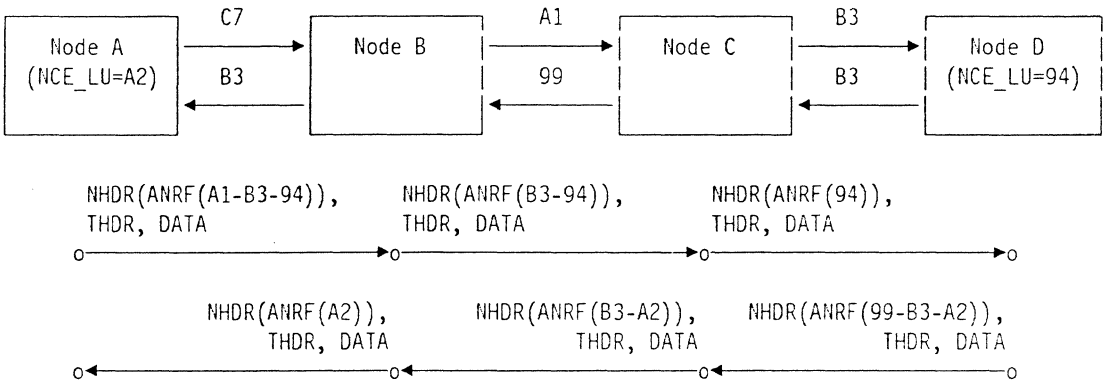


Figure 7-2. ANR routing example

7.1.6 Priority Routing in Intermediate Nodes

The NHDR of each NLP contains a Transmission Priority field (see A.1.4, "Network Layer Header (NHDR)" on page A-5) specifying one of the four transmission priorities: network, high, medium, and low. The priority function is implemented in intermediate nodes of the ANR path via the use of priority queues. Thus, higher-priority NLPs may pass lower-priority NLPs when being routed through an intermediate node. Products may use "aging" algorithms to ensure that lower-priority traffic is eventually transmitted; such algorithms are not architecturally defined.

7.2 HPR Usage of Rapid Transport Protocol (RTP)

RTP is a connection-oriented, full-duplex protocol designed to transport data in high-speed networks. HPR uses RTP connections to transport LU-LU and CP-CP session traffic. RTP provides reliability (i.e., error recovery via selective retransmission), in-order delivery (i.e., a first-in-first-out [FIFO] service provided by resequencing data that arrives out of order), and adaptive rate-based (ARB) flow/congestion control. Because RTP provides these functions on an end-to-end basis, it eliminates the need for these functions on the link level along the path of the connection. The result is improved overall performance for HPR.

For data that does not require reliability but does require in-order delivery and flow control, RTP provides a no-retry mode of operation on a connection basis. In this mode, RTP provides the other functions but does not retransmit lost data.

The following is a brief summary of the RTP functions:

- Reliable mode of operation

  In this mode of operation, RTP reliably delivers user data in both directions on the connection and notifies the sending RTP user upon the successful arrival of the data at the other RTP connection endpoint. RTP selectively retransmits the user data in any lost or errored packets; user data that successfully arrives out of order at the other connection endpoint is buffered so that its retransmission is not required.

- No-retry mode of operation

  For a connection using this mode of operation, RTP does not retransmit the user data in lost or errored packets but does notify the receiving RTP user when data is lost. The other RTP functions are provided in this mode of operation.

- ARB flow/congestion control

  The ARB flow/congestion control mechanism protects both the buffers at the endpoints of the connection and the network resources. See 7.2.4.4, "Adaptive Rate-Based Flow/Congestion Control Algorithm" on page 7-21.

- Segmentation and reassembly

  RTP will perform the necessary segmenting and reassembly based on the minimum link BTU size on the path between the two endpoints of the RTP connection.

- Connection maintenance

  When an RTP endpoint does not receive a packet from its partner within a specified period of time, it will send a "liveness" message to its partner to ensure that the connection is still active.

7.2.1 Relationship of Sessions, COS, and RTP Connections

RTP connections are used to transport session data between HPR nodes (with the HPR Transport tower) operating within an HPR network. An RTP connection has the following properties:

- Full-duplex logical connection

  RTP provides a full-duplex logical connection between two HPR nodes over a specific path through the HPR network.

- Single COS per connection

  Each RTP connection transports session data for a single COS (specified in the BIND). An RTP connection is not used for more than one COS in order to simplify the path switch algorithm. When a path switch occurs the new path must satisfy the COS. If an RTP connection were to support multiple COSs and its path were to fail, under certain conditions the following would be required:

  1. The computation and setup of multiple new paths (one for each COS)
  2. The activation of new RTP connections over these paths

3. The splitting of the sessions of the original RTP connection among the new set of RTP connections

Because of this complexity, only sessions of a single COS are transported across an RTP connection.

- Multiple connections per COS

  If node *X* wishes to send a BIND to node *Y*, and no existing RTP connection between the nodes[1] for the session's COS can be used, node *X* initiates an RTP connection to node *Y* and sends the BIND over the new connection. Node *X* may send additional BINDs (assuming they are for the same COS) over this connection.

  If node *Y* wishes to send a BIND to node *X* over the same path, it uses the existing connection established by node *X*; otherwise, it activates a separate RTP connection. If node *Y* uses the connection established by node *X*, it can send the BIND over that connection immediately. In node *X*, connections established by node *X* are called outbound connections, and connections established by node *Y* are called inbound connections.

  A node may (optionally) activate multiple RTP connections (each over a different path) to the same partner for the same COS in order to distribute the traffic among the links of the various paths.

  The node that activates an RTP connection is responsible for initiating its deactivation when there are no sessions using it (or about to use it). But if its partner refuses to bring the connection down because it has sessions using the connection, deactivation responsibility is transferred to the partner. This protocol is described in 7.2.3, "RTP Connection Deactivation" on page 7-9. The deactivation protocol is necessary for situations in which one node attempts to deactivate a connection while its partner node is sending a BIND over the same connection.

All traffic from one session flows over a single RTP connection, but many sessions may be multiplexed onto one RTP connection. All sessions between two nodes[1] that have the same COS and use the same path are transported over a single RTP connection between the HPR nodes containing the session endpoints (or BFs). Figure 7-3 on page 7-6 shows the relationship between half-sessions (or session connectors in the BF case) and RTP connections.

---

[1] Actually, the RTP connection is established between an NCE in node *X* and an NCE in node *Y*. Only sessions flowing between these two NCEs may be multiplexed on the RTP connection. This section assumes an implementation with a single NCE in each node. The reader should understand that for implementations with multiple NCEs, the term "NCE" should be substituted for "node."

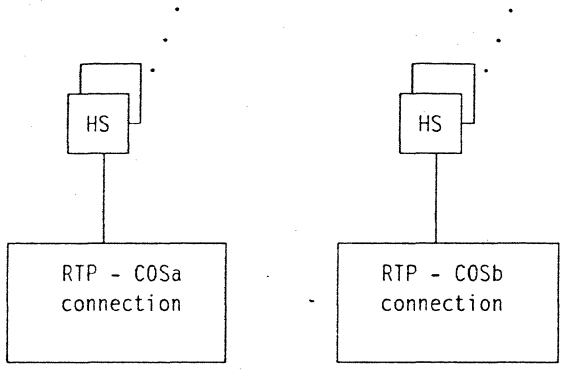


Figure 7-3. Relationship between sessions and RTP connections for different COSs

7.2.2 RTP Connection Activation

Figure 7-4 shows the RTP connection setup protocol. An RTP connection is activated when a node[1] wishes to activate a session and no RTP connection exists for the path selected by the route computation function. Note that the path information (i.e., forward path ANR labels and reverse path ANRs labels, etc.) are provided by the preceding route-setup procedure: please refer to Chapter 9, "LU-LU Sessions (HPR-HPR)" on page 9-1 for details on the route-setup flows. Please see Appendix B, "Sequence Notation" on page B-1 for a description of the notation used in the following sequences.

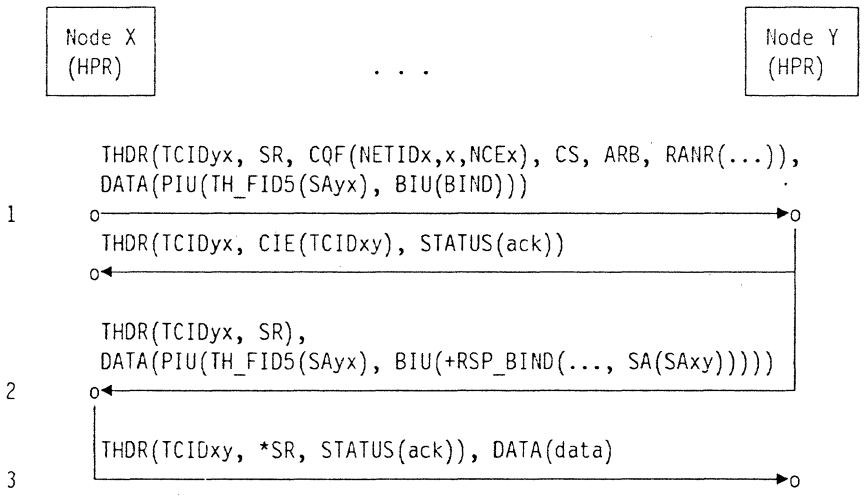


Figure 7-4. RTP connection activation

Annotations:

1 Node *X* assigns TCIDyx which is the TCID to be used on all packets sent from Node *Y* to Node *X* over this RTP connection. The RANR field contains an ANR path from node *Y* back to node *X* which is used to send packets in the reverse direction. The BIND is carried in the DATA field of the packet. Status (acknowledgment) is requested.

Note that for activation of a no-retry connection, the RETRYI (retry indicator) bit in the RTP header is set to ¬RETRY. No packets that belong to a no-retry connection are retransmitted when lost.

A packet with a Status segment and a Connection Identifier Exchange (CIE) segment is returned immediately. The CIE segment is used to return TCIDxy which is the TCID to be used in all packets sent from Node *X* to Node *Y* over this RTP connection. An acknowledgment (Status segment) is sent to acknowledge the successful receipt of the data (BIND); it also acknowledges implicitly that the Connection Setup segment has been received and processed.

2 The RSP(BIND) is carried as data and an acknowledgment is requested (SR).

3 Node *X* acknowledges the receipt of the RSP(BIND) and acknowledges implicitly the CIE segment. An RTP connection is now fully active. This connection can be used by both nodes to send additional BINDs.

Figure 7-5 on page 7-8 shows a scenario where a BIND is followed immediately by an UNBIND due to some error detected at node *X*. In this case, the UNBIND is sent as data across the RTP connection during its activation procedure.

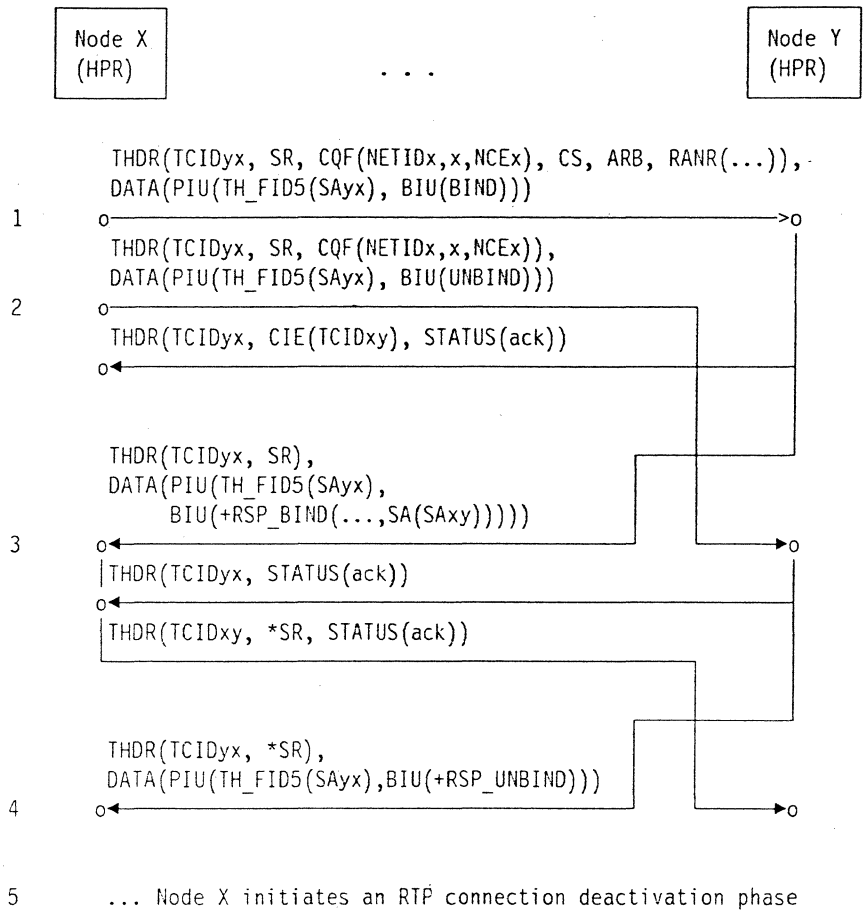


Figure 7-5. RTP connection activation (BIND is followed by an UNBIND)

Annotations:

1 Node *X* initiates the setup of an RTP connection as the result of a BIND as described in the previous flow.

2 Node *X* detects an error and decides to deactivate the session; it sends UNBIND to node *Y*. The UNBIND is sent as data by RTP across the connection which is being established. Because the CIE segment has not yet been received by node *X*, the UNBIND is sent in a packet with a Connection Qualifier field and the first bit of the TCID set to 1. Meanwhile, node *Y* sends a packet containing a CIE segment and a Status segment to node *X*. The acknowledgment (Status segment) is sent to acknowledge the successful receipt of the data (BIND); it also acknowledges implicitly that the Connection Setup segment has been received and processed. Then, node *Y* finishes processing the BIND and prepares a RSP(BIND) to send back to node *X*.

3 Node *Y* receives the UNBIND, starts terminating the session, and immediately returns a Status segment to acknowledge receipt of the data (UNBIND). Meanwhile, the RSP(BIND) is received by node *X* which is expecting the subsequent RSP(UNBIND). An acknowledgment is requested (SR) for the data (RSP(BIND)).

4 Node *X* acknowledges the receipt of the RSP(BIND) and acknowledges implicitly the CIE segment. Meanwhile, node *Y* returns a RSP(UNBIND).

5 The RTP connection is active; however, node *X* detects that there is no session using this connection. As a result, it initiates RTP connection deactivation. Although not shown, node *X* RTP will acknowledge the data (RSP(UNBIND)).

7.2.2.1 TCID Uniqueness

An RTP process assigns a unique TCID for each connection it either initiates or for which it sends an CIE segment. A node may have multiple RTP processes, each residing in separate, independent processors. Each processor is associated with an NCE, and an arriving packet is passed to the appropriate RTP process based on the ANR label for that NCE in the packet's transport header. Thus, each TCID associated with a label for an NCE is unique, and as previously described, all the ANR labels within a node are unique.

TCIDs are not required to be unique within a node. Two (or more) RTP processes within a node may assign the same TCID. Of course, a product that has a structure with a single RTP process, will assign TCIDs that are unique within its nodes.

7.2.3 RTP Connection Deactivation

In this section, the protocol used to deactivate an RTP connection that carries SNA session traffic is described. The protocol handles a situation in which one RTP endpoint attempts to deactivate a connection because no session is using the connection while the other RTP endpoint is sending a BIND for a new session. This can happen because both RTP endpoints of a connection may send BINDs for new sessions. If an RTP connection is used to carry non-SNA traffic, its endpoints are not required to follow this protocol and can use the normal RTP deactivation protocol.

Figure 7-6 on page 7-10 and Figure 7-7 on page 7-12 show the protocol sequence for the deactivation of an RTP connection that was originally initiated by Node *X*. (The initiator of a connection is initially responsible to start the deactivation process.) Note that the deactivation process is performed when the number of active, pending active, and pending deactivate sessions on the RTP connection goes to 0. The first figure depicts a successful deactivation sequence in which the RTP connection is terminated. The second describes the case when the partner RTP endpoint still wants to use the connection; therefore, the connection remains active, but deactivation responsibility is transferred to the partner. The protocol uses the Client-Out-Of-Band (COB) signalling function provided by RTP. No user data is carried in a packet containing an COB segment. Session UNBINDs flow as user data before the deactivation process is started.

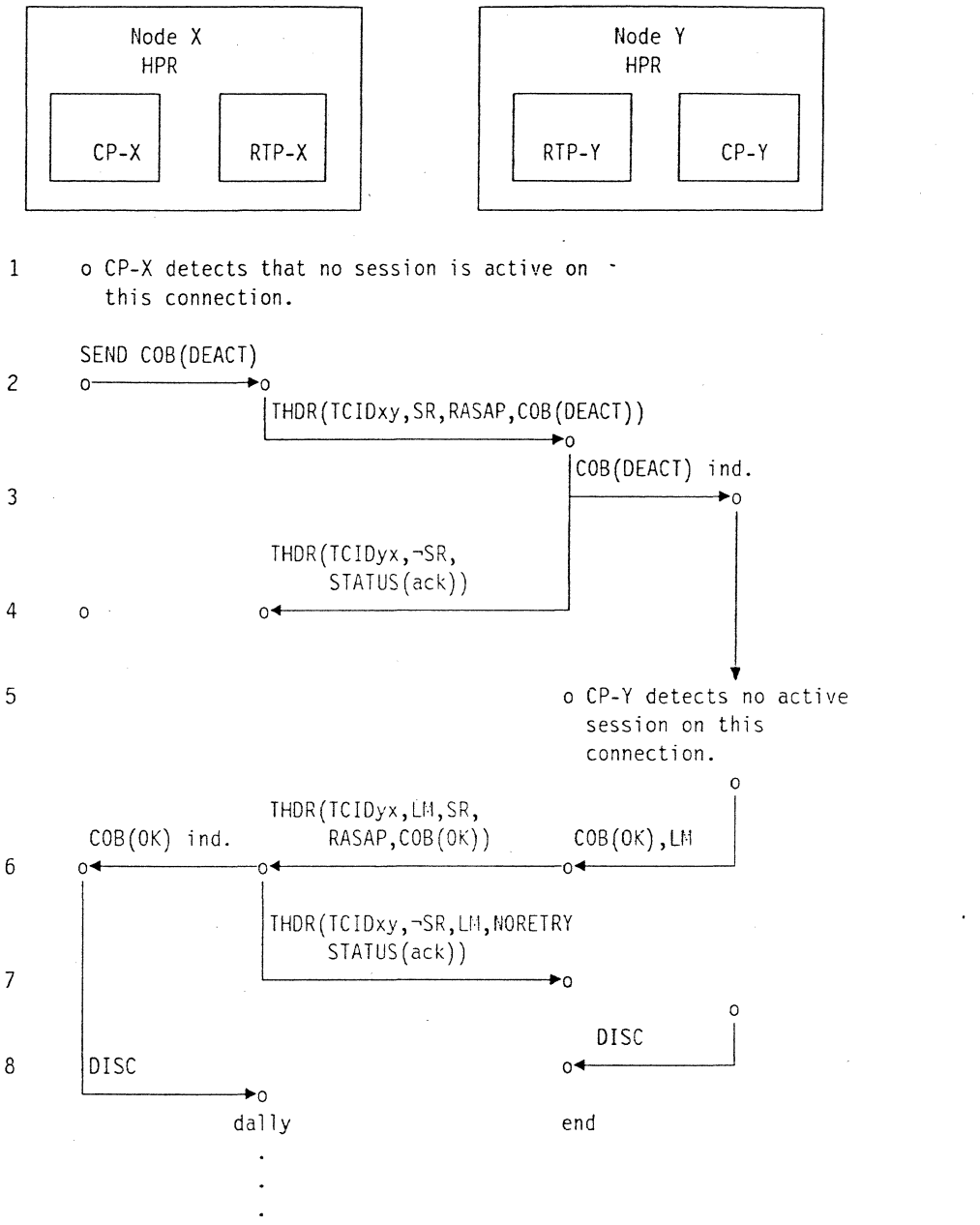


Figure 7-6. Successful RTP connection deactivation protocol

Annotations:

1 CP-X detects that the number of CP-CP sessions using this RTP connection has decreased to 0.

2 CP-X sends a Client-Out-of-Band (COB) signal to RTP-X indicating that it wants to deactivate this RTP connection. RTP-X sends this signal to RTP-Y as a COB segment and requests status as soon as possible.

3 RTP-Y delivers the COB signal to CP-Y.

4 RTP-Y sends RTP-X an implicit acknowledgement for the COB segment.

5 CP-Y agrees with the deactivation because, in its view, the number of sessions using this RTP connection is also zero.

6 CP-Y then sends a COB signal to RTP-Y indicating it agrees to the deactivation; CP-Y also indicates that it will send no further user data on this connection. RTP-Y sends a COB segment to RTP-X with the RASAPI and LMI bits set to 1. RTP-X delivers the COB signal to CP-X.

7 RTP-X sends status back to RTP-Y to acknowledge the last message and to implicitly acknowledge the COB segment. At this point, the connection is considered disconnected at both ends.

8 Both CPs issue Disconnect to the RTPs. RTP-Y will clean up the connection context immediately. RTP-X will dally because it is possible the last Status segment it sent to RTP-Y (in step 7) was lost.

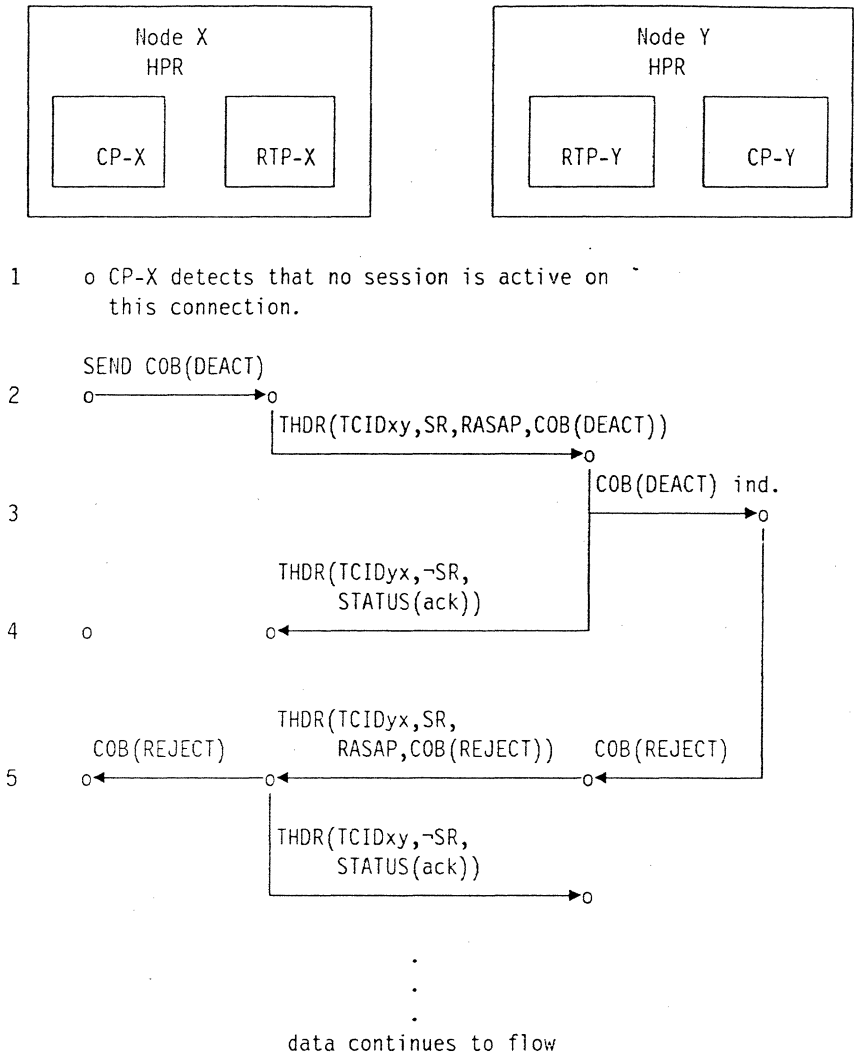


Figure 7-7. Rejection of RTP connection deactivation signal

Annotations:

1 CP-X detects that the number of CP-CP sessions using this RTP connection has decreased to 0.

2 CP-X sends a Client-Out-of-Band (COB) signal to RTP-X indicating that it wants to deactivate this RTP connection. RTP-X sends the COB segment and requests status as soon as possible.

3 RTP-Y delivers the COB signal to CP-Y.

4 RTP-Y sends RTP-X an implicit acknowledgement of the COB segment.

5 CP-Y detects that it is still using this connection (i.e., the number of sessions using this RTP connection is not zero). This situation arises when CP-Y has recently sent a BIND over this connection which CP-X did not receive prior to starting the deactivation protocol. CP-Y, as a result, sends a COB signal indicating it rejects deactivation. CP-Y now assumes the responsibility of deactivating this connection. RTP-Y sends the COB segment to RTP-X which, in turn, delivers the COB signal to CP-X. CP-X, upon receipt of the COB signal, stops the deactivation process and yields the responsibility for deactivation to CP-Y.

7.2.4 Session Traffic over RTP Connections

7.2.4.1 Reliable Data Transport and Error Recovery

All SNA session traffic is transported reliably over RTP connections. The Retry indicator and the Status Requested indicator (with its associated Status segment) are used by RTP in providing this service.

7.2.4.1.1 Retry Indicator: The retry indicator is set by the sending RTP endpoint in packets containing user data[2] to indicate that the sender will resend the data if it is not successfully received by the partner. When sending data reliably, the sending RTP keeps a copy of all unacknowledged data in send buffers allocated to the RTP connection so that it can be resent if necessary (i.e., when not successfully acknowledged by the receiver).

7.2.4.1.2 Status Requested Indicator: This indicator is set by the sender to request the receiver's status (i.e., acknowledgment of data successfully received). When the sender receives a Status segment from the receiver, it finds out what (previously sent) user data has been received. The send buffers containing copies of the user data successfully received are freed at this point. Data not successfully received (i.e., receiving partner detects gaps) will be retransmitted.

7.2.4.1.3 Timers: This section describes the RTP timers.

7.2.4.1.3.1 ALIVE Timer: The ALIVE (or liveness) timer is used to make sure that both the other endpoint of an RTP connection and the path between the endpoints are still operational after a period of inactivity. When this timer expires and no packet has arrived from the partner since it was last set, a packet with a Status Request indicator will be sent and the SHORT_REQ timer is started. If a Status segment is received from the partner, the SHORT_REQ timer is stopped. Otherwise, when the SHORT_REQ timer expires, the status request is retransmitted. After K retransmissions, if no Status segment is received, an attempt will be made by the sender to find a new path for this connection. If the partner is not operational or there is no suitable path to the partner, the sender will eventually terminate the connection.

The purposes of the liveness timer are as follows:

1. Keep limited resource links active

   Limited resource links are automatically deactivated in HPR when no traffic flows over the link for a specified period of time (link deactivation timer period). In order to keep these links up while RTP connections are using them, traffic must flow to keep the link deactivation timer from expiring. If there is no session traffic, RTP uses a liveness message which is sent at intervals set by the RTP liveness timer. After the RTP connection is deactivated, no RTP messages flow, and the limited resource links will be brought down upon expiration of the link deactivation timer.

---

[2] It is also set in packets containing a zero-length user message. Such messages may be used for reliable transmission of optional segments.

2. Detect hung RTP connections

   If the partner RTP endpoint is not operational or a link along the path fails, and the RTP endpoint is idle awaiting session data from the partner, then RTP is "hung." The liveness message is used to detect this condition; such detection triggers a path switch as described above.

The following describes how the liveness timer is used for the various types of RTP connections:

- RTP connection for Route Setup (long-lived)

  No liveness messages are ever sent on these connections. The Route Setup RTP connections are activated over each link and are deactivated when the link is taken down (i.e., a connection is only needed while its link is active). Thus, there is no need to detect a hung connection condition, and the liveness timer is not required to trigger a path switch. In addition, a Route Setup connection should not keep limited-resource links active unnecessarily.

- RTP connection for CP-CP or LU-LU sessions with one or more limited-resource links along the connection path

  For this case, the liveness timer is used to both detect a hung connection and to keep limited-resource links active. A liveness timer value is associated with each limited-resource link. The default value for this timer is 10 seconds but it may be overridden by the customer. When a Route Setup protocol is performed, the smallest limited-resource liveness timer value is obtained for the entire path (each node provides the value for its outgoing link if smaller than the value received in the request packet) and used by the RTP endpoints to govern the sending of liveness messages. (See A.1.15, "Route Setup GDS variable X'12CE'" on page A-31.)

  Note, the link deactivation timer is always larger than the liveness timer; the default is 3 times the liveness timer.

- RTP connection for CP-CP or LU-LU sessions with no limited resource links along the connection path

  In this case the liveness timer is used only to detect a hung connection and upon detection to trigger a path switch. The timer value may be associated with COS or TPF, or there may be one value used for all RTP connections within the node. The default for this timer is 30 minutes (but may be overridden by the customer).

The value of the liveness timer is communicated to the partner in the HPR Routing Information (X'83') control vector which is included in the Routing Information optional segment in the THDR (see A.1.7.6, "HPR Routing Information CV (X'83')" on page A-22.) The value communicated depends on the cases outlined above. Note that the liveness timer value may change during the life of the active RTP connection (e.g., when a path switch is done and the new path contains one or more limited resource links).

7.2.4.1.3.2 SHORT_REQ Timer: The SHORT_REQ timer is used to perform error recovery. When a sender of a packet with the SR bit set to 1 receives no response within a SHORT_REQ interval, the sender will initiate a state exchange with

RASAPI set to 1. If after K retransmissions, there is still no response, the sender will initiate a path switch for the connection (refer to 9.6, "LU-LU Path Switch" on page 9-10 for more details on path switch.) If, on the other hand, a response is received within SHORT_REQ interval, the SHORT_REQ timer is cancelled.

The SHORT_REQ timer is estimated dynamically based on Phil Karn's algorithm[3] . This algorithm has been implemented widely in TCP/IP networks; it is adapted for RTP and works as follows:

- Sample the roundtrip delays each time a status request is sent. This is done by marking the time S_i when a status request message is sent. When the appropriate Status segment is received at time R_i, the roundtrip delay RTT_i is obtained: $RTT_i = R_i - S_i$.

  Note that as a result of the ARB mechanism, status request and response occur at least periodically every time an ARB rate request is sent.

- Use exponential filtering to smooth the roundtrip time. Let $SRTT_{i+1}$ be the smoothed/filtered roundtrip time at time $i+1$, then: $SRTT_{i+1} = \alpha \times SRTT_i + (1-\alpha) \times RTT_i$. The parameter α here is used to determine how quickly we want to adapt to changes/fluctuations in RTTs with respect to the past estimates of $SRTT$.

- Set the SHORT_REQ timer to be: $\beta \times SRTT_i$, where $\beta \geq 1$. $SRTT$ is essentially the median of the roundtrip time, and β takes into account the variance.

- Use exponential backoff mechanism when timeout occurs. This mechanism doubles the SHORT_REQ timer for every retry until an RTP state exchange is completed successfully. The SHORT_REQ timer will be set to the timer period when the state exchange is finally successful. It will then be adapted dynamically based on the condition of the path.

  Due to this exponential increase in the SHORT_REQ timer, it may become unnecessarily large when the number of retries is high, which will lead to longer path switch time. To solve this problem, the SHORT_REQ timer shouldn't exceed 4 times the original SHORT_REQ timer period (i.e., the value after the second timeout occurs). For example, if the number of retries is six, the third through sixth retry will use the same SHORT_REQ timer period as the one computed for the second retry (third overall transmission). Once the status is returned and received, this period will be set to the new value.

The factors α and β are set to 0.875 and 2 respectively. Studies[3] have shown that these values are quite effective in estimating the roundtrip delay without requiring a lot of overhead (i.e., the algorithm can be performed with shift and add operations) for HPR networks. Furthermore in HPR, ARB is the mechanism used for congestion avoidance/control rather than the timeouts used by other protocols (e.g., TCP). As a result, additional overhead is not necessary in order to yield a more accurate (i.e., tighter bound) estimate of the roundtrip time.

The number of retries, K, is defaulted to 6, which should cover a wide range of link error rates. But a customer can override the value of K for the particular network. When K is exposed to the customer, preferably, it should be associated with COS or

TPF or node as different types of traffic have different type of responsiveness requirements in terms of failure detection and path switching.

7.2.4.1.3.3 Re-FIFO timer: See 7.2.4.3, "Re-FIFO" on page 7-18 for a description of this timer and how it is set.

7.2.4.1.3.4 Path Switch Timer: This timer is used to monitor the length of time that RTP should attempt a path switch for a connection upon detecting its failure. Section 9.6, "LU-LU Path Switch" on page 9-10 describes the path switch mechanism and the path switch timer in detail.

7.2.4.1.3.5 Dally Timer: The Dally timer is used by an RTP endpoint to make sure that its partner receives the last acknowledgment that it sent before issuing Disconnect. Once this timer expires, the connection context can be safely released or reused for a new connection (see 7.2.3, "RTP Connection Deactivation" on page 7-9 for a scenario where the dally timer is used). The dally timer is set based on the SHORT_REQ timer that is associated with the connection along the number of retries K, $dally\_timer = K \times (4 \times SHORT\_REQ)$.

7.2.4.1.4 Sending a Status Segment in Response to Status Requested: When a packet is received requesting status, the receiver responds immediately by sending a Status segment. The Status segment is piggybacked with user data if any such data is queued. Piggybacking is desirable because it lowers the packet processing load.

Alternatively, a timer could be used for piggyback optimization. For example, a SHORT_RSP timer could be set when a packet is received with status requested but the RASAP bit is set to 0. The Status segment would not be sent in a standalone packet unless the timer expired with no user data queued for transmission. The HPR design does not use such a timer. This will save some overhead in terms of timer processing at the RTP endpoints at the cost of more control traffic in the network.

7.2.4.2 Segmentation and Reassembly

RTP connection paths traverse a series of HPR links and nodes. Each link along the connection has a maximum BTU size. The link BTU sizes may differ from link to link. When RTP sends an NLP, its size (including NHDR, THDR, and DATA) must not exceed the smallest link BTU size along the path, otherwise data will be lost. This smallest link size will be referred to as the minimum link size (MLS) and is used for segmenting.

When RTP is requested to send a user message that would result in an NLP larger than the MLS, it will segment the NLP into MLS-size segments before sending it. The first segment includes an NHDR, a THDR, and the first portion of the DATA. Subsequent segments include an NHDR, a THDR, and portions of the DATA.

---

[3] P. Karn, C. Partridge: "Improving Round-Trip Time Estimates in Reliable Transport Protocols". ACM, 1988.

D. Sanghi, et al.: "A TCP Instrumentation and Its Use in Evaluating Roundtrip-Time Estimators". Internetworking: Research and Experience, Vol. 1, 77-99, 1990.

D.L. Mills: "Internet Delay Experiments". RFC-889, Dec 1983.

The receiving RTP will reassemble the pieces into the original user message (i.e., the session PIU when session traffic is being transported across the RTP connection). The Start-of-Message (SOM) and End-of-Message (EOM) indicators in the THDR are used for this purpose. Products are required to implement the segmentation and reassembly functions.

Figure 7-8 shows how segmentation and reassembly are done. The NLP is segmented into MLS-size segments by the RTP sender (the last piece, part n, may be smaller than MLS). The RTP receiver reassembles the pieces back into the original user message before passing it to the half session or session connector.

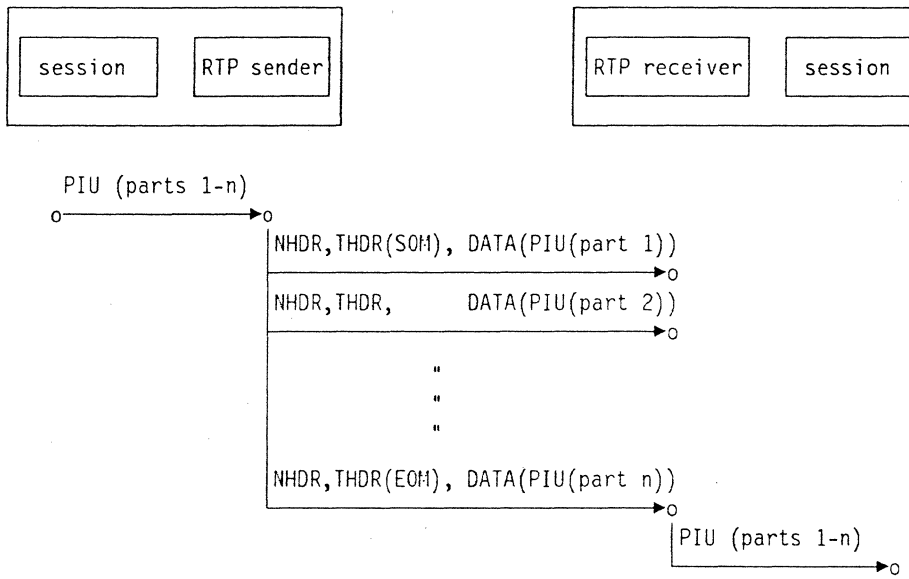


Figure 7-8. Segmentation reassembly of session PIUs on an RTP connection

If the RTP connection path is altered due to a path switch, RTP must be informed of the new MLS value so that it can do the segmenting properly.

7.2.4.3 Re-FIFO

One of the functions of RTP is to re-fifo data that arrives out of sequence before delivering it to users. Because RTP is designed to operate in a connection-oriented network, it expects the network to deliver data in sequence unless a packet has been lost. Therefore, when a packet arrives out of sequence, RTP treats it as an error and attempts to perform error recovery immediately. This procedure consists of sending a "gap-detected" message to the sender which requests the sender to retransmit the lost packet(s), and queueing subsequent packets until this gap is filled (i.e., selective retransmission is a function of RTP). Since HPR supports MLTG (Multi-Link TG)[4], packets may arrive out-of-order as depicted in the figure below. Traffic for an RTP connection from A to to D are sent over the ANR path 1, 2, and 3. ANR 2 is an MLTG with 3 physical links a, b, and c. Once the packets belonging to this connection arrive at B, they can be distributed over a, b, and c as part of the load balancing across these links. As a result, packets could arrive at D

out-of-sequence. By having RTP in D delay its error recovery procedure, packets will be given time to arrive without having to be retransmitted unnecessarily.

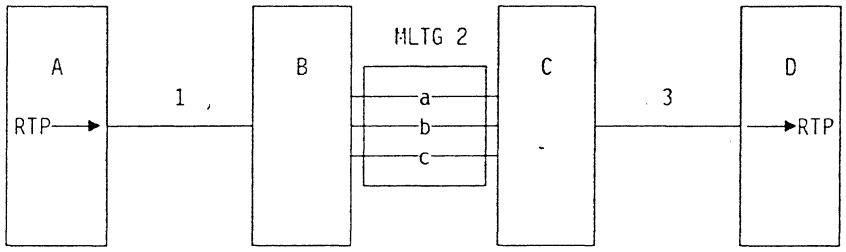


Figure 7-9. A connection traverses a path which contains a MLTG

As indicated in the above example, it is not desirable for RTP to react prematurely upon detecting a gap if a connection traverses one or more MLTG links. This would cause wasted throughput due to unnecessary retransmissions if packets are delayed over the MLTG links and arrive out-of-order. Therefore, in order to allow for the late arrival of packets, if there are one or more MLTGs along a path of a connection, RTP delays its error recovery if the connection is reliable, or delays forwarding out-of-sequence data to users if the connection is non-reliable. If there is no MLTG along the connection path, the RTP error recovery function is activated immediately once a gap is detected.

This algorithm yields optimal performance with respect to recovering lost data. Indication of whether or not a path has an MLTG is done during route setup and is reflected by the Re-FIFO bit in the route-setup reply message returned to the originator of the route-setup message. This bit can be set by any node that owns an MLTG on the connection path (see Appendix A, "Formats" on page A-1 for details of the format).

As mentioned in 7.2.4.1.3, "Timers" on page 7-14, RTP keeps a running average of the roundtrip delay time for a connection. By delaying the error recovery procedure for a reliable connection (or by holding out-of-sequence data temporarily before forwarding it to the user for a nonreliable connection) by half of this roundtrip time (i.e., the time it takes for a packet to traverse from source to destination including some amount of queueing time), it is highly likely that delayed packets will arrive before error recovery is triggered (or lost data is reported to the user). Note, however, if the disparity of the link speeds in a MLTG is high (e.g., 9.6Kbps vs. T1[5]), half of the roundtrip time might not be long enough and may cause unnecessary retransmissions.

---

[4] Currently, there is no architecture defined to support MLTG for APPN. The support of MLTG in HPR is mainly due to it being available in the Subarea SNA networks.

[5] Such an MLTG can distort COS routing as well.

If an RTP connection consists of only one way traffic (this is not true for connections carrying SNA session traffic which are always 2-way because they carry BIND and RSP(BIND)), one end of the RTP connection will not have a running average of the roundtrip delay. In this case, the receiver can use the measurement interval supplied by the sender on rate request messages as an estimate for the roundtrip time. This is reasonable because the measurement interval is of the same order as the roundtrip time.

7.2.4.4 Adaptive Rate-Based Flow/Congestion Control Algorithm

7.2.4.4.1 Introduction: The adaptive rate-based (ARB) flow/congestion control algorithm is a congestion avoidance and control mechanism that allows Rapid Transport Protocol (RTP) connections to make more efficient use of network resources. The basic approach is to regulate the input traffic (offered load) during changing network conditions. When the algorithm detects that the network is approaching congestion (i.e., there is increasing delay and decreasing throughput), it reduces the input traffic entering the network until the precongestion indications go away. When the network is sensed to have enough capacity to handle the offered load, the algorithm adapts by allowing more user traffic to enter the network unless doing so will exceed the rate that the receiving endpoint can handle. The ARB algorithm is designed to meet the following objectives:

- The algorithm should be adaptive to the network conditions and be responsive to the arrival rate of user traffic in order to maximize throughput and minimize congestion.
- The algorithm should smooth the input traffic into the network (i.e., avoid large bursts) when the physical capacity of the access link to the network is larger than the allowed input rate. This helps prevent long queues from developing in the network.
- The algorithm should provide both effective congestion control and end-to-end flow control.
- The algorithm should be simple to implement and require minimum overhead in both processor cycles and network bandwidth.
- The algorithm should be fair in providing equal access to all RTP connections.

The ARB algorithm employs a closed-loop, distributed, control mechanism based on information exchanged between two partner RTP connection endpoints. Figure 7-10 shows an overview of the closed-loop control mechanism between a sender and a receiver:

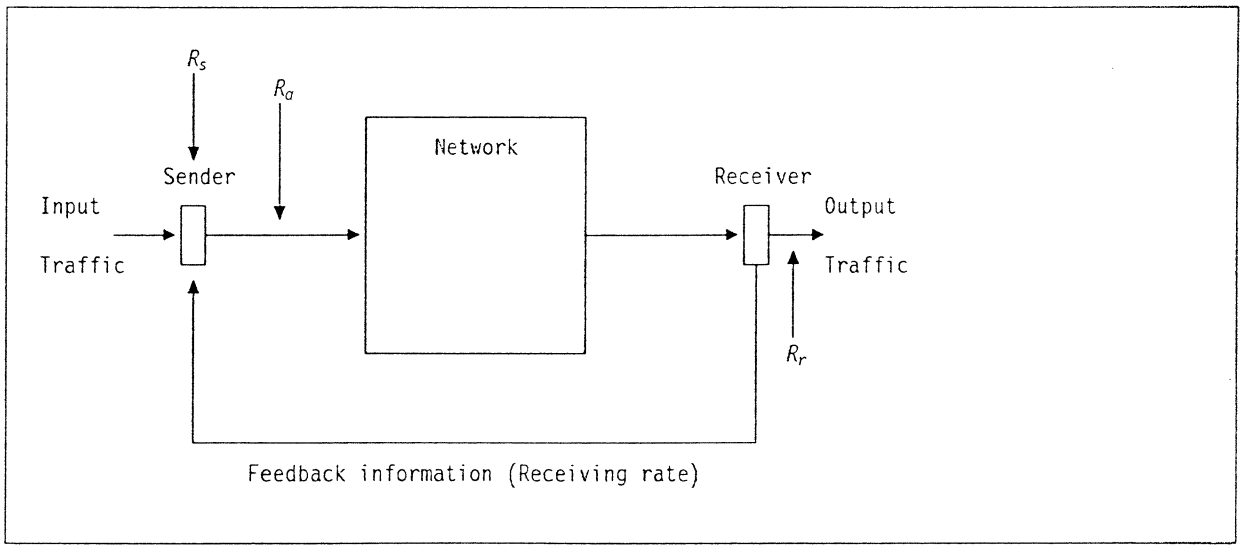


Figure 7-10. Overview of a close-loop control mechanism

7.2.4.4.2 Integration of ARB into RTP: The ARB algorithms are implemented in the endpoints of RTP connections. Each connection is a full duplex pipe; at each end of a connection, there are two components, a sender and a receiver. The ARB procedures performed by the sender at one end are the same as those performed by the sender at the other end; the same is true for the receivers. RTP endpoints will perform these functions:

- Regulate/monitor the traffic (i.e., packets) which it transmits into the network based on its allowed send rate
- Send rate requests and rate-replies to the RTP machine in the other endpoint
- Monitor the receive rate and perform rate computation upon receiving a rate request

7.2.4.4.3 Exchange of Rate Information: A receiver in one RTP endpoint provides receive rate information to the sender in the other endpoint. This information reflects the state of both the receiver and the network. The sender, based on the information, regulates the input traffic into the network.

The receive rate provided to the sender is the minimum of two rates; one is the rate at which the receiver accepts arriving data from the network, and the other is the rate at which the receiver delivers data to the end user. The former indicates the status of the path in the network and the latter the receiving capacity of the end user. The sender uses the minimum of the two rates to regulate the input traffic so that it can both avoid overloading the path (thus providing congestion control) and also avoid overloading the receiving capacity of the end user (thus providing end-to-end flow control). Based on the rate information received, the sender can determine when congestion is about to develop in the network and take appropriate actions to avoid it. When congestion does occur, the sender takes extraordinary measures to relieve the network congestion.

ARB information (e.g. request/reply of rates) will be carried in a new ARB Segment which is part of the Optional Segment field in the RTP transport header.

7.2.4.4.4 Definitions of ARB Parameters: Definitions of ARB parameters are given below:

- R_s is the maximum average rate at which the sender is allowed to transmit data into the network during each burst time B_t. Packets are actually transmitted into the network at the physical rate of the access link; thus, when R_s is less than the physical rate, there must be a sufficient portion of each burst time during which no data is transmitted into the network.
- R_a is the actual rate, averaged over one measurement interval M, at which the sender transmits data into the network. $R_a \leq R_s$, because there may be periods during which an end user has nothing to send (refer to Figure 7-11 on page 7-23 for an illustration.)
- R_r is the minimum of the rate at which the receiver accepts data from the network and the rate at which the end user can process the data. This quantity is measured by the receiver and returned to the sender.
- B_t is the burst time.

- B_s is the number of bits that the sender is allowed to send into the network during one burst time. These bits may be sent continuously at the physical rate of the access link. $B_s = B_t \times R_s$
- M is the length of the measurement interval. M is used in computing the actual rate, $R_a = N_M/M$, where N_M is the number of bits sent during the measurement interval of length M.
- T_{out} is the period of the RTP SHORT_REQ timer and represents the maximum time that a sender waits for a reply from a receiver. When the roundtrip delay exceeds this value, it is an indication that network congestion may have caused the packet either to be discarded or delayed. Appropriate action, discussed later, is taken to relieve the congestion.

Figure 7-11 shows the relationship between the parameters described above within one measurement interval. In the figure, R_s is 50 percent of the physical rate.[6] R_a is less than R_s because there are periods when the end user has nothing to send. Specifically, all queued user data is transmitted during the second burst time; addition user data arrives during the third burst time, and its transmission is completed during the fourth burst time.

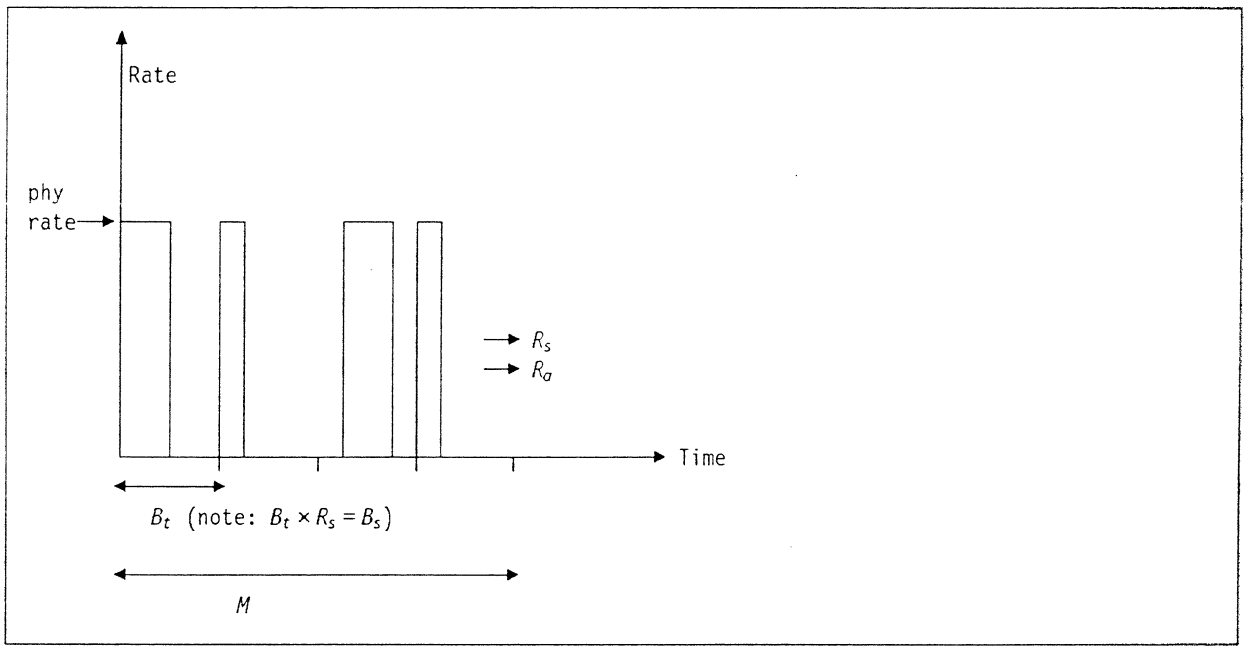


Figure 7-11. ARB Parameters (shown within one measurement interval)

The value of the following parameters are dependent on the path selected at the time of RTP connection activation or of a path switch; they are derived as follows:

- The initial value of R_s is a fraction of the capacity of the slowest link; it is adapted to the network conditions throughout the life of the connection.

[6] In the figure, R_s is constant during the measurement interval; however, R_s may change during a measurement interval when the receiver returns its receive rate.

- The maximum rate, R_{max}, is the maximum value to which R_s can be set. This maximum rate may be set to the smallest link capacity along the path or system defined to a lower value based on the class of service (COS).
- The timer period, T_{out}, is computed dynamically as described in section 7.2.4.1.3, "Timers" on page 7-14
- There are additional ARB parameters that need to be set during RTP connection activation; they will be discussed later in 7.2.4.4.6, "Analysis of ARB parameters" on page 7-28.

7.2.4.4.5 ARB Algorithm:

The feedback mechanism, and frequency of measurements

At the end of each measurement interval of length M, the sender includes an ARB rate request in the next packet transmitted to the receiver in the partner RTP machine; i.e., the sender inserts an ARB Segment with a Message Type of either "Rate Request" or "Rate Request and Rate Reply" in the Optional Segment field of the packet's RTP transport header. The ARB segment includes the sender's measurement interval M. Upon receipt of this rate request, the receiver computes its current receive rate R_r, which is the smaller of the total number of bits either received from the network or forwarded to the end user since the receipt of the previous rate request divided by the receiver's measurement interval M_r (i.e., the elapsed time since the receipt of the previous request). See Figure 7-12 on page 7-25.

The receiver returns the receive rate R_r to the sender in an ARB rate reply; i.e., the receiver transmits a packet with an ARB Segment of Message Type "Rate Reply" or "Rate Request and Rate Reply" to the sender. The ARB Segment is piggybacked with user data when possible. Figure 7-12 on page 7-25 illustrates requests and responses; it also shows how the rates are measured:

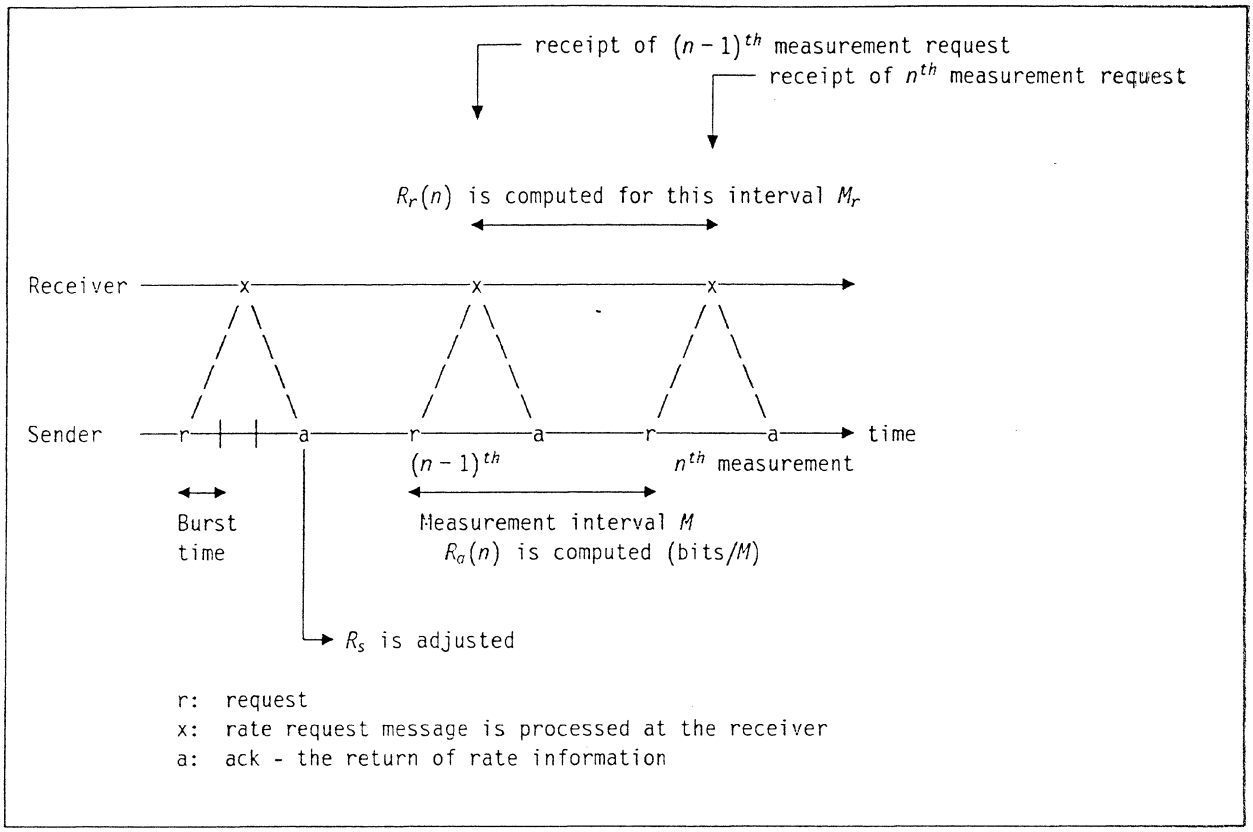


Figure 7-12. Rate Measurement

Adjustment of sending rate (R_s) - adaptive mechanism

The sending rate R_s is adjusted based on the feedback rate information returned by the receiver. This information is used by the sender to determine the throughput conditions along the path of the connection. Figure 7-13 on page 7-26 shows the network throughput versus input rate for a path. The operating point (point A) is the point beyond which the path starts to become congested (i.e., transmission queues develop along the path). Beyond point A, an increase in the send rate does not result in an increase of throughput as reflected by the receive rate. ARB detects this precongestion condition (saturation) and adjusts the sending rate accordingly, thus preventing the network from operating around the cliff (point B). The cliff is the point beyond which congestion results in significant packet loss and large queueing delays at the links along the path.

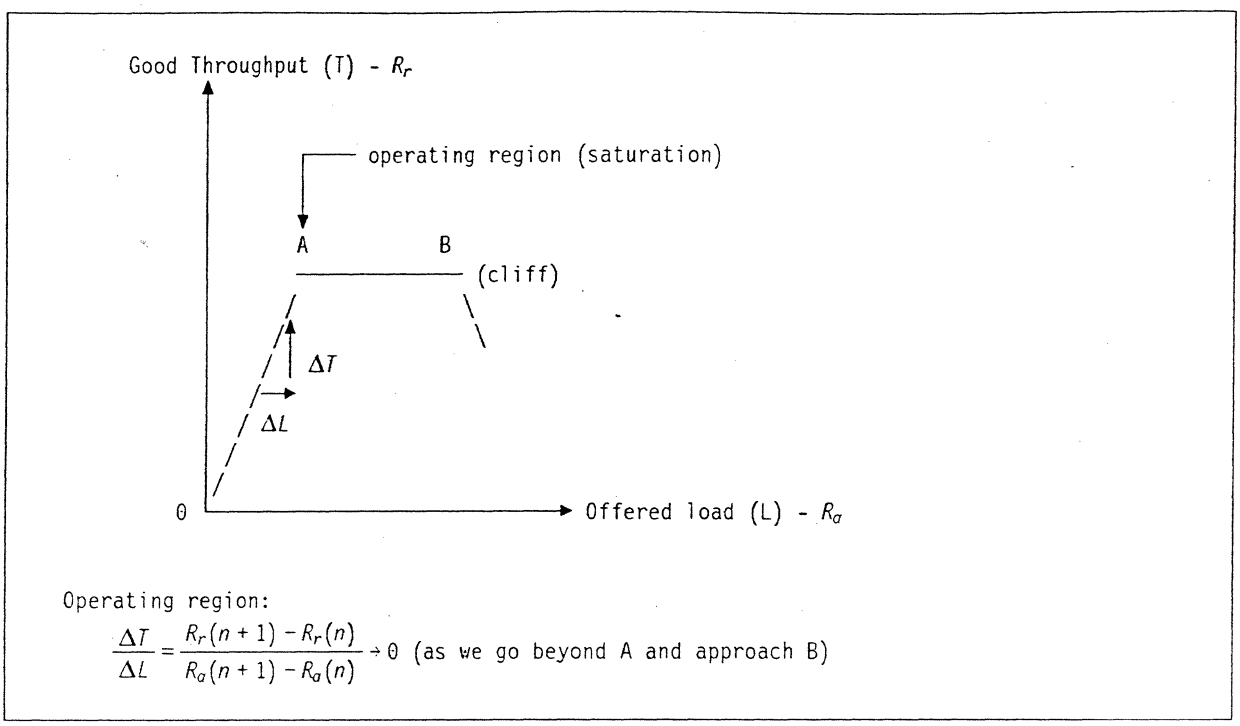


Figure 7-13. ARB operating region

We define three operational modes which determine how the rate is adjusted based upon the feedback information, they are represented as RED, YELLOW and GREEN modes as shown in Figure 7-14 on page 7-27. where the horizontal line represents the actual send rate at a sender. The operating mode is set as follows:

- Set to GREEN when $R_r \geq R_s - \Delta_R$. Δ_R is referred to as the sensitivity threshold and is discussed in detail in 7.2.4.4.6, "Analysis of ARB parameters" on page 7-28.
- Set to YELLOW when $R_r < R_s - \Delta_R$.
- Set to RED when a timeout waiting for a reply occurs (e.g., waiting time exceeds T_{out}). R_s is decreased by half when this situation occurs.

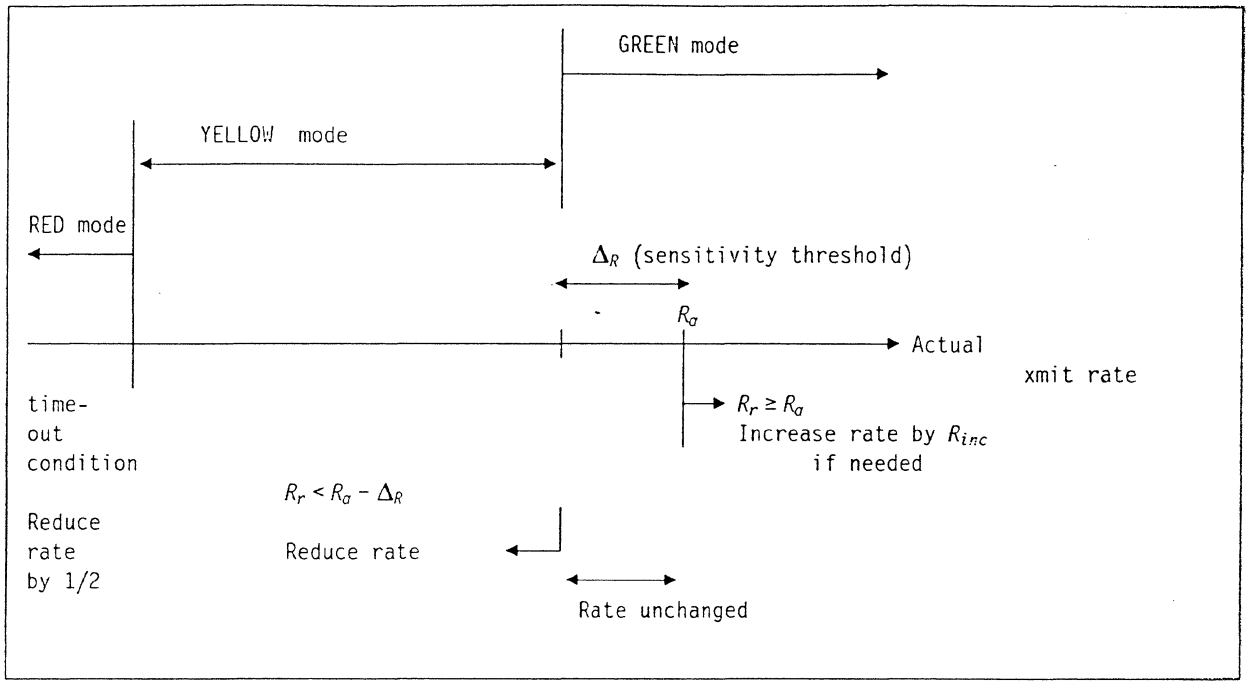


Figure 7-14. Rate adjustment based on feedback

If the current operating mode is GREEN, and the receiver indicates that it can handle a higher rate (i.e., $R_r \geq R_a$), R_s can be increased additively by an amount of R_{inc}, as long as the increased send rate is not higher than its maximum value R_{max}. The operating mode remains in GREEN.

If the current mode is not GREEN, the rate should not be increased but remains the same until the next measurement; the purpose is to reduce oscillation in the send rate and in the network loading.

If the receiving rate is slower than the sending rate by more than Δ_R ($R_r < R_a - \Delta_R$), the sending rate will be reduced to $\theta \times R_r$ (see Figure 7-14 for an illustration, and section 7.2.4.4.6, "Analysis of ARB parameters" on page 7-28 for more discussion on Δ_R). The operating mode is then set to YELLOW. The slower receive rate indicates that either the path is saturated, or the receiver can only handle up to this rate. Note that modeling results indicate that θ should be set such that $0.9 < \theta < 1$.

With respect to the parameter θ, note that when a link operates near its capacity, it becomes a bottleneck, and each of the RTP connections traversing the link will receive a fraction of the capacity of the bottleneck link. This represents the path saturation point for these connections. θ should be set such that the connections are allowed to oscillate closely around this point. For HPR, θ is set 0.9375 (a shift and subtract operations).

R_{inc} can be defined to be large at first so that the sending rate can be increased quickly to the operating range. Once the path or receiver saturation point is reached, this value is reduced to a small value to avoid wide fluctuations around the operating range. When path bandwidth is detected to be available again, R_{inc} can

gradually be increased (see 7.2.4.4.7, "Variations of ARB algorithm" on page 7-33) to take advantage of the newly available capacity.

When a sender times out while waiting for a reply (i.e., feedback of rate information) from the receiver or detects packet loss, the mode is set to RED, and R_s is reduced by half ($R_s = 0.5 \times R_s$), or set to the receiving rate R_r, whichever is lower.

7.2.4.4.6 Analysis of ARB parameters: The most important parameter in the ARB algorithm is the sensitivity threshold, Δ_R. This parameter determines how sensitive a sender is to the path and receiver condition when adjusting its send rate based on the receive rate returned from the receiver (see Figure 7-14 on page 7-27.) The larger Δ_R is, the less sensitive to the saturation condition along the path the send rate adjustment becomes. In other words, if Δ_R is too small, a sender would react very quickly to any queueing delay in the network, and it would result in lowering the network throughput. On the other hand, if Δ_R is too large, a sender wouldn't react until there is large queueing in the network; this would drive the network to operate near the cliff (see Figure 7-13 on page 7-26) and result in potential traffic loss. The remainder of this section gives an analysis of Δ_R.

Let D_s be the size of the data (in bits) sent by a sender in a measurement window when the corresponding sending rate is R_s.

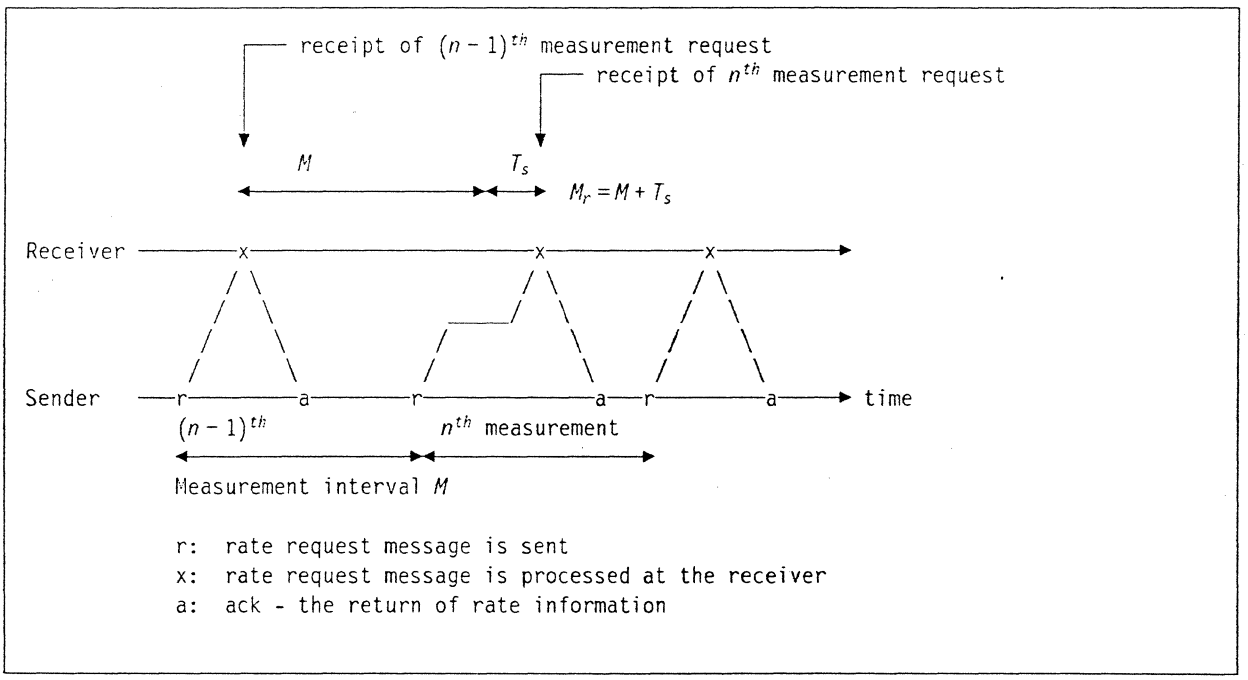


Figure 7-15. Delay of rate request message

Let T_s be the difference in the time interval of the sender's measurement and the receiver's measurement. T_s is a function of time which quantifies the change in the time-varying delay associated with the path. A positive T_s indicates an increase in the network queueing delay time of the rate request messages which, if large enough, will cause the ARB algorithms to lower the receive rate. If T_s is positive and just large enough to cause the sender to decrease R_s, then $R_r + \varepsilon = R_s - \Delta_R$, where ε is

infinitesimal. We also assume that the actual send rate R_a is equal to the allowed send rate R_s, i.e., the sender has data to send in each burst interval. T_s can be expressed as:

$$T_s = \frac{D_s}{R_r} - \frac{D_s}{R_s}.$$

Substituting R_r with $R_s - \Delta_R$ (ignoring ε), yields:

$$T_s = \frac{D_s}{R_s - \Delta_R} - \frac{D_s}{R_s} \qquad (1)$$

$$= \frac{D_s}{R_s}\left(\frac{1}{1 - \Delta_R/R_s} - 1\right) \qquad (2)$$

From Eq. (1), Δ_R can be expressed as a function of R_s for a particular value of T_s and M as follows:

$$\Delta_R = \frac{R_s^2 \times T_s}{D_s + R_s \times T_s} \qquad (3)$$

or $$\Delta_R = \frac{R_s \times T_s}{M + T_s} \qquad (4)$$

Let T_q be the maximum delay that a connection is allowed along a path in the network. That is if the delay is greater than or equal to T_q, then the sender is to reduce the transmission rate. If T_q is substituted for T_s in equation (4), Δ_R can be expressed as a fraction of R_s as follows:

$$\Delta_R = \left(\frac{T_q}{M + T_q}\right) \times R_s \qquad (5)$$

T_s is the measure of increased delay along the path; Setting T_s to T_q, the maximum acceptable delay, means that the sender algorithm will be unable to detect a gradual increase in the delay across the path. This problem is solved by enhancing the receiver algorithm to detect accumulation of delay; the enhancement is described in 7.2.4.4.6.1, "Drift Analysis of Path Delay" on page 7-30.

Equation (2) shows that as the ratio (Δ_R/R_s) approaches zero (fixed Δ_R and increasing R_s), T_s approaches zero as well. This causes the sender to be overly sensitive in reacting to changes in the network condition, even to a very small delay (e.g., much less then a packet transmission time). This suggests that Δ_R should vary with the transmission rate to ensure efficient utilization of the network.

Equation (5) shows that Δ_R is a fraction of R_s with the fraction defined in terms of two parameters, M and T_q. T_q is dependent on the characteristics of the path; thus, it remains constant as long as the RTP connection traverses the path. Consequently, if M is kept stable (i.e., doesn't vary significantly from its mean), the fraction need not be adjusted for every measurement interval, and processing overhead is reduced.

In HPR, the roundtrip delay of a path is estimated dynamically as described in 7.2.4.1.3, "Timers" on page 7-14. This roundtrip delay time is used as a basis for setting the measurement interval M. It is worth noting that M determines the level of adaptivity to network changing conditions; a smaller M (i.e., relative to the roundtrip time) provides better adaptivity but is costly in terms of processing overhead at both the receiver and the sender of a connection. M also depends on the capability of buffering in the networks in that if the network has large buffering capacity capable of handling short term (i.e., on the order of roundtrip time) bursts, then M can be made long relative to the roundtrip time and overhead can be reduced. On the other hand, if the network has small buffering capacity, M should be small to be able to react quickly to any fluctuations in the network.

Also, M should also be made dependant on the send rate in that a sender with high throughput (i.e., high transmission rate) requirement should be required to sample the network/path more frequently. On the other hand, a sender with smaller throughput requirement should be allowed to increase M. This policy, "pay-as-you-use," is fair to users of shared resources.

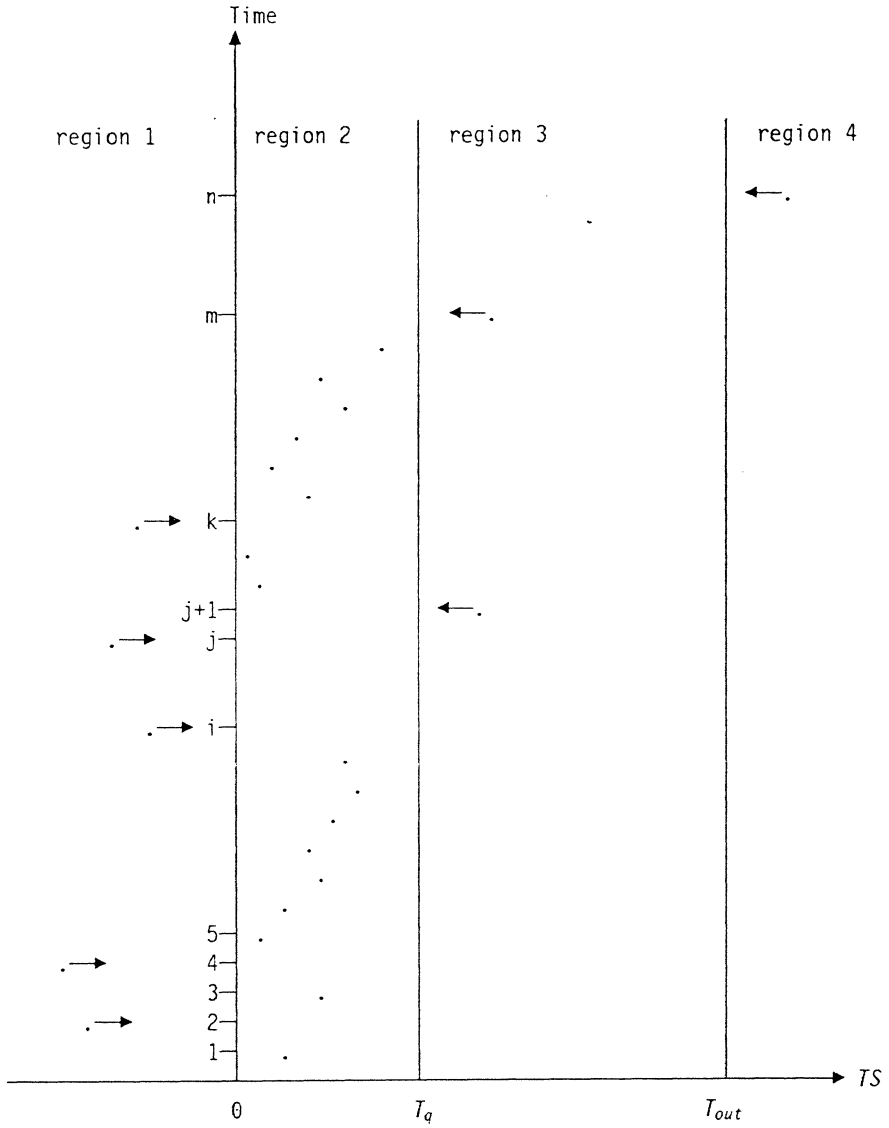
The setting of T_q depends on the characteristics of the path, such as the path throughput capacity which is constrained by the bottleneck link(s) and the buffering capability. Note that for an M/M/1 queue, the maximum power is achieved when the average queue length is 1. Therefore, T_q does not have to be large to gain optimal efficiency as will be shown later in our simulation studies (i.e., 2 to 4 packet worth of queueing time has been shown to yield very good performance). In fact, large T_q will cause wide oscillations and high variance in network performance.

Let's consider the following example in setting T_q. Let a connection with a path that traverses a token ring of 16Mbps, a T1 link, and another token ring of 16Mbps respectively. Let each packet be of fixed size of 500 bytes. Let T_q consist of 4 packets worth of transmission time (i.e., when a sample packet arrives, it sees one packet starting transmission and three packets waiting in the queue) at each hop. Clearly, the queueing time at the T1 link will be the dominating factor. The total T_q is then equal to: $T_q = 13ms$.

The parameter T_q determines how much queueing time in the network a connection is allowed. A connection with larger queueing time allowed at each link in the network will gain higher throughput compared with a connection which, on the same path, has smaller value of queueing time. This is because the latter will be more sensitive to any queueing developed along the path in the network. Connections with equal queueing time will be equally sensitive and therefore, will get equal throughput. The setting of T_q can be tied to the COS of a connection in order to provide higher throughput when required.

7.2.4.4.6.1 Drift Analysis of Path Delay: T_s is the measure of increased delay along the path; however, it was set to T_q, the maximum acceptable delay. Next, we will study the drift (i.e., the stability of the system) of T_s and describe how the ARB receiver algorithm keeps the network in a stable state (i.e., with an acceptable level of queueing in the network). To do so, the receiver algorithm needs to detect gradual queue build up along a path and to notify the sender accordingly. The following figure will be used to demonstrate this stability behavior of an HPR network.

Drift of T_s



Each arrow (→) represents the direction of the drift of T_s as a result of an action taken by the ARB algorithm. It also represents the beginning of a new T_s cycle.
Each dot (.) represents the sum of T_s's values since the restart of the current T_s cycle. This sum is denoted as TS.

Figure 7-16. Drift analysis

Let TS be the sum of T_s's values since the start of the current cycle, we have:

$$TS = \sum_{i=1}^{n} T_s(i)$$

after n measurement intervals, where $T_s(i)$ is the T_s value for the i^{th} measurement interval of the current cycle. TS is set to 0 at the start of each cycle. Figure 7-16 shows four regions with respect to TS:

- Region 1 is when $TS \leq 0$. This happens when there was queueing delay in the network during previous rate requests, and the queues have either disappeared or decreased in size at the time of the current rate request. Because $T_s(n) < 0$, the rate is increased gradually to take advantage of the capacity of the network. Therefore, the drift is toward the right, which means increasing throughput.
- Region 2 is when $0 < TS \leq T_q$. Here the sending rate is either increased gradually ($T_s(n) < 0$) or remains the same ($T_s(n) \geq 0$, wherein the rate is within the sensitivity threshold).
- Region 3 is when $TS > T_q$. The sending rate will be reduced since the allowed network queueing time has been exceeded. The drift is therefore toward the left to reduce the level of queueing in the network. TS enters this region under 2 conditions: first, TS consists of small values of $T_s(i)$'s (i.e., gradual build up of queues in the network); second, $T_s(n)$ in the last measurement of the current cycle exceeds T_q.
- Region 4 is when $TS > T_{out}$. The receiver can safely assume that a timeout condition will be detected by the sender, and that the sending rate will be cut by half to get the network back to the stable state quickly. The drift is again toward the left.

A new cycle is started upon entry into Regions 1, 3, or 4. By starting a new cycle when Region 1 is entered, TS is made a better estimate of the path's queueing delay. A new cycle is started upon entry into Regions 3 or 4, and the sending rate is also reduced accordingly. This is done to ensure that an RTP connection that has observed the path's queueing delay and has taken appropriate actions as a result does not need to reduce its rate again; other connections should cut their send rates before this connection makes additional cuts.

Figure 7-16 on page 7-31 shows some examples of TS as a function of the $T_s(i)$'s. At time t = 1, $T_s(1)$ is positive, indicating that there is some level of queueing along the path in the network but within the allowed threshold. At time t = 2, $T_s(2)$ is negative indicating that the queue(s) has either disappeared or been reduced, it also causes TS to become negative. As a result, the sender will increase the rate and a new cycle of TS is started. Similar events happen at time t = 3 and t = 4. From t = 5 to t = $i - 1$, TS fluctuates within the threshold as a result of variations of the queues along the path. At t = i and t = j, TS becomes negative and each time a new cycle is started. At t = $j + 1$, there is a sudden burst of traffic which causes $T_s(j + 1)$ to exceed the threshold T_q. The sending rate is reduced as a result, and a new cycle of TS is started. From time t = k to t = m, the queues along the path fluctuate and gradually build up to exceed the threshold ($TS > T_q$); the sending rate will be reduced and, a new cycle of TS is started. At time t = n, the sender will detect a timeout. When the queues fluctuate within region 2 without exceeding the threshold, after some number of measurements, the send rate should also be reduced to allow the clearing of queues at bottlenecks along the path in order to improve stability.

From the above discussion, we have shown that the ARB algorithms will operate within region 2 which is a safe and stable region with relation to network loading. The receive algorithm is enhanced to achieve this stable behavior. This algorithm keeps track of TS which is used to adjust the value of M_r used to calculate R_r when there has been a gradual build up of network delay.

- Receiver algorithm to compute M_r

```
- Measure elapsed time Mr since last rate request.
- Compute Ts = Mr - M
- Accumulate TS = TS + Ts
- If (TS < 0)
      Reset TS = 0  /* start a new cycle */
  Else
     If (TS > Tq) or (persistent queueing delay)
        If (Ts < Tq)
              Adjust Mr = Mr + Tq
               /* force the sender to cut the rate */
        End
        Reset TS = 0  /* start a new cycle */
     End
  End
```

7.2.4.4.7 Variations of ARB algorithm: In the previous sections, the basic ARB algorithms and the delay measurement enhancement were described. In the following, a list of potential variations in the ARB algorithms is given:

1. A sender can set R_{inc} to a larger value after a number of measurements which indicate that the path is clear (the receiving rate always catches up with the sending rate). This will allow a sender to increase its rate more quickly. This option is beneficial in situations where a number of connections have just been deactivated.

2. A sender can reduce R_s by a small amount (e.g., R_{inc}) after each measurement interval in which there wasn't any data to send. This will reduce the probability that a number of connections will transmit a burst of data at the same time at a high rate after a long idle period.

3. A sender can also be flexible with respect to the burst size, B_s. If a packet arrives carrying s bits of data, the remaining number of bits in a burst, b, is smaller than s, and if $(s - b)$ is small, a sender can send a packet into the network anyway.

4. A sender can also use different values of Δ_R's for different ranges of its sending rate. If the sending rate is below certain level, a smaller Δ_R can be used (see 7.2.4.4.5, "ARB Algorithm" on page 7-24 for more discussions on this parameter). When the sending rate exceeds this level, switch to a larger value of Δ_R. Δ_R should be reset to a minimum value when a timeout occurs.

   Δ_R can also be dynamically adjusted based on eq. (5) in 7.2.4.4.6, "Analysis of ARB parameters" on page 7-28. This is the current design direction.

7.2.4.4.8 RTP Connection Activation: The following flow highlights ARB activity during RTP connection activation:

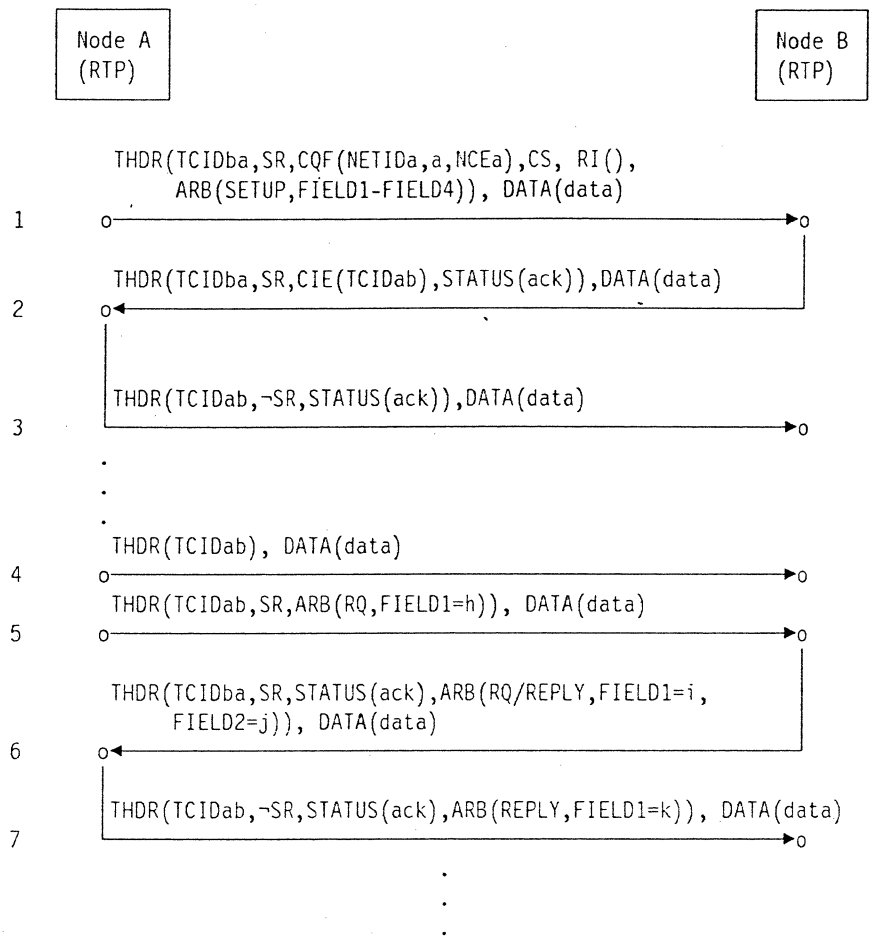


Figure 7-17. RTP Connection activation with ARB option

Annotations:

1 Node A RTP sends a connection setup request to Node B RTP. The ARB segment contains the following ARB setup parameters for the associated connection path: measurement interval, maximum allowed queueing time, initial send rate, and minimum rate increment. The connection setup request carries user data and has the Connection Setup Indicator and Status Requested Indicator bits set to 1. This packet marks the beginning of the first measurement interval at the sender in node A RTP.

2 Node B RTP sends back an acknowledgement along with the Connection Identifier Exchange Segment and user data. This packet marks the beginning of the first measurement interval at the sender in node B RTP.

3 Node A acknowledges the receipt of the user data, and sends along more user data.

4 Node A continues to send user data to node B.

5 The measurement interval elapses at node A; node A RTP sends the ARB segment to request rate information from the corresponding receiver in node B (node A RTP also provides the value of the measurement interval 'h' in the ARB segment) piggybacked with additional user data.

6 The receiver in node B RTP processes the ARB request, computes the receiving rate $j = R_r$, and tells the sender in node B RTP to send back the response. The sender in node B RTP detects that its measurement interval has also elapsed; it creates an ARB Segment both to return a rate reply in response to the earlier rate request, and to request the receive rate from node A. The sender in node B RTP also sets i to its measurement interval for R_e and sends the ARB Segment in a packet with the Status Requested Indicator bit set to 1, and with a Status Segment and additional user data included.

7 The receiver in node A RTP processes the ARB request along with the status segment. It computes the corresponding rate $k = R_r$ for the reverse direction, and tells the sender in node A to send back a rate reply and a Status Segment. The sender in node A RTP sends the rate information and the status, along with the next user message.

7.2.4.4.9 ARB Modeling: A simulation model has been written in SIMSCRIPT language to study and verify the ARB algorithms described above. The figure below (Figure 7-18 on page 7-36) shows the general configuration of the model. The model consists of:

- Four communication nodes: A, B, C, and D
- Bidirectional links shown as pairs of unidirectional links numbered 1 to 6 (Each link is associated with a capacity and a propagation delay.)
- Multiple "half-duplex" connections either between node A and node D or between node B and node D. And there are connections, in some cases, from node D back to node A to simulate two way traffic scenarios.
- Multiple external traffic sources entering each link (These external sources can be used to simulate traffic that is not regulated by the ARB mechanism, e.g., FID2 traffic.)

All traffic sources are on-off exponentially distributed processes.

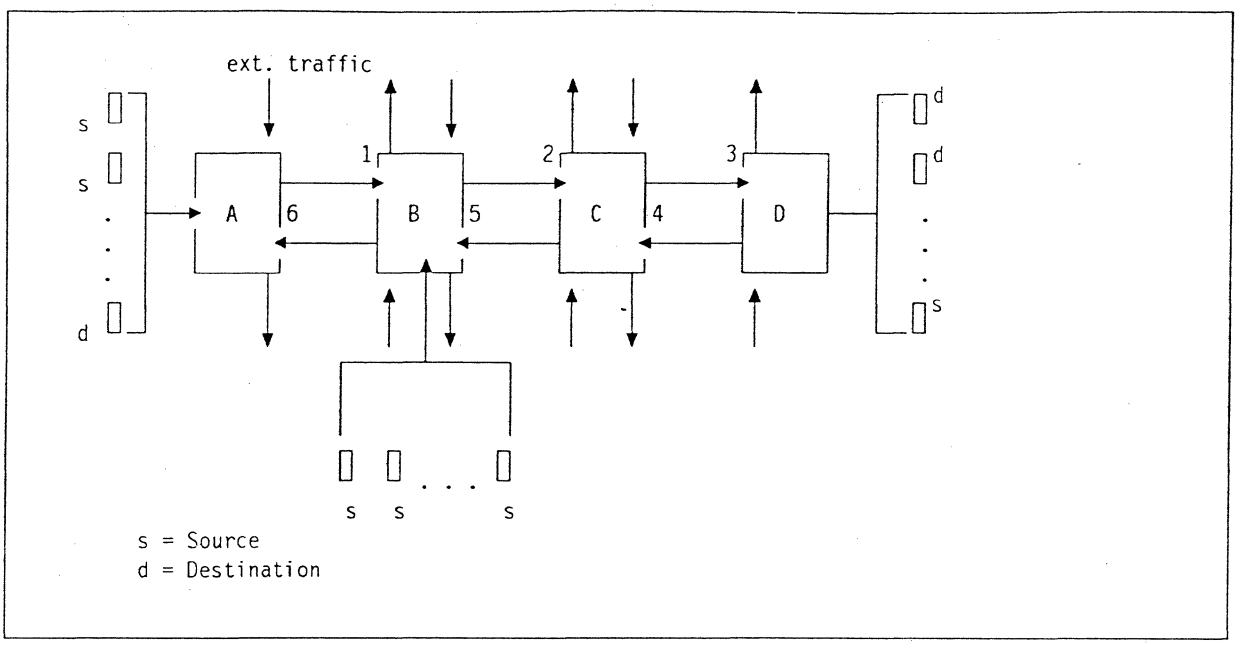


Figure 7-18. Simulation configuration

Extensive simulation has been conducted for different link capacities (e.g., from 50Kbps to T3), propagation delays, number of sender-receiver pairs etc.. In the following, we show some sample results of three configurations: 50Kbps, T1, and T3.

The parameters for all configurations are set as follows. The sensitivity threshold Δ_R is derived using eq. (5) described in 7.2.4.4.6, "Analysis of ARB parameters" on page 7-28. T_q is the sum of 4-packet transmission time at each link along a connection path. M is dynamically set based on the connection roundtrip delay which is estimated dynamically based on the algorithm described in 7.2.4.1.3, "Timers" on page 7-14. Δ_R is obtained as a fraction of the actual sending rate R_a (note again that $R_a \leq R_s$). θ is set to 0.9375.

50Kbps configuration

In this configuration, links 1 and 3 (and therefore 4 and 6) are given a capacity of 4Mbps which simulates a token-ring LAN. Link 2 (5) is given a capacity of 50Kbps. This is a LAN-WAN-LAN configuration. There are 2 connections originating from node A to node D (i.e., connection 1 and 2) and 2 connections in the reverse direction (i.e., connection 3 and 4), that is from node D to node A. The maximum packet length is 500 bytes (4000 bits). Each ACK data (i.e., ARB rate reply message) is 400 bit long. Each connection is an on/off process with an average transmission rate of 40Kbps, a peak rate of 50Kbps, and a mean busy period of 4000 bits. Two connections together create a demand of 80Kbps on the average, which exceeds the 50Kbps link capacity of the WAN. Therefore, the ARB algorithm will have to regulate the traffic and give equal share of throughput to each connection. Minimum R_{inc} is set at 500bps. The propagation delay is set at 0.1 ms for the 4Mbps ring, and 1 second for the 50Kbps. The measurement interval is initially set at 3 sec. but will dynamically adjusted based on the minimum roundtrip delay. The sender's burst size (B_s) is 8000 bits or 2 maximum sized packets. T_q can

be derived based on the maximum packet size and the link speeds, it is approximately 330 ms from node A to node D and visa versa (i.e., 4 packet transmission time at 4Mbps is 4ms, and at 50Kbps is 320ms). The initial rate of each connection is set to be 10% of the 50Kbps link.

The simulation was run for 2000 seconds. The following is a time diagram of the throughput (e.g., upper figure shows the sending rate and the actual rate of the connection with the actual rate being the dotted line) of connection 1 with others following the same patern, along with the utilization level of link 2 (i.e., the 50Kbps bottleneck link). The lower-part of the figure shows the transmission buffer occupancy at link 2.

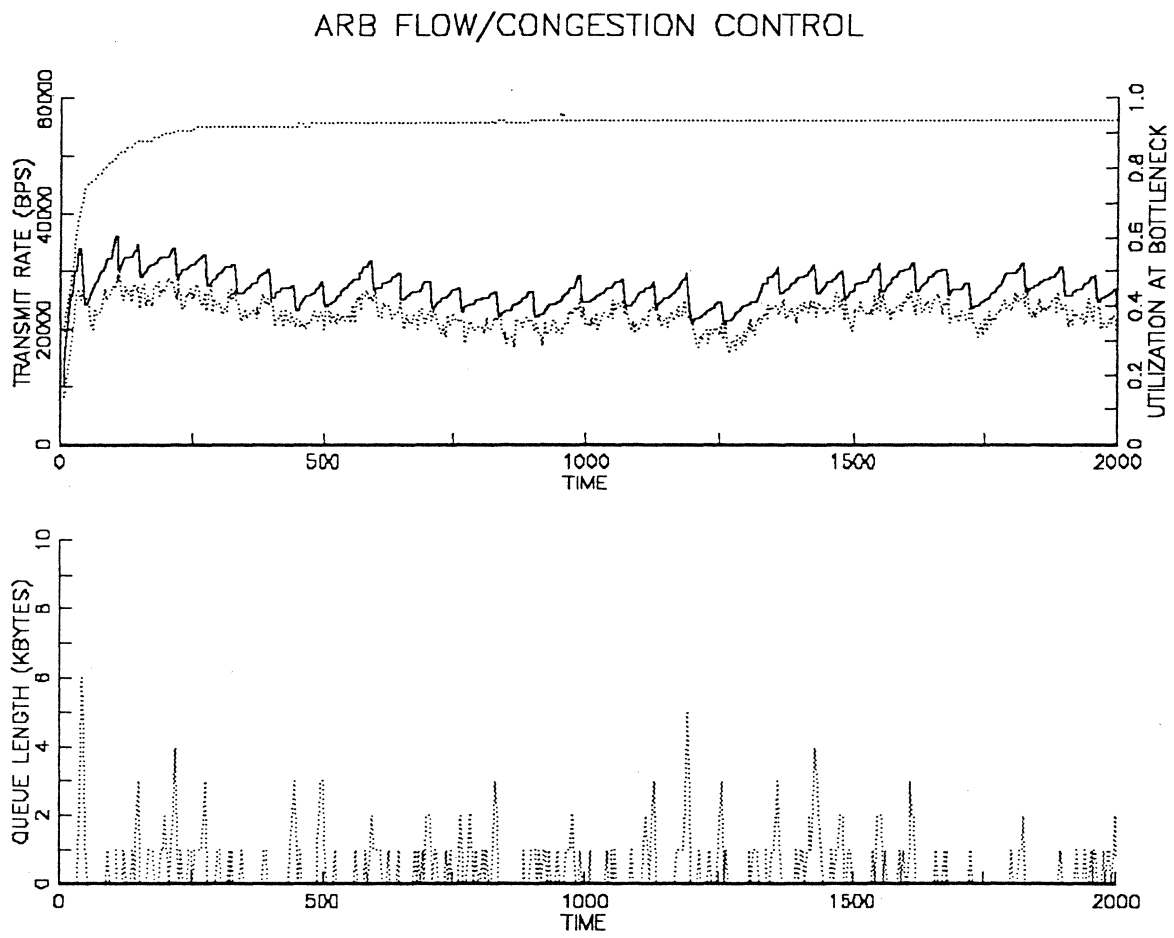


Figure 7-19. 50Kbps configuration - connection 1 send and actual rate, link utilization and transmission queue occupancy.

The next figure (Figure 7-20 on page 7-39) also shows a 50Kbps configuration with 2 connections from A to D with connection 2 being a short-live connection. The propagation delay of the 50Kbps link being 1s. Connection 2 is activated at time 200 and deactivated at time 1500. This configuration is to show the adaptivity to

network changes of the ARB algorithm in that connection 1 slow down after time 200 to share the throughput with connection 2. Once it detects that connection 2 is gone at time 1500, it increases its rate accordingly to take advantage of the available network capacity and to satisfy the input rate from the source (i.e., 40Kbps mean rate). The actual rates (dotted curves in both upper and lower graphs) of both connections are also shown.

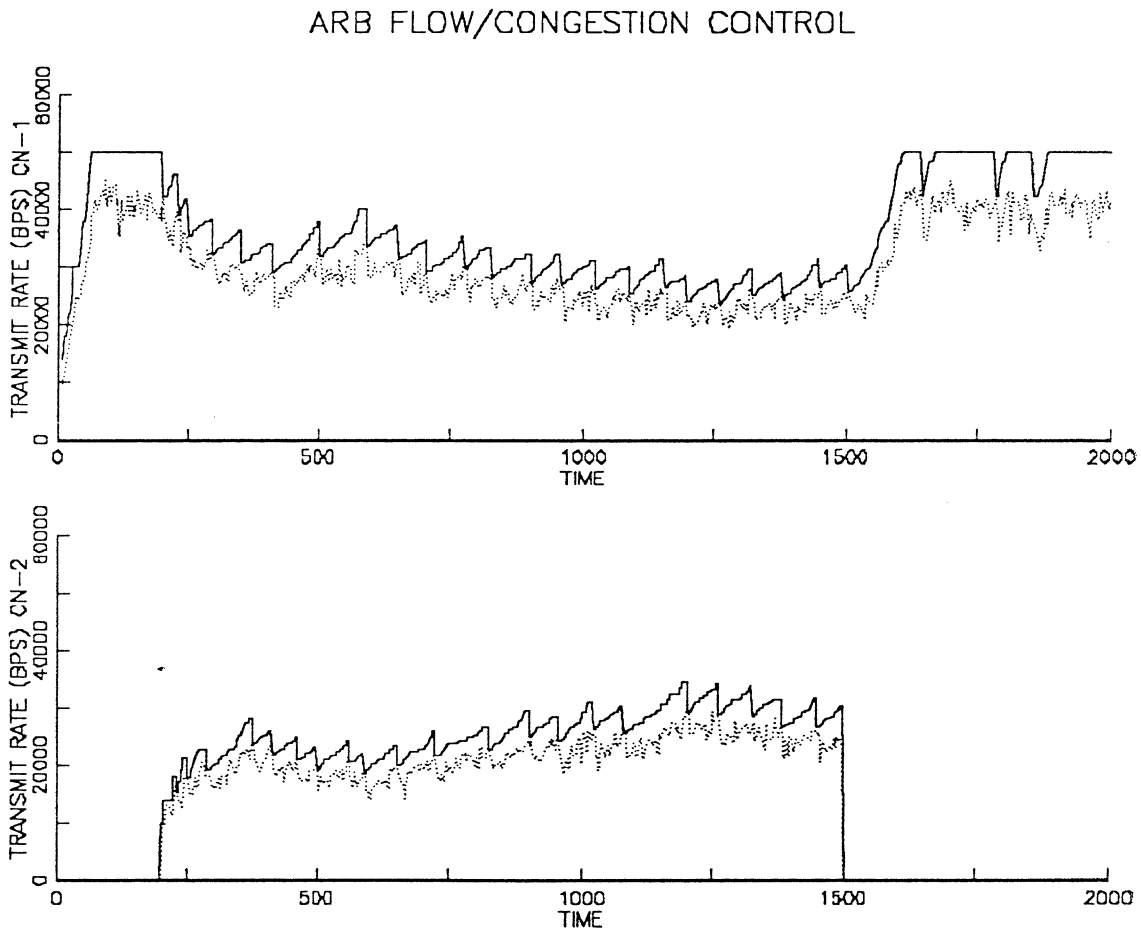


Figure 7-20. 50Kbps configuration - sending rates of connections activated at different times

T1 configuration

We again show the LAN-WAN-LAN configuration with WAN being a T1 link. The LAN has a 16 Mbps rate and a 0.1 ms propagation delay. There is a 30ms propagation delay for the T1 link (i.e., little more than crossing the US continent). The maximum packet size is 1K bytes, R_{inc} is set at 1.5Kbps, and 4 packets at each link are used in computing T_q. There are 10 connections, 6 of them (1-6) originate from node A and terminate in node D, these are long-live connections. The other 4 (7-10) originate from node B and terminate in node D and are short-live connection, that is they are activated at time 50 and deactivated at time 80. Each connection as a mean rate of 200Kbps, peak rate of 256Kbps, and a mean busy period of 8000 bits. Note that when all 10 connections are active, the total bandwidth requirement will exceed that of the T1 link. The initial rate of each connection is set to be 10% of the T1 link. In the figure, the upper graph shows the sending rate (actual rates are not shown in this figure) of one of the 6 long-live connections along with the buffer occupancy at the T1 link (dotted line). The lower graph shows the sending rate of one of the 4 short-live connections being active at time 50 and off at time 80.

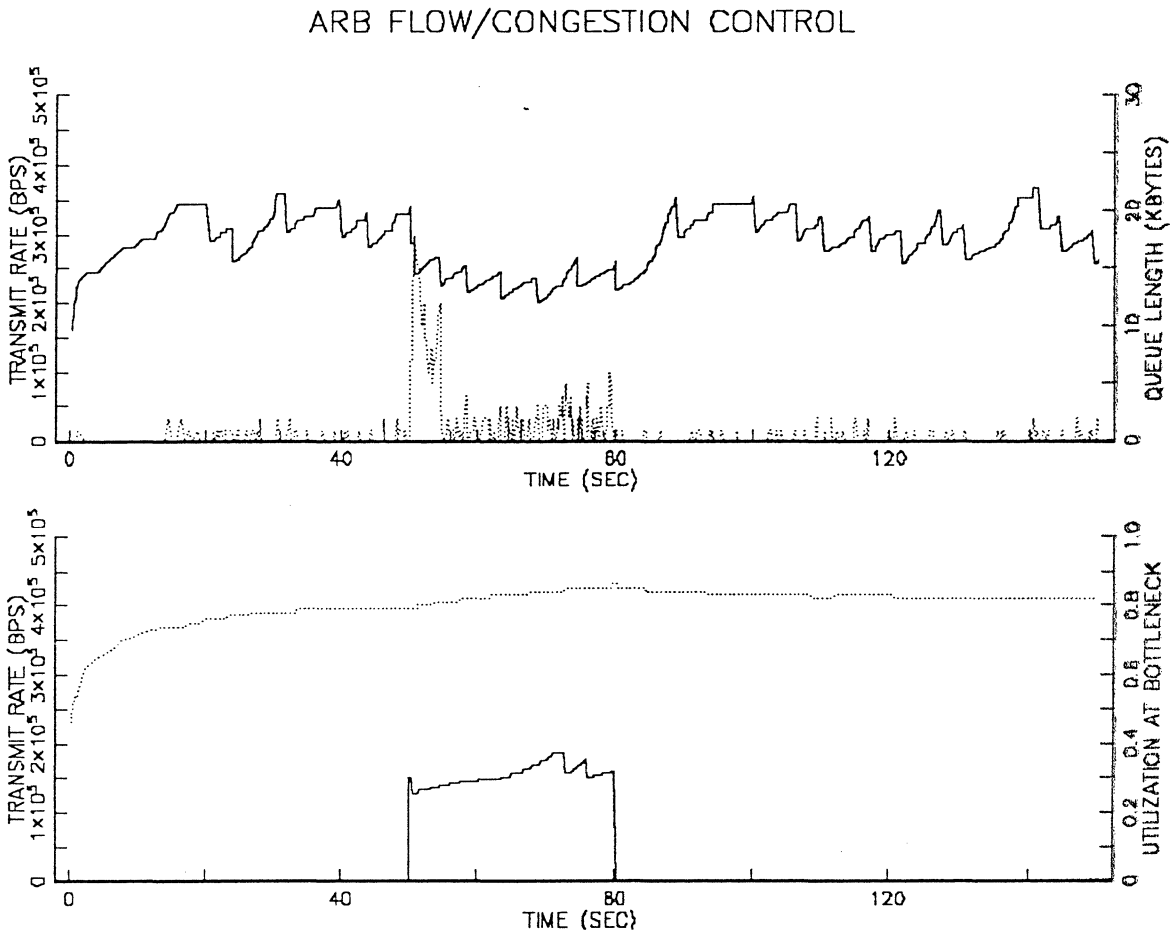


Figure 7-21. T1 configuration - sending rates, T1 link utilization and buffer occupancy

T3 configuration

Two variations are shown here for this configuration, they are LAN-WAN-LAN and LAN-WAN, with WAN being a T3 link with propagation delay of 30ms for the LAN-WAN-LAN configuration, and 2 T3 links with propagation delay of 20ms

each for the latter configuration. The LAN will be modeled as 100Mbps with propagation delay of 10 microseconds. Other parameters are set as follows:

- The maximum packet size is 1K bytes (8K bits).
- Each on-off traffic source/connection has a mean rate of 10Mbps, a peak rate of 16Mbps, and a mean busy period of 16K bits. There are 10 connections six of which are activated at time 0 and are active for the entire simulation time (500 seconds). The remaining 4 are activated at time 30 sec. and are active until time 300 in the LAN-WAN-LAN configuration, and time 400 in the LAN-WAN configuration.
- The number of packets used in computing T_q is 4 at each hop along the path.
- The initial rate for each connection is 4Mbps in the LAN-WAN-LAN configuration. This is to inject a sudden burst of traffic when the latter 4 connections are activated. In the LAN-WAN configuration, the first 6 connections are given an initial rate of 4Mbps, but the latter 4 are given an initial rate of 2Mbps instead. The objective is to observe the difference in the queue build up at time 30sec when all 10 connections are active.

Figure 7-22 on page 7-43 shows the sending rate of connection 1 (one of the first 6) in the LAN-WAN-LAN configuration. notice the rate adjustment at time 30 when the remaining 4 connections are activated. Also notice a sharp increase in the queue length at link 2. the bottleneck T3 link (the maximum transmission buffer occupancy at this link is a little over 60Kbytes), shown in the upper graph in the figure along with the utilization level. The rates are reduced immediately and a drastic reduction in the queue length can be observed. The send rates of the latter 4 connections are not shown in this figure but are comparable to that of connection 1.

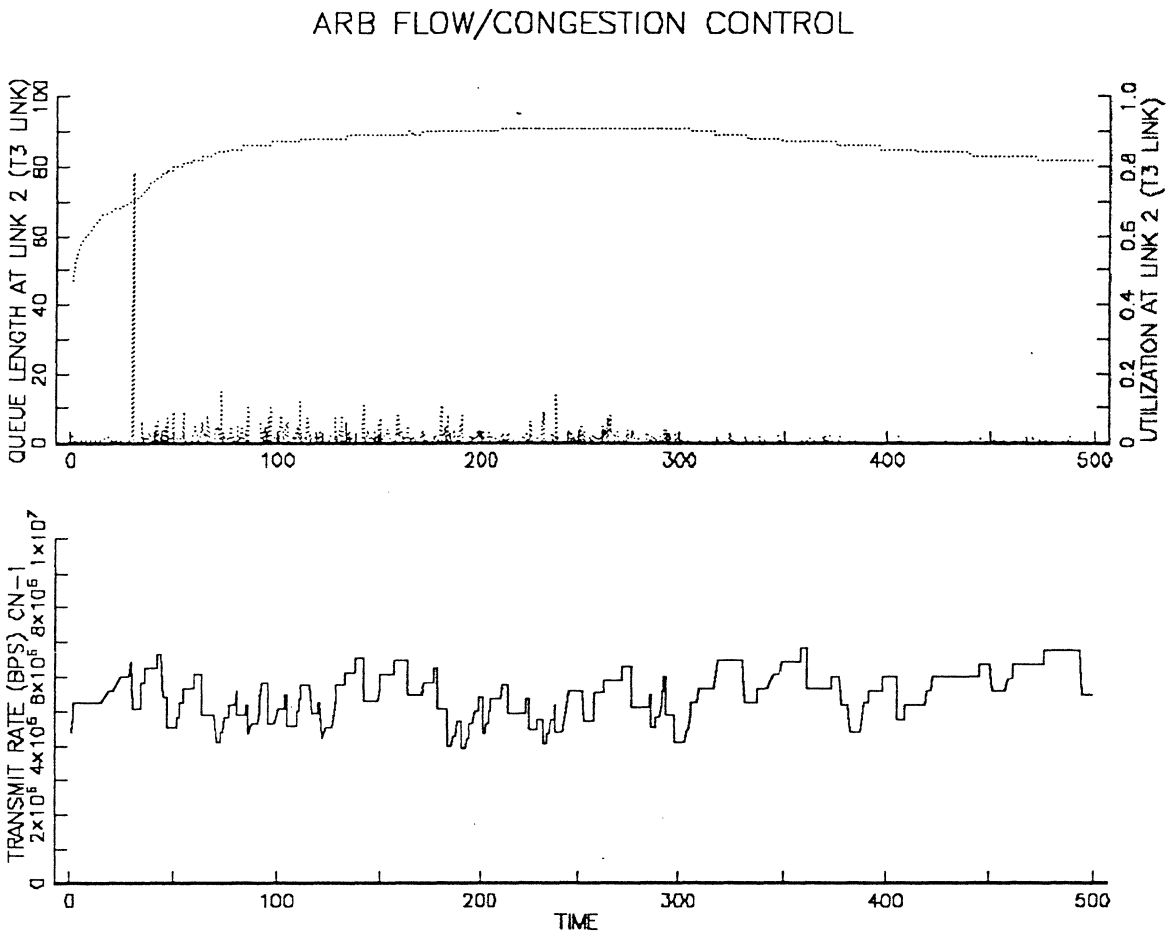


Figure 7-22. T3 configuration (LAN-WAN-LAN): sending rates, T3 link utilization, and T3 queue lengths.

Figure 7-23 on page 7-44 shows the sending rates of connection 1 and connection 10 (one of the 4 latter connections) in the lower and upper graph respectively. The utilization of link 2 is shown in the lower graph, and the queue length is shown in the upper graph. The big difference shown in this configuration is that the sharp increase in queue length at link 2 at time 30 is not high compared with the previous configuration. This is due to the smaller initial rate given to the latter 4 connections.

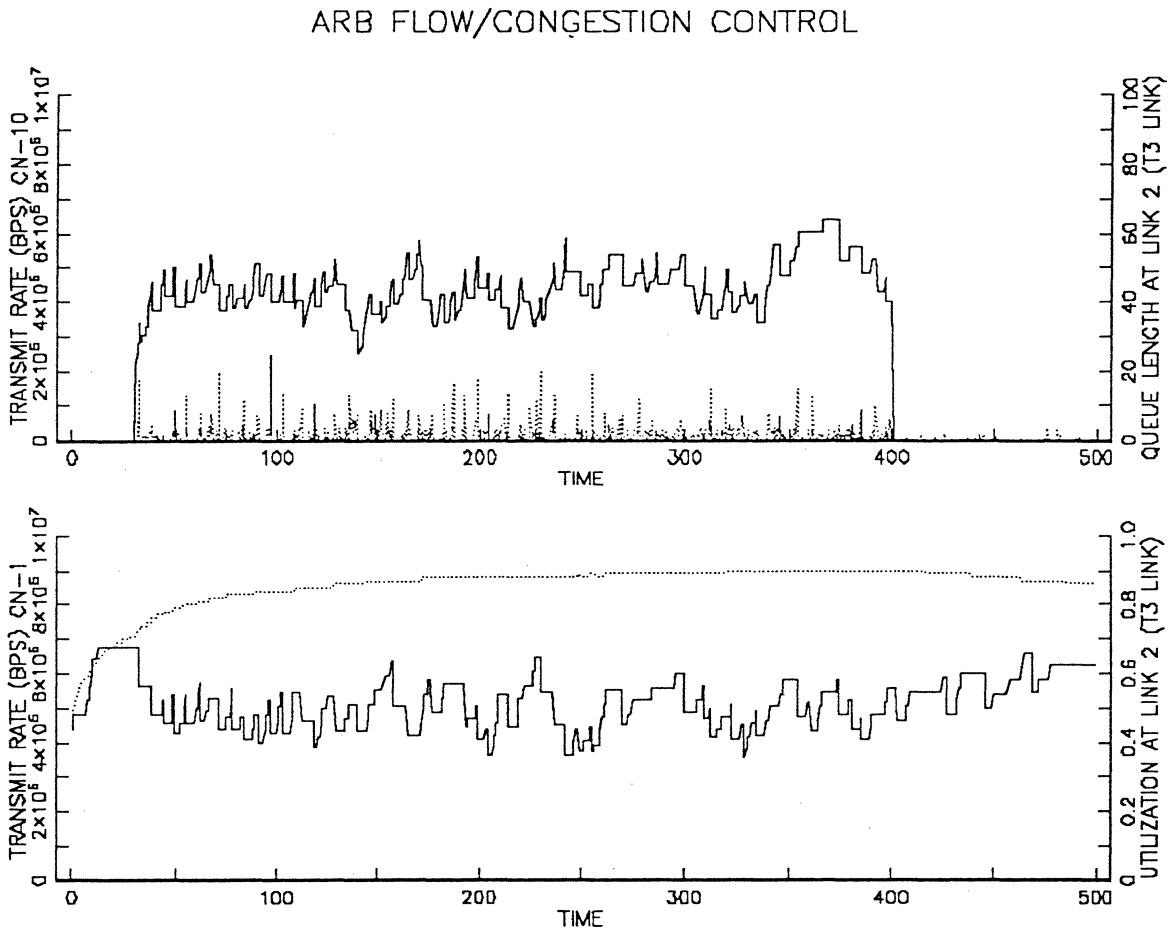


Figure 7-23. T3 configuration (LAN-WAN): sending rates, T3 link utilization, and T3 queue lengths.

7.2.4.5 Session Adaptive Pacing

In HPR, multiple sessions with the same COS/TPF are allowed to be multiplexed onto a single RTP connection. The ARB mechanism in RTP provides fairness among the RTP connections but does not provide fairness at the session level. Therefore, existing session pacing over an RTP connection is necessary to maintain fairness among sessions and prevent any session from using buffers unfairly as demonstrated in the following example:

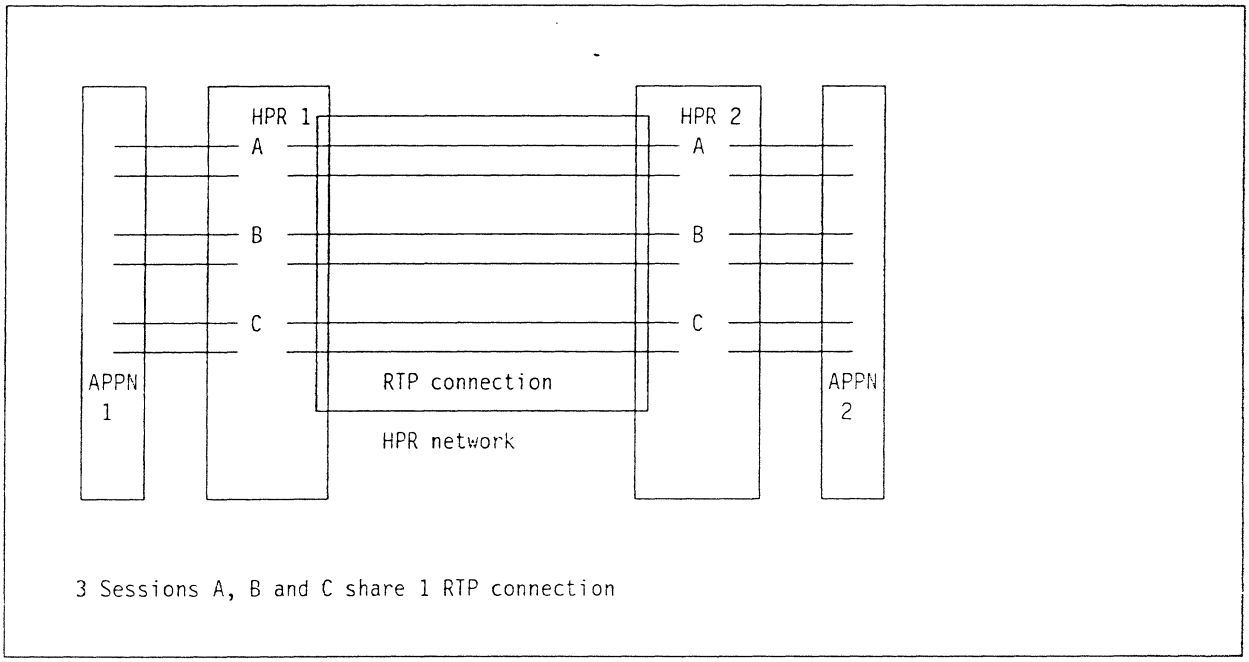


Figure 7-24. Session Pacing on Top of RTP

In this example, APPN node 2 withholds session-pacing window for session A while pacing for sessions B and C proceeds normally. Pacing for session A is normal between APPN node 1 and HPR node 1. Therefore, as data for session A arrives at HPR node 2, buffers for this RTP connection become depleted, and eventually the ARB mechanism stops transmission on this RTP connection. The result of this action is that sessions B and C will also be stopped. With session pacing in addition to ARB, this scenario will not occur because session A will be stopped by the normal pacing mechanism.

Since ARB already performs the flow control function on behalf of the sessions which share the connection, the session pacing should not be used to achieve the same result as it would result in the additional overhead of flowing IPMs on the RTP connection. Rather, session pacing should be used to protect the sessions sharing the connection from the situation described above in the worst case. This can be accomplished by setting large windows (i.e., worst case buffer space which can be occupied by a single session) for sessions running over RTP. In addition, the window size in the IPMs should immediately or quickly (i.e., rather than gradually as it might be implemented in the current APPN implementation) be incremented to take advantage of this large window size.

The session pacing between an HPR node and an APPN node (e.g., APPN 1 and HPR 1 in the above example) should be done in the same way as it is done between 2 APPN nodes. That is, when RTP/ARB detects saturation in the network, it will slow down the sending rates of the corresponding connections. As a result, packets entering the network on these connections will be queued, and the back-pressure mechanism should be enforced (i.e., holding IPMs to be sent back to the neighbor APPN nodes). In the configuration above, if the RTP connection for session A is slowed, IPMs from node HPR 1 to node APPN 1 for session A will either be held/delayed or the window size decreased.

7.2.4.6 BIND Pacing on top of RTP in HPR

In HPR, the BIND pacing mechanism will not be performed over an RTP connection. BIND pacing is not required due to the following considerations:

- BINDs no long require intermediate nodes to reserve resources for the sessions. Sessions are transported over RTP connections and session data is routed using ANR routing. This eliminates the potential deadlock situations that can arise in the network when nodes must wait for resources to be allocated for BINDs.
- When the local buffer resources are low, RTP will be notified to slow down the remote RTP partner and the flow of BINDs is slowed down as a result. If RTP receives a BIND but the condition of local resources does not warrant new session establishment, UNBINDs are returned with appropriate sense data.
- If local resources are depleted, RTP will discard packets including BINDs and force the remote partner to resend the data. Dropped packets cause the remote RTP partner to slow down significantly while attempting to retransmit them.

7.3 Path Switching

Path switching is performed for both CP-CP and LU-LU sessions when they flow over RTP connections. The path switching function is non-disruptive for the sessions.

See 8.4, "CP-CP Session Path Switch" on page 8-4 and 9.6, "LU-LU Path Switch" on page 9-10.

Path switching is not done for long-lived, route-setup RTP connections. Each route-setup connection is associated with a specific link and does not carry session traffic. See 9.2, "Route Setup protocol" on page 9-2.

7.4 Priority

HPR uses the same transmission priorities as the base APPN architecture (i.e., low, medium, high, and network). Transmission priority is associated with COS. APPN has a set of architecturally-defined COSs with each having a specified transmission priority. For example, the CPSVCMG COS, used by CP-CP sessions, has a transmission priority of network (the highest). COSs that are not architecturally-defined may be used for LU-LU sessions; these COSs are associated with the priority of the session, either high, medium, or low.

APPN intermediate network nodes provide priority queues to allow higher-priority traffic to pass lower-priority traffic. The transmission priority function is also provided for packets flowing on RTP connections. Each RTP connection is associated with a COS and, therefore, has an assigned transmission priority. See 7.1.6, "Priority Routing in Intermediate Nodes" on page 7-3.

7.5 Compression

The data compression function is the same as is currently defined by the base APPN architecture. The NHDR and the THDR are not compressed.

7.6 Encryption

The data encryption function is the same as is currently defined by the base APPN architecture. The NHDR and the THDR are not encrypted.

7.7 Security

No new security functions have been defined for HPR. Security in HPR is the same as is defined for the base APPN architecture.

7.8 Route Calculation

The APPN route calculation algorithm is used by HPR. HPR links are marked in the topology database, and the routes (paths) within HPR networks are specified with ANR labels instead of TG numbers and CP names. Nonetheless, the algorithm to compute these paths is unchanged.

The existing APPN architecture provides various means that could be used to make HPR links more preferred than regular APPN links. For example, HPR links can be made to have better characteristics in terms of cost, delay, etc. than APPN links even if the physical link characteristics are the same. Another possibility would be to use the user-defined fields in the COS tables and link characteristics to give HPR links smaller weights so that the route calculation algorithm is more likely to choose HPR links than APPN links. However, as the routing decision is not a local optimization (i.e., within a node) but rather a global one (i.e., within the network), a small change in a link's characteristics may change the whole network's distribution of traffic. As a result, by artificially lowering the weight of an HPR link, the network could route all its traffic through that link causing a network collapse. Therefore, when a node activates an HPR link, the link characteristics, broadcast as part of topology function, should accurately characterize the link, and the node should not artificially modify the link characteristics.

This architecture allows seamless migration from an APPN network to an HPR network. To take full advantage of the HPR function, a customer should plan to upgrade APPN nodes with HPR function in an unscattered manner (i.e., HPR-HPR-APPN is better than HPR-APPN-HPR). Links with the heaviest traffic (e.g., backbone links) should be added to the HPR network first.

7.8.1 HPR-Only Route For Path Switch

Upon detecting a time-out while attempting to transmit data or ALIVE messages, RTP will request a new path from route selection services (RSS). If the new path between the two RTP endpoints contains one or more APPN links (i.e., supporting only ISR function), the path switch will fail and the sessions traversing that RTP connection will be deactivated. This is not acceptable if there is a path between the two nodes that uses only HPR links. Because of the importance of session availability, such a HPR-only path is used even when it has a higher weight. The simplest way to compute an HPR-only route specifically for path switch is to include an HPR-only option when requesting a route from RSS. RSS is enhanced to implement this option. Specifically, RSS will compute a tree for the requested COS/TPF that consists only of HPR links from the origin node (i.e., the RTP endpoint that detected the failure) to the destination node. This is accomplished by giving non-HPR links infinite weight during the route computation process. The resulting HPR-only tree can be cached and reused for later path-switch operations. These HPR-only trees are not used to obtain routes for new sessions as they will likely produce higher-weight paths. Thus, there will be two sets of trees per COS/TPF in a mixed APPN and HPR network. But when a network is upgraded to contain only HPR nodes, the HPR-only tree and the normal session setup tree will be the same.

7.9 Enhanced Session Addressing

An enhanced addressing algorithm has been developed for sessions flowing over RTP connections. The LU[7] or BF at each RTP endpoint assigns a 4-byte session address to be used to identify the session data received from the partner LU or BF. Thus, there are two addresses (one in each direction) for each session flowing on the RTP connection. The new FID5 TH contains the session address. The format of the session address is described below.

The first bit (bit 0) in the 4-byte session address will be used as an indicator of whether the address was assigned in the local node or by the remote partner. If the first bit is set to 0, the address was assigned in the local node, and the session-related information (e.g., session control block) can quickly be accessed. If the first bit is set to 1, the address was assigned by the remote partner.

When a local LU or BF initiates a session, it must assign and place into the BIND PIU the address that will be used later by the remote partner LU or BF to return RSP(BIND), session data, UNBIND, etc. This address in the BIND PIU will have the first bit set to 1. Later, when the remote partner returns the RSP(BIND) and session data, this address will be used with the first bit set to 0 to indicate to the local partner that it selected the address. The remote partner will assign its own corresponding address for the session and return it in the RSP(BIND) (see the sample scenario described below.) Figure 7-25 on page 7-49 shows the format of the session address.

[7] The LU is the CP in the case of CP-CP sessions.

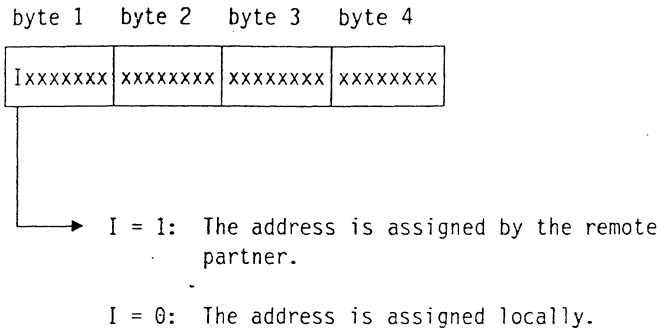


Figure 7-25. Enhanced session address format

As previously mentioned, there are two addresses associated with each session. Each LU or BF assigns the address to be used on the PIUs it receives. The PLU or the BF at the PLU-end of the RTP connection assigns an address and sends it to the SLU or the BF at the SLU end in the FID5 TH of the BIND PIU. The SLU or BF assigns an address and sends it to the PLU end in the Session Address (X'65') control vector of the RSP(BIND). Data sent from the PLU to the SLU will have the address assigned at the SLU end carried in the FID5 TH. Data sent from the SLU to the PLU will have the address assigned at the PLU end carried in the FID5 TH. See Figure 7-26 on page 7-50. Of course, all the session traffic is carried on an RTP connection (not shown in the figure).

Since multiple sessions of the same COS can be multiplexed onto a single RTP connection, the enhanced session addresses must at least be unique per RTP connection. (The ANR labels for the NCEs must be unique per node, the transport connection identifiers [TCIDs] must be unique per NCE, and the session addresses must be unique per TCID.) This uniqueness will allow RTP connection endpoints to demultiplex session addresses based on TCID and the individual session addresses associated with that RTP connection.

Session addresses can be reused once the session has been deactivated. In APPN the reuse of session addresses must be done with care because a BIND which is sent at network priority can get to the remote node before an UNBIND which is sent at session priority. However, BINDs and UNBINDs are sent FIFO (with session priority) by RTP. Therefore, RTP connections do not have an APPN-like race condition, and as a result, the enhanced session addresses for a particular TCID can be reused once the UNBIND for a session address has been forwarded to RTP for transport to the partner.

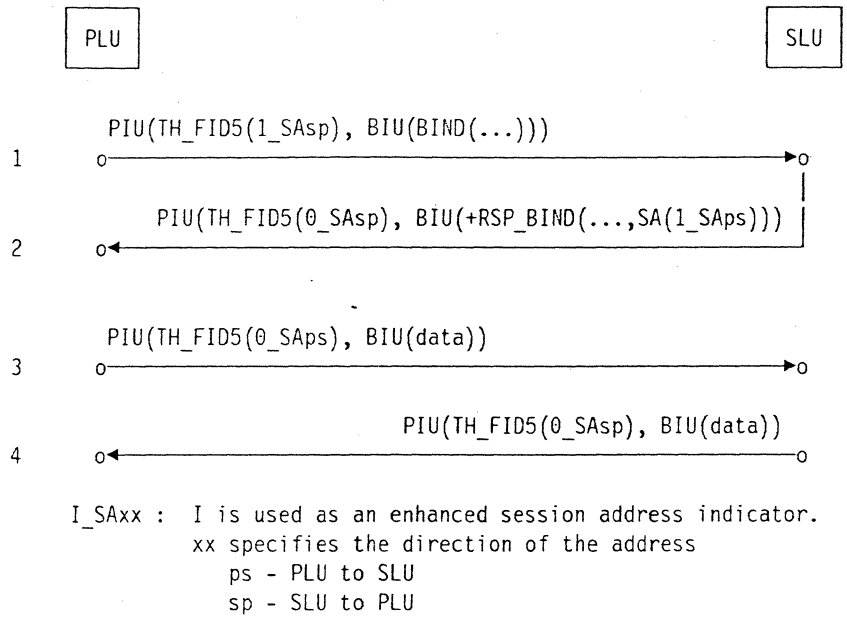


Figure 7-26. Enhanced session addressing protocol

Annotations:

1 The PLU assigns a session address (1_SAsp) to be used for data flowing from the SLU to the PLU and sends it (in the TH) for the BIND. Note that 1 indicates that the first bit in the session address is set to 1. When the SLU receives the BIND, it knows that the session address 1_SAsp is for a new session and is not associated with an existing session (i.e., the address is not locally assigned, and the SLU does not try to do a look up for an existing session with a matching address).

2 The SLU assigns a session address (1_SAps) to be used for data flowing from the PLU to the SLU and sends it in the RSP(BIND). The TH for the RSP(BIND) still contains the same address as received on the BIND with the first bit set to 0 (because the PLU will know that it is a locally assigned session address and use it in the process of correlating the RSP(BIND) with the BIND).

3 Data sent from the PLU to the SLU uses 0_SAps in the FID5 TH. Note again that the first bit is set to 0.

4 Data sent from the SLU to the PLU uses 0_SAsp in the FID5 TH. Note again that the first bit is set to 0.

Chapter 8. CP-CP Sessions

CP-CP sessions in HPR are used the same as in APPN. Some modifications have been made to CP-CP sessions for HPR nodes that support the HPR Transport Tower. Otherwise, no changes have been made.

For HPR nodes that support the HPR Transport Tower, the CP-CP session traffic flows over RTP connections These RTP connections are referred to as CP-CP RTP connections. For HPR nodes that do not support the Transport Tower, CP-CP session traffic is carried in FID2 PIUs (just as in today's APPN).

8.1 Activation

CP-CP session activation is triggered in HPR the same as in APPN (e.g. when links are activated, etc.). Contention winner and contention loser LU 6.2 sessions are activated between the CPs. If both CPs (nodes) support the HPR Transport Tower, either one or two CP-CP RTP connections are activated. If both sides activate simultaneously then there will be two CP-CP RTP connections, one for each CP-CP session. However, if one side (CPa) manages to activate a CP-CP RTP connection and the partner (CPb) recognizes this before activating a second CP-CP RTP connection, then CPb will use the RTP connection activated by CPa. In this case, traffic for both CP-CP sessions will be carried by the one CP-CP RTP connection.

Rules for mapping CP-CP sessions to RTP connections are the same as for mapping LU-LU sessions to RTP connections (see 7.2.1, "Relationship of Sessions, COS, and RTP Connections" on page 7-4 for more details).

Figure 8-1 on page 8-2 shows the CP-CP RTP connection activation flows (including CP capabilities) between HPR nodes that both support the Transport Tower. See Chapter 7, "Common Session Support (CP-CP and LU-LU)" on page 7-1 for more details about RTP connection activation.

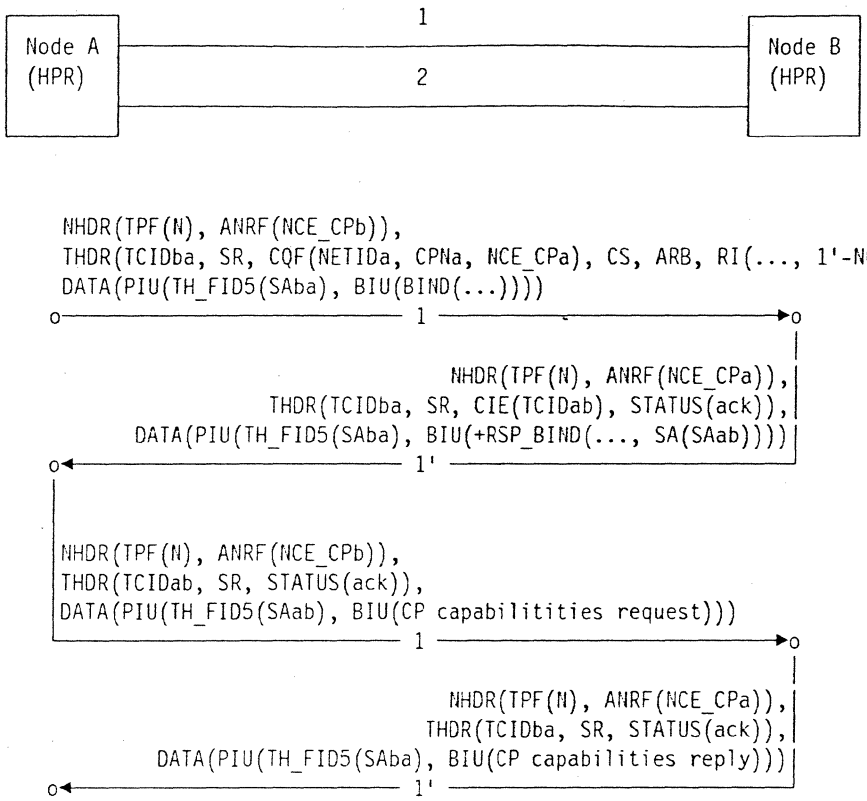


Figure 8-1. CP-CP RTP Connection Activation including capabilities exchange

Annotations:

1 The RTP connection activation (CONNECTION_SETUP(CS)) and session activation are carried on the same flow. The NCE_CPb in the ANR routing field (ANRF) has previously been obtained at during link activation on XID3.

2 The NCE_CPa in ANRF has previously been obtained at during link activation on XID3.

3-4 CP capabilities flows on the RTP connection as data.

8.2 Deactivation

CP-CP session deactivation is triggered the same as in APPN (on protocol errors, etc.). The CP-CP session UNBINDs flow over the RTP connection just like any other data.

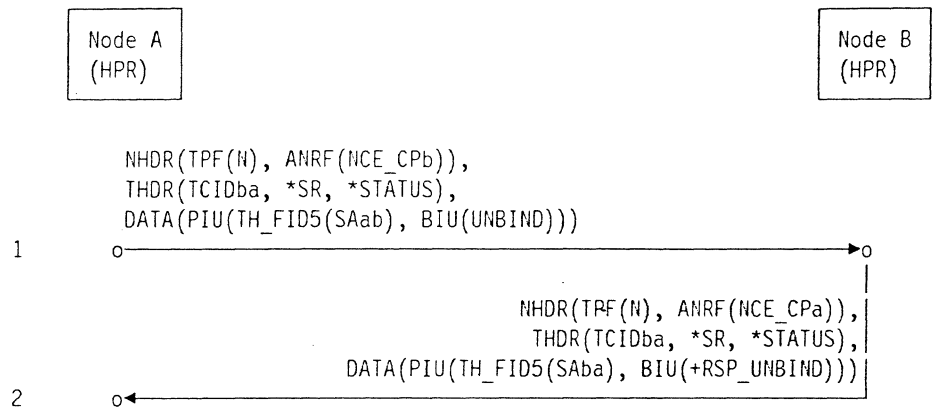


Figure 8-2. CP-CP session deactivation

8.3 Session traffic

CP-CP session traffic includes:

- CP capabilities
- Directory searches
- Topology information

The CP capabilities flows are shown in Figure 8-1 on page 8-2 and the others sre described in the following sections.

8.3.1 Directory Searches

Directory search protocols in HPR are the same as in APPN except for the following:

- HPR nodes that contain a target LU (i.e. one being searched for) will include that LU's NCE on the search reply in a new control vector (see A.1.12, "NCE identifier CV (CV62)" on page A-28). See Chapter 9, "LU-LU Sessions (HPR-HPR)" on page 9-1 and Chapter 10, "LU-LU Sessions involving APPN nodes (APPN/HPR Boundary Function)" on page 10-1 for more details as to how this LU's NCE is used.
- End node TG vectors (tail vectors) indicate their HPR capability via the TG type field in the TG Descriptor control vector (see A.2.7, "TG Identifier TG Descriptor Subfield (X'4680')" on page A-39). These TG Descriptor CVs may later become part of an RSCV where the TG type fields are used to understand the HPR capability of links and nodes along the path.

Figure 8-3 on page 8-4 shows the HPR NLP for a directory search flowing on a CP-CP RTP connection.

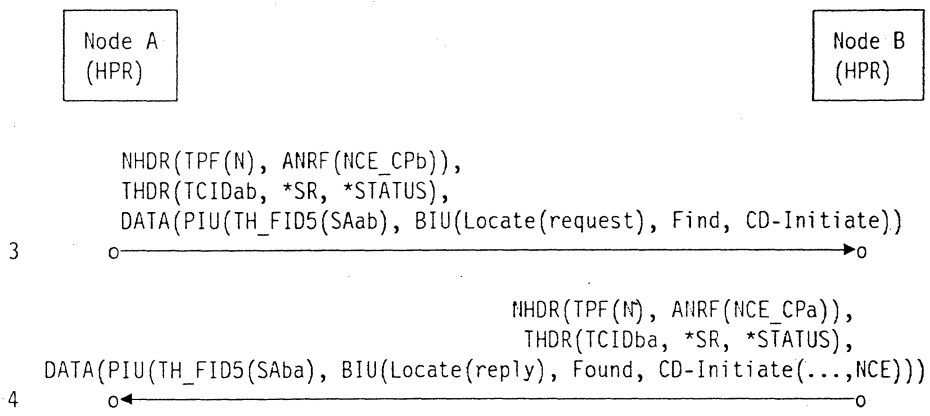


Figure 8-3. Directory search request and reply flowing over CP-CP RTP connection

8.3.2 Topology

The topology formats and protocols in HPR are the same as in APPN except an indication of the HPR capability has been added to the TG type field in the TG Descriptor control vector (see A.2.7, "TG Identifier TG Descriptor Subfield (X'4680')" on page A-39). The TG type field of the CV4680 is stored and propagated by APPN nodes.

Figure 8-4 shows the HPR NLP for a TDU flowing on a CP-CP RTP connection.

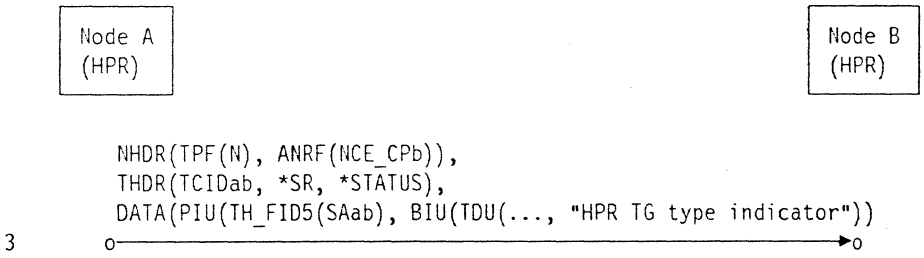


Figure 8-4. TDU flowing over CP-CP RTP connection

8.4 CP-CP Session Path Switch

When a link carrying a CP-CP RTP connection fails, that connection is rerouted, if possible, to another link that is capable of supporting CP-CP sessions. Since CP-CP RTP connections are always between adjacent nodes, the Route Setup protocol is not done. All the necessary ANR/RTP related information about the "single-hop" link has been obtained at link activation time via the XID3 exchange (ANR labels, maximum packet sizes, ...). In order to switch paths the best alternative HPR link is chosen and then used for sending and receiving data by the RTP connection.

See 9.6, "LU-LU Path Switch" on page 9-10 for more information about path switch operation.

Chapter 9. LU-LU Sessions (HPR-HPR)

This chapter describes LU-LU session operation within an HPR subnet. Please refer to Appendix B, "Sequence Notation" on page B-1 for an explanation of the notation used in the sequences in this chapter.

9.1 Session Activation

Activating an LU-LU session within an HPR subnet is done as follows:

- A directory search is performed to locate the target LU.

  The directory search formats and protocols are identical to those for today's APPN except:

  - HPR-level nodes provide the NCE associated with the target LU on the directory search reply.

    This LU NCE is used as the final destination ANR label (last ANR label in the ANR routing field) when sending to this LU. When the destination node containing the LU receives a message it uses this LU NCE to internally route it to the component that processes received messages for this LU.

- Perform a route setup (if necessary)

  Once the RSCV is obtained (either locally computed or received on the directory search reply) the desired route is known. If routing information is needed for this desired route, then a Route Setup protocol is performed. The route setup protocol obtains routing information such as forward and reverse ANR label strings and the largest packet size that may be sent over the route. This information is used for ANR routing and RTP connection operation. See 9.2, "Route Setup protocol" on page 9-2 for more information.

- Activate RTP connection (if necessary)

  An RTP connection is activated to carry the LU-LU session traffic. Activating an RTP connection involves the origin sending an RTP Connection Setup message to the destination. The BIND request is carried in the data portion of the Connection Setup message. When the destination receives the Connection Setup message it establishes an RTP connection and then processes the BIND request. It then sends a Connection ID Exchange message (carrying the TCID to be used by the origin when sending RTP traffic to this destination) which may carry the BIND response as data. The Connection ID Exchange message may also acknowledge the successful receipt of the Connection Setup message. The origin node sends an acknowledgment to the destination to acknowledge receipt of the Connection ID Exchange message.

  It may be possible to use an already active RTP connection for the LU-LU session in which case it is not necessary to activate another one. In this case the BIND is simply sent as normal data over the existing RTP connection.

The BIND request and response use a new FID5 TH in order to use HPR's enhanced session addressing (see 7.9, "Enhanced Session Addressing" on page 7-48).

9.1.1 BIND size

The same limitations that apply to the BIND in APPN also apply in HPR (i.e., the BIND RU is still limited to 512). HPR does not remove or alleviate these restrictions.

9.2 Route Setup protocol

The route setup protocol is initiated whenever it's necessary to obtain information about a route (between an origin and destination node) over which an RTP connection will be established. The protocol consist of a Route Setup request and a Route Setup reply. The Route Setup request is sent by the origin node (initiator) to the destination node over the exact route that is to be used. It stops at each intermediate node along the way to gather information associated with the forward path. The Route Setup reply is returned by the destination node after receiving the Route Setup request. The reply follows the same path as the request (in the reverse direction) and stops at each intermediate node along the way to gather information about the reverse path. The destination node maintains no memory of the information it received on the Route Setup request. When the origin node receives the reply it uses the information obtained in the appropriate manner (e.g. to establish a new RTP connection or reroute an existing one).

9.2.1 Route Setup path

The Route Setup request and reply follow the exact path they are gathering information for. There are several reasons for this:

- Allows off-loading of processing into the link adapters

  Knowledge about a particular link may be localized in the link adapter. This adapter may process and update the Route Setup before sending it out over the link to the next node.

- Checks that the links along the route are active and, if not, activates them.
- Simplicity: following the exact path to be used is a simple concept.

9.2.2 Directing the Route Setup

The determination of where to send the Route Setup is based on the RSCV that is associated with the BIND request that is triggering the RTP connection activation. Each TG within the RSCV indicates whether the link is capable of carrying HPR traffic and whether the TG goes to an HPR node that supports the Transport Tower or not. The Route Setup request will be sent to the furthest node along the path that supports the HPR Transport Tower with the additional constraint that all TGs between the node initiating the Route Setup and the furthest node are HPR capable. Some examples are shown in Figure 9-1 on page 9-3. In all the examples, the RSCV specifies the route 1-2-3-4, but the capabilities of the links differ. For instance, in Example 1, TGs 1, 2, and 3 go to HPR nodes that do not support the HPR Transport Tower. TG 4 goes to a node that supports the HPR Transport

Tower. The Route Setup flows over all the links 1, 2, 3, and 4 to the last node which supports the HPR Transport Tower.

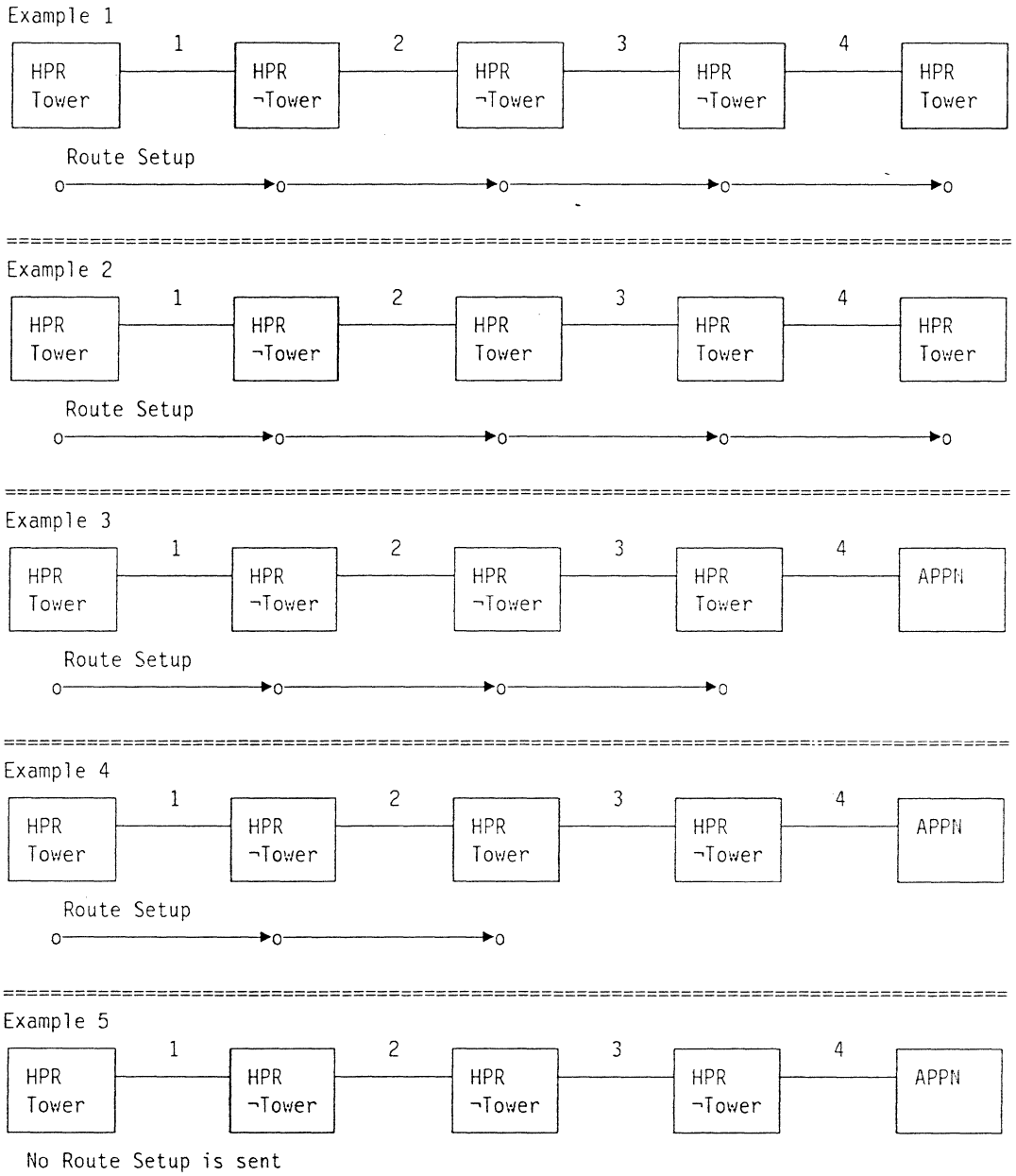


Figure 9-1. Directing Route Setup Examples

The Route Setup request is directed along the proper path using the RSCV in the same manner as a BIND hops along in today's APPN. The last hop is indicated by a "last hop count" field in the Route Setup request.

9.2.3 Automatic activation of links

If the Route Setup encounters a link which is not currently active but is activable on demand (e.g. a switched line that is currently inactive but has not been reported inactive in topology), then it is activated prior to forwarding the Route Setup. The Route Setup causes activation of links exactly as BINDs do in today's APPN.

9.2.4 Route Setup priority

The Route Setup request and reply flow at network priority.

9.2.5 Route Setup Format

The Route Setup fields are defined in a new GDS variable (see A.1.15, "Route Setup GDS variable X'12CE'" on page A-31). However, this GDS variable may be carried in a FID2 PIU or NLP. If both nodes support the HPR Transport Tower then it's carried in an NLP; otherwise, it's carried in a FID2 PIU.

9.2.6 Route Setup Operation for Nodes that Support the HPR Transport Tower

When both sides (nodes) of the link support the HPR Transport Tower, the Route Setup messages are sent in NLPs over long-lived RTP connections. The NCEs associated with these long-lived connections are exchanged on XID3 at link activation time.

9.2.6.1 Long-lived RTP Connections

The Route Setup requests and replies are transported reliably over the appropriate link using a previously established long-lived RTP connection. Transporting the Route Setup messages reliably means RTP requests the appropriate status (acknowledgements) for Route Setup requests and replies. (These acknowledements are generally not shown in the sequences). RTP connections are necessary for reliable transport because link-level error recovery may not be present.

Using long-lived connections results in improved performance since connections need not be activated and deactivated for each Route Setup message.

When a link is activated, the node with the higher CP name (the comparison of the CP names is done exactly as in today's APPN) activates the RTP connection and the connection remains active as long as the link remains active. Because of this, the Route Setup RTP connections are sometimes referred to as "long-lived". Route Setup RTP connections are established using a unique globally-defined topic identifier of "RSETUP" (see A.1.6.1, "Connection Setup (CS) Segment" on page A-10). All HPR links have an RTP connection to transport the Route Setup messages. Because each link has a Route Setup RTP connection, path switching of these connections is unnecessary. Figure 9-2 on page 9-5 shows the activation of a Route Setup RTP connection.

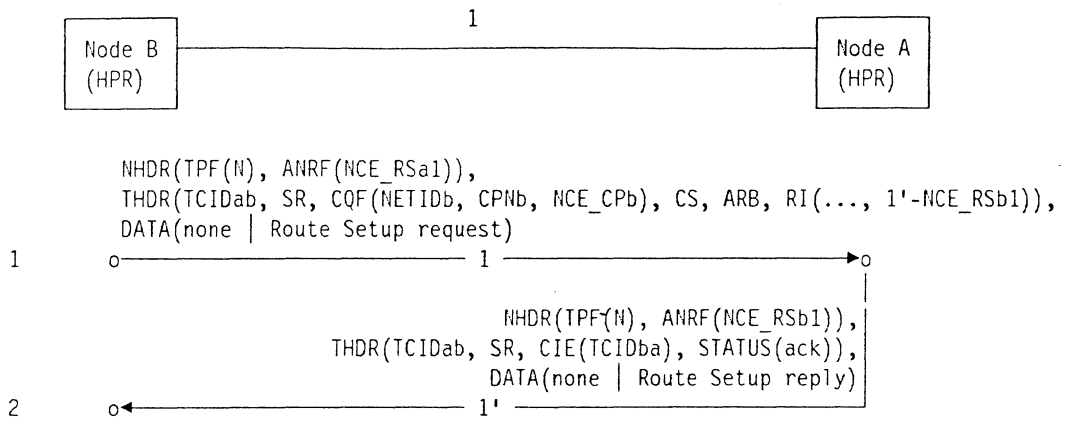


Figure 9-2. Route Setup Long-lived RTP Connection Activation

Annotations:

1 After link 1 has been activated, Node B (the node with the higher name) sends a Connection Setup (CS) RTP message to Node A. The Route Setup NCE identifier in node A for link 1 (NCE_RSa1 in ANRF) was previously obtained by node B during the link activation XID exchange. If the link activation was triggered by the need to send a Route Setup request, then the Route Setup request is carried in the DATA; otherwise, there is no data.

2 Node A may send the Route Setup reply (in the DATA) on the Connection Id Exchanged (CIE) message if it is available.

9.2.6.2 Route Setup NCEs

A Route Setup NCE is exchanged whenever an HPR link is activated (it is carried in the XID3). Assignment of Route Setup NCEs is implementation dependent. A node may assign one per link, one per group of links or, one per node. This NCE is used in the ANR routing field for all traffic sent on the long-lived Route Setup RTP connection.

9.2.7 Route Setup Operation for Nodes that do not Support the HPR Transport Tower

The Route Setup messages are sent in FID2 PIUs when both sides (nodes) of the link support HPR but one or both do not support the HPR Transport Tower. The FID2 PIU for Route Setup is a Network Control RU category so it can be easily distinguished from other FID2 PIUs. See A.1.14, "FID2 Route Setup" on page A-30.

9.2.7.1 Why Route Setup is carried in FID2

Route Setup requests and replies must be sent reliably. The choices are to use either link-level error recovery or an RTP connection. Since nodes not supporting the HPR Transport tower are already required send FID2 CP-CP session PIUs reliably using link-level error recovery (i.e. as in today's APPN), link-level error recovery was chosen. This choice also provides two other advantages. The first is that it's simple to understand that FID2 PIUs always require reliable transport (as opposed to having NLPs handled differently based on the message they are car-

rying). The second is that nodes not supporting the HPR Transport Tower do not have to implement RTP.

9.3 Connection networks

Connection network (CN) purposes and concepts are the same in HPR as they are in APPN (e.g. minimize sysdef, minimize the number of topology updates). The rest of this paragraph summarizes CN operation in today's APPN. In order to determine the real partner node (as opposed to the virtual node) on a CN it is necessary to have the "address" of that partner's port. For example, the port "address" is the adapter address for Token Ring LAN CNs. This port "address" (also called DLC data) is distributed on topology updates for the virtual link connecting the port to the virtual node. When calculating a route, the "address" is included in RSCV TG Descriptor CVs that pass through CNs. When an intermediate node receives a BIND and the next hop passes through a CN, a real link connection is activated (if one does not already exist). The "address" in the TG Descriptor CV is used to address the port of the real partner node to which the link connection is to be activated. After successful activation, the link is associated with a session connector and the BIND is forwarded on.

HPR uses ANR routing and RTP connections in place of session connectors. Because of these differences, some changes are required for CN operation in HPR subnets.

One simple way for HPR to route across CNs would be to include all the necessary information about the route (including the "addresses") in the network header ANR routing field. The problem with this is that the ANR routing field would become too large. Having long network headers on every message passing through the HPR subnet CN is undesireable because it adds additional price/performance costs that the customer must pay. The HPR solution to this problem is to activate the real link connection across the CN (and obtain ANR labels during activation) prior to doing ANR routing across it. Having to send a Route Setup request over a CN triggers the real link activation.

So, if a Route Setup route passes through a connection network and a TG is not currently active across it to the appropriate partner, then the TG and associated long-lived Route Setup RTP connection are activated. Once the TG is active, the associated information (like ANR label) is added to the Route Setup request and the request is forwarded on.

Connection networks are one of reasons the Route Setup protocol is needed. Since TDUs are not sent for connections across CNs, ANR labels (and other information) associated with those connections are not known at route calculation time. In fact, they can't be known until the actual link is activated. The Route Setup request causes the link to activated and obtains the ANR labels, etc.

9.4 Session data traffic and UNBIND

Session data traffic and UNBIND flow over an RTP connection as normal RTP data. The RTP indicators are set independently of whether it is transporting data or UNBIND. A count of the number of active, pending active, and pending deactive sessions is kept for each RTP connection. When this count goes to zero, the RTP connection may be deactivated. See 7.2.3, "RTP Connection Deactivation" on page 7-9 for further details.

9.5 LU-LU Session Sequences

Figure 9-3 on page 9-8 shows a sample sequence of an LU-LU session between HPR level nodes all of which support the HPR Transport Tower. If the intermediate nodes (NNb, NNc, and NNd) did not support the HPR Transport Tower the only difference in the sequence would be that the Route Setup request and reply would be carried in a FID2 PIU rather than an NLP.

Figure 9-3. LU-LU Session : HPR -- HPR

Conditions:

A If an RTP connection exists with matching RSCV(1-2-3-4), COS, NCE_LUa, and NCE_LUe then use the existing RTP connection to transport this session (i.e. go to step 13).

B If an RTP connection exists with matching RSCV then use routing information associated with the existing RTP connection to activate a new RTP connection (i.e. skip the route setup protocol).

Annotations:

1 A directory search request (directed or broadcast) is sent exactly as in APPN.

2 Since Node c is an HPR node, it assigns and returns an NCE for LUe (NCE_LUe) on the search reply. Everything else is the same as in APPN.

3 A Route_Setup request is sent to node b over the long-lived route setup RTP connection. Node a fills in the appropriate information for the forward ANR (1).

4 Node b fills in the appropriate information for the forward ANR (2), updates the current hop count in the RSCV (to 2) and forwards the Route_Setup request on to node c.

5-6 Nodes c and d do processing as was done at node b in step 4.

7 Node e returns a Route Setup reply. This reply now contains all the necessary information about the forward path. Node e adds information about the reverse ANR (4').

7-9 Node's d, c, and b add information about the reverse ANR path. After node b adds information about link 1' the reverse path information is complete.

10 Node a now has all the information associated with the route between node a and node e.

11 Node a activates a new RTP connection to node e (by sending a Connection Setup(CS) segment). The NHDR ANR field (ANRF) contains the forward route ending with the NCE associated with LUe (2-3-4-NCE_LUe). The THDR reverse ANR segment field (RANR) contains the reverse route and it ends with the NCE associated with LUa (4'-3'-2'-1'-NCE_LUa). The BIND request is carried as data with the Connection Setup (CS) with the current hop count (CHC) in the RSCV updated appropriately.

12 Node e responds with a Connection ID Exchange (CIE) to establish a TCID (TCIDae) to be used for data sent from node a to node e over this RTP connection. The CIE message also acknowledges (i.e. STATUS(ack)) that the Connection Setup was received successfully. The BIND response is carried as data along with the CIE and it includes the HPR enhanced session address (SAae) to be used on the session when sending data from node a to node e.

13 Node a sends the BIND on an existing RTP connection. The BIND is carried as data with the RSCV current hop count updated appropriately.

14 The BIND response is carried on the existing RTP connection and contains the enhanced session address (SAae) to be used when sending session traffic from node a to node e.

15-16 Session traffic flows over the established RTP connection and includes UNBINDs as well as session data. The appropriate TCIDs and SAs are used.

9.6 LU-LU Path Switch

The HPR path switch function is used to automatically reroute RTP connection traffic around a failed link or node. This function only operates within an HPR subnet. It does not operate within or across an APPN subnet. When a failure occurs and an alternate path (consisting of all HPR capable links) exists that satisfies the class of service (COS), the RTP connection traffic using the failed path is rerouted over the new alternate path. It will be done in such a manner as to be transparent to the sessions being carried over the RTP connection (i.e., it is non-disruptive to the sessions).

9.6.1 Who initiates a path switch

Which RTP partner initiates the path switch depends on the partner types. There are two types of RTP partners with respect to path switch initiation: controllers (those that prefer initiating the path switch) and non-controllers (those that will yield to the partners wishes). There are three possible combinations:

1. Both partners are non-controllers

   In this case, both partners will initiate path switches when necessary. Each partner is "active".

2. Both partners are controllers

   In this case, both partners will initiate path switches when necessary. Each partner is "active".

3. One partner is a controller and the other is a non-controller

   In this case, the controller will initiate path switches when necessary (the controller is the "active" partner). The non-controller will never initiate path switches (the non-controller is the "passive" partner). It will passively wait for the controller partner to do the switch.

Whether or not an RTP end point is a controller is determined by customer sysdef.

9.6.1.1 Examples of path switch controllers

A good example of a path switch controller is a mobile (e.g. wireless) node. In the case of mobile node talking to a (static) host, the RTP end point in the mobile node could be a controller and initiate the path switches (i.e. become the "active" path switch partner) because only it knows when it is in the process of moving from one base station to another (which could take some time). The RTP end point in the host could be the non-controller (and become the "passive" path switch partner) rather than waste a lot of time and energy trying to do the switch while the mobile node is in the process of moving. If both RTP end points reside in mobile nodes then both could be controllers and become "active" partners.

9.6.1.2 How controller capability is communicated to the RTP partners

Controller capability is communicated via the following. ("Origin" indicates the node that sends the RTP Connection Setup message. "Destination" indicates the node that receives the RTP Connection Setup message).

- Route Setup reply

  The destination communicates it's controller capability to the origin by setting a controller indicator in the Route Setup reply.

- RTP Connection Setup (HPR Routing Information Control Vector)

  The origin communicates it's controller capability to the destination by setting a controller indicator in the HPR Routing Information CV (which is contained within the RTP Routing Information segment) that accompanies the Connection Setup message.

Once the origin and destination know each other's controller capability, it can be determined who will do the path switch (i.e. who is "active" and who is "passive").

9.6.2 What triggers a path switch

A path switch is triggered by any of the following.

1. RTP "connection failure detection".

   A message is sent asking for status (status requested) and the short request timer (SRT) expires before receiving the status reply. The sender then attempts to determine the status of the partner by doing a status exchange. A status exchange message is sent to the partner asking for status and also including the status of the sender. Again the short request timer is used to wait for the status reply from the partner. If the timer expires, the status exchange message is retried until the retry limit is reached. At this point the sender concludes that connection has failed and triggers a path switch.

2. Local link failure

   If a local link being used by an RTP connection fails, this triggers a path switch. This trigger can cause CP-CP RTP connections to be switched much faster than using the RTP connection failure detection trigger.

3. Remote link failure (optional)

   A TDU is received indicating that a remote link has failed and RTP connections in this node are using that link. This trigger is used only by NNs (since ENs don't get TDUs).

4. Operator request (optional)

   The node operator or network management operator requests that the path be switched. A specific path may or may not be specified by the operator. This function is especially useful for switching an RTP connection back to it's original path after having been switched to an alternate (and possibly inferior) path as a result of a failure on the original path. Because of the importance of the switch-back function architecture strongly recommends that products implement some form of this function.

Products are required to implement the triggers 1 and 2 above but the others are optional. Trigger 1 is required because it handles the cases where remote links are links to ENs or are dynamic link connections across connection networks. Trigger 2 is required because it causes CP-CP session path switches to occur quickly (rather than wait for the RTP retry/liveness protocols to detect the failure). If CP-CP connections are not switched quickly, directory searches may be significantly delayed. If a directory search is done as a result of a path switch for an RTP connection carrying LU-LU sessions, a long delay (waiting for the CP-CP session path switch) may cause that RTP connection to fail because it's path switch timer expires.

9.6.3 Path switch timer

Once it is determined that a path switch needs to be done, a path switch timer (PST) is started. This timer indicates the time allowed to accomplish the switch. If this timer expires and the path has not been successfully switched, the RTP connection is failed. The timer is stopped (reset) when a new path is successfully obtained.

The path switch timer may be associated with either COS or transmission priority (TPF). Architecture strongly recommends that it be associated with COS since this provides maximum granularity and flexibility for the customer. The default value for this timer should be "large" (i.e. 1-3 minutes) to handle the majority of network configurations. However, the customer must be able to override this default value in order to tailor it to his specific environment. Some applications (for example, interactive real-time) may require that a path switch be done in a short period of time (say under 30 seconds) whereas others (for example, a midnight batch job) might have a much longer limit.

Note that the use of a path switch timer handles the case where a path switch is attempted before the TDU indicating the link failure has arrived. In this case, the same (bad) route may be calculated again and the RTP retries may fail again. This procedure could be repeated several times before the TDU arrives and a new (good) route is calculated.

Path switch may be disabled (not used) by setting the path switch timer to a value of zero. In this case, RTP will not attempt to do a path switch.

9.6.4 Path Switch FSM

FSM_PS (Figure 9-4 on page 9-13) is a finite state machine describing the general operation for the HPR non-disruptive path switch function. There is one instance of this machine for each RTP connection. The sections following the FSM explain the notation, inputs, states, etc. The FSM only shows the path switch being triggered by RTP connection failure detection (trigger 1), but the generated path switch line flows are the same regardless of what triggers them.

FSM_PS	Reset (PST not running)		Pending Path Switch (PST running)		
	not ERP (not echo pend, SRT may be running, RC=0) 1	ERP (echo pending, SRT running, RC¬=0) 2	Pending Get Path (¬passive, echo pending, SRT not running, RC=0) 3	Pending RIs (¬passive, echo pending, SRT running, RC¬=0) 4	Waiting for partner (passive, echo pending, SRT not running, RC=0) 5
SRT_expires, ¬Limit_reached	2(S_SR_ST(S+), Start_SRT_RC+)	-(S_SR_ST(S=), Start_SRT_RC+)	/	-(S_SR_ST(S=)_RIs, Start_SRT_RC+	/
SRT_expires, Limit_reached	/	passive: 5(Start_PST, RC=0) ¬passive: 3(Start_PST, RC=0, Get_path)	/	3(RC=0, Get_path)	/
Get_path_successful	-(Discard)	-(Discard)	4(old path=new path, RIs=new path, S_SR_ST(S+)_RIs, Start_SRT_RC+)	-(Discard)	/
Get_path_unsuccess- ful_retry	-(Discard)	-(Discard)	Delay: 4(RIs=old path, S_SR_ST(S+)_RIs, Start_SRT_RC+) ¬Delay: -(get path) (see note 1)	-(Discard)	/
Get_path_unsuccess- ful_no_retry	-(Discard)	-(Discard)	1(Fail)	-(Discard)	/
Rcv,¬RIr	-	E=S :1(RC=0, Stop_SRT) else:-	E=S :1(Use_old_path, S_ST(S=), Stop_PST) else:-	E=S :1(Use_RIs, RC=0, Stop_SRT, Stop_PST) else:-	E=S :1(Use_old_path, S_ST(S=), Stop_PST) else:-
Rcv, RIr, old	-(Discard)	-(Discard)	-(Discard)	-(Discard)	-(Discard)
Rcv, RIr, ¬old	-(Use_RIr, S_ST(S=))	E=S :1(Use_RIr, S_ST(S=), RC=0, Stop_SRT) E¬=S:-(Use_RIr, S_SR_ST(S+), RC=0, Stop_SRT, Start_SRT_RC+) (see note 3)	E=S :1(Use_RIr, S_ST(S=), Stop_PST) E¬=S:2(Use_RIr, S_SR_ST(S+), Start_SRT_RC+, Stop_PST) (see note 3)	¬win,E=S:1(Use_RIr, S_ST(S+), RC=0, Stop_SRT, Stop_PST) (see note 2) ¬win,E¬=S:2(Use_RIr, S_SR_ST(S+), RC=0, Stop_SRT, Start_SRT_RC+, Stop_PST) (see note 3) win: -(Discard)	E=S :1(Use_RIr, S_ST(S=), Stop_PST) E¬=S:2(Use_RIr, S_SR_ST(S+), Start_SRT_RC+, Stop_PST) (see note 3)
PST_expires	/	/	1(Fail)	1(Fail)	1(Fail)

Output Codes	Output Actions
Discard	Discard the received message. No RTP processing of any kind is done on the message (e.g. Status is ignored, data is ignored). It's just as though the message has been lost in the network.
Fail	Fail the RTP connection.
Normal	Perform normal RTP processing (i.e. receive the data, respond to SR, etc.).
S_...	Send a message with some or all of the following: SR - status is requested (SR bit set on in the THDR) ST - Status segment with: S+ - the SYNC value is 1 greater than the last sent SYNC value S= - the SYNC value is the same as the last sent SYNC value RIs - Routing Information segment for path chosen by this node (sender)
RC=0	Set the Retry Count (RC) to 0.
Start_SRT_RC+	Start the Short Request timer and increment the Retry Count (RC) by 1.
Stop_SRT	Stop (cancel) the Short Request timer.
Start_PST	Start the Path Switch timer.
Stop_PST	Stop (cancel) the Path Switch timer.
RIs=new path	Set the send Routing Information to the new path information obtained on the Get_path_successful input.
RIs=old path	Set the send Routing Information to the old path information.
old path=new path	Replace the old path information with the new path information.
Get_path	Request a new path. This may include directory search, RSCV calculation, and Route Setup protocol. See section "Obtaining a New Path".
Use_RIr	Start using path RIr.
Use_RIs	Start using path RIs.
Use_old_path	Use the old path.

Figure 9-4 (Part 2 of 2). FSM for path switch

9.6.5 Abbreviations

- **SRT** - Short Request Timer
- **PST** - Path Switch Timer
- **RC** - Retry Count
- **ERP** - Error Recovery Procedure
- **RIr** - Routing Information received from the partner
- **RIs** - Routing Information sent by this node
- **SR** - The Status Requested indicator in the THDR.

9.6.6 FSM_PS Input Definitions

- **SRT_expires** - The Short Request Timer (SRT) has expired.
- **PST_expires** - The Path Switch Timer (PST) has expired. The maximum time allowed to perform the path switch function has elapsed.
- **Limit_reached** - The Retry Count (RC) is equal to the RTP retry limit (i.e. the alloted number of retries have been done). The retry limit must always be greater than 0.
- **Get_path_successful** - A new path has been obtained. The routing information for the new path is provided.

- **Get_path_unsuccessful_retry** - A new path was not obtained but it is possible that a new route will be obtainable later. See 9.6.10, "Obtaining a New Path" on page 9-18 for details as to when this condition occurs.
- **Get_path_unsuccessful_no_retry** - A new path cannot be obtained. See 9.6.10, "Obtaining a New Path" on page 9-18 for details as to when this condition occurs.
- **Rcv** - A message has been received on the RTP connection. The message may contain data, Status segment, etc.
- **RIr** - The received message contains the Routing Information and Status segments and the Status Requested (SR) indicator is on. If a Routing Information segment is received without an accompanying Status segment or SR indicator on then it is a protocol error and the RTP connection is failed.
- **old** - The received message contains a Status segment with a SYNC that is older than the current SYNC. At the receiver, CUR_SYNC is the most recently received (newest) SYNC from the partner. RCVD_SYNC is the SYNC in the status message just received. CUR_SYNC is initialized to 0 when the connection is established. The following pseudo code shows how to determine whether or not a received message containing a SYNC is old or not.

```
If (RCVD_SYNC ≥ CUR_SYNC  AND (RCVD_SYNC - CUR_SYNC) < 2**15)  OR
   (RCVD_SYNC ≤ CUR_SYNC  AND (CUR_SYNC - RCVD_SYNC) > 2**15) then
    (The received message contains new SYNC or a SYNC equal to the
     previous one.  In either case, it is not considered old.)
    CUR_SYNC = RCVD_SYNC (update the current SYNC)
Else (received message is OLD)
   Don't update the current SYNC.
```

- **General processing of received message** - In general, all received messages are fully processed by RTP. Data is received, Status information is checked and recorded, Status Requested is acknowledged, etc. An exception is when the received message is explicitly discarded by FSM_PS, in which case, no RTP processing is done.

9.6.7 State Definitions

- **Reset** - Not in the process of doing a path switch.
- **Pending Path Switch** - In the process of doing a path switch.
- **ERP** - In the process of doing error recovery for a sent message.
- **echo pending** - Have sent a message that contains a STATUS segment and also requests status (SR indicator set on) from the partner. The value of the SYNC field in the sent Status segment is saved (last outstanding sent SYNC) and compared against a received ECHO to determine whether the sent SYNC message is being acknowledged by the partner.
- **SRT running** - The Short Request Timer is running.
- **RC = 0** - The Retry Count is 0.
- **RC ¬ = 0** - The Retry Count is greater than 0.
- **Pending Get Path** - In the process of obtaining a new path (i.e. RSCV and associated ANR routing information). Getting a new path may involve a direc-

tory search, RSCV calculation, and Route Setup protocol. This state is not used by a passive partner. A passive partner is one who will not initiate a path switch but, instead, wait for the active partner to do it.

- **Pending RIs** - Have sent Routing Information to the partner and are waiting for an acknowlegment. This state is not used by a passive partner.
- **Waiting for partner** - Waiting for the partner to do the path switch (i.e. waiting to receive RIr). This state is only used by a passive partner.

9.6.8 FSM_PS Predicate Definitions

FSM predicate definitions appear with the action code and used to indicate further qualifications (more input conditions) within the state.

- **Delay** - The product has chosen the option to delay before reattempting to obtain a new path.
- **passive** - This node is acting as a passive partner for path switch (i.e. is depending upon the partner to do the switch).
- **E=S** - The received message contains a Status segment and the value of the ECHO field is equal to the last outstanding sent SYNC value.
- **E¬=S** - The received message contains a Status segment and the value of the ECHO field is not equal to the last outstanding sent SYNC value.
- **win** - This node is the winner (i.e. the one that activated this RTP connection by sending the Connection Setup). In the case of a path switch race, the winner's path is used.
- **else** - Indicates the action to be taken when none of the other predicates are true.

9.6.9 Notes

9.6.9.1.1 Note 1: When a Get_path_unsuccessful_retry occurs it means another attempt should be made to get a new path. There are two options (product implementation choices):

1. Delay

   Delay the attempt to obtain a new path for the amount of time it takes to retry (i.e. do full RTP error recovery) the old path. Delaying allows time for any outstanding topology updates to arrive, links to be fully activated, etc. Delaying also minimizes the number of directory searches and Route Setups flowing through the network.

2. No delay

   Some products may elect to try to get a path (again) immediately and not to delay at all. This has the advantage of possibly getting a new path sooner thus decreasing the time it takes to do a path switch.

9.6.9.1.2 Note 2: The reason the SYNC is incremented in this case is illustrated in Figure 9-5 on page 9-17. If the SYNC number in the RIa "acknowledgment" (flow 5) was left at 4 (and not incremented to 5 as shown) then when RTP A received the

RIb message (at 6) it would not be "old" and would result in RTP A using RIb. In the meantime RTP B would be using RIa and so the route would be asymmetric.

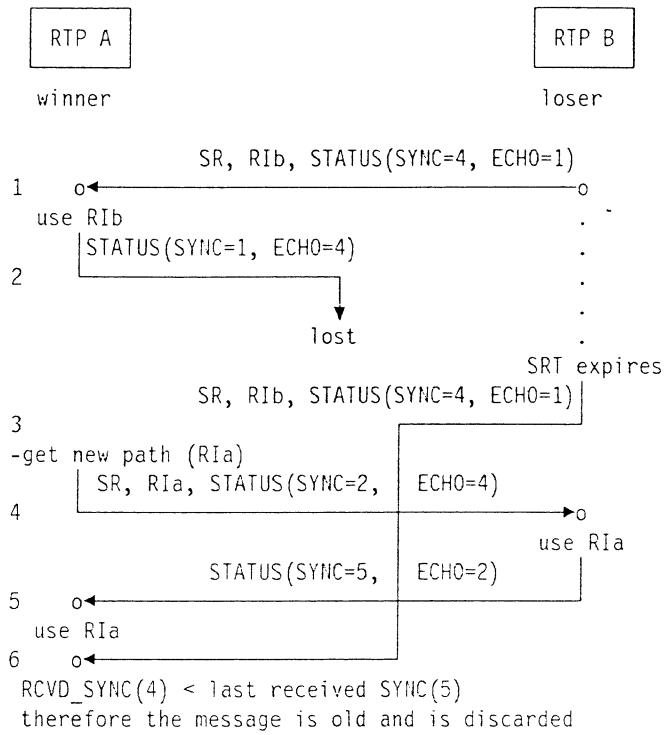


Figure 9-5. Reason for incrementing SYNC when acknowledging RI during race

9.6.9.1.3 Note 3: A new route (RIr) has been received from the partner but an ECHO has not yet been received to the outstanding SYNC. In these cases, we will switch over to the new route and restart the error recovery process (i.e. the retry count is reset, status is requested on the message sent to acknowledge receipt of the new route, a new SYNC number is used, and the path switch timer is canceled).

The reason that the SYNC number is incremented before sending it on the Status "acknowledgment" (S_SR_ST(S+)) is so that the partner can determine whether a received RI message is old. Figure 9-6 on page 9-18 shows the case where a RI message with a SYNC the same as a previous one must not be considered as old. In this figure the "acknowledgment" (flow 2) to the received RI message is lost so RTP B retries the message. RTP A receives the RI message with SYNC=4 (flow 3) a second time and must again process and acknowledge it. It must not consider it old and discard it.

Figure 9-7 on page 9-18 shows the case where the SYNC number must be incremented on the "acknowledgment" so that truly old RI messages can be recognized. If the SYNC on the "acknowledgment" (flow 3) was not incremented, the RIb message received by RTP A would be considered acceptable (as in Figure 9-6 on page 9-18).

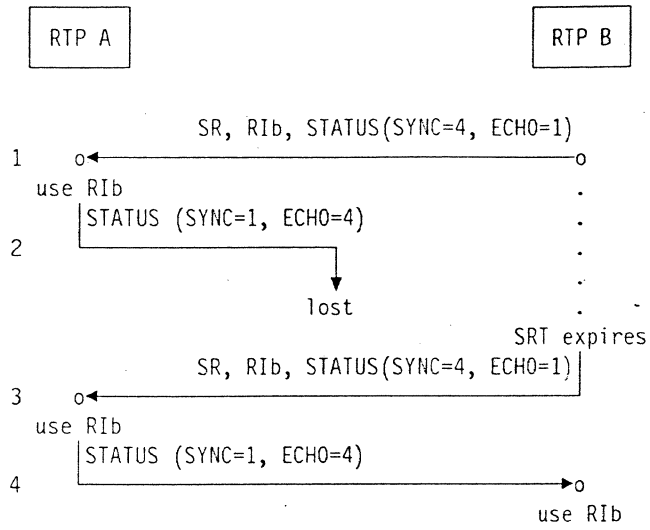


Figure 9-6. Legitimate case for receiving RI messages with same SYNC

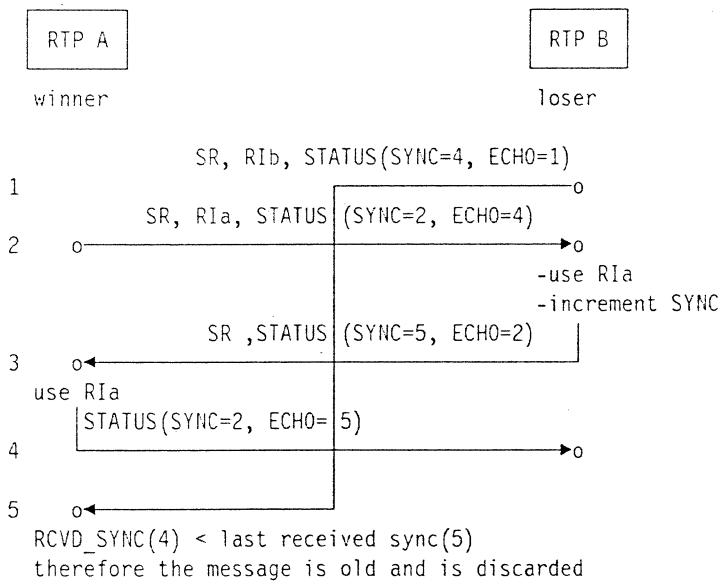


Figure 9-7. Why SYNC is incremented on acknowledgment during race

9.6.10 Obtaining a New Path

An attempt is made to obtain a new path when performing a path switch for an RTP connection. The new path connects the RTP connection endpoints and all links along the path are HPR capable (i.e. can perform ANR routing). The following modifications have been made to obtain an HPR only path.

- A new indicator has been added to the LOCATE/CDINIT so an EN can request an HPR only route from its NN server. Note that this indicator will only be recognized by HPR-level NN servers. Note that an HPR-level NN

server is one that contains any level of HPR support (e.g. it doesn't have to support the Transport tower). In other words, all NN servers that support the HPR base support understand the new "HPR only route" indicator on LOCATE/CDINIT and how to calculate HPR only routes.

If an EN request an HPR only route and the NN server cannot calculate one (because an HPR only path does not exist to the destination) then the LOCATE/CDINIT is rejected by the NN server with sense code X'80140006'.

- A modification has been made to the Route Calculation algorithm to calculate HPR only paths - see 7.8.1, "HPR-Only Route For Path Switch" on page 7-48.

New paths are represented by RSCVs (just as in APPN). Obtaining a new path may involve some or all of the following.

- Directory search

  The target resource used for directory searches is always the CP name of the node in which the remote RTP connection partner resides. The resource type is LU.

- RSCV calculation (NNs only)

- Route Setup protocol

  When a new route is obtained as a result of a path switch, the following conditions are checked to determine whether the Route Setup protocol needs to be performed.

  1. an existing RTP connection is already using the new route

     All RTP connections may be searched to see if any are already using the new route. The RTP connection that is doing this path switch is also searched because the new route could be the same as the old one.

  2. the new path does not require a Route Setup

     Each RTP connection saves information about the route it is currently using. This route information includes everything obtained from the Route Setup protocol. Any route that passes through a subnet (e.g. CNN) has the potential of failing within the subnet (i.e. a CNN subarea link fails) and therefore requires that a Route Setup protocol be done on path switch so as to allow the subnet to attempt to find an alternate path around the failure.

     The Route Setup request contains a field that is set to indicate whether or not a Route Setup is required on path switch (see A.1.15, "Route Setup GDS variable X'12CE'" on page A-31).

  3. the route is in a usable state

     This test only applies to RTP connections other then the one doing this path switch.

     The state of this RTP connection indicates whether the route is currently usable (i.e. it is not in the process of doing a path switch).

  If all of the above conditions are true then the route information saved by the existing RTP connection may be used for the new route. (i.e. the Route Setup

protocol need not be used to obtain it). Otherwise, the Route Setup protocol must be used to obtain the route information.

Exactly which of the above functions are performed depends on the type of node in which the RTP endpoints reside. For this discussion, the node initiating the path switch is called the initiator node and the node containing the partner RTP endpoint is called the partner node. The logic for the various cases is described using psuedo code. Also shown are the settings of the "return" values (Get_path_successful, Get_path_unsuccessful_retry, and Get_path_unsuccessful_no_retry) used in the FSM. "Do Route Setup protocol" means do it if necessary (as described above). If it's not necessary to do the protocol (because the routing information is already available) then the protocol is consisdered to have been successful.

- The initiator node is a NN and the partner node is a NN

  In this case the RSCV is calculated by the initiator since it knows the location of all NNs in the network.

```
Calculate an RSCV to the partner
If successful then
   Do Route Setup protocol
   If successful then
      return "Get_path_successful"
   Else
      return "Get_path_unsuccessful_retry"
Else
   return "Get_path_unsuccessful_retry"
```

- The initiator node is a NN and the partner node is an EN

  The NN sends a directory search (broadcast or directed) to get the latest TG vector information about the partner EN. After receiving the search reply, the NN calculates a new RSCV.

```
Do directory search for the EN
If successful then
   Calculate an RSCV to the partner
   If successful then
      Do Route Setup protocol
      If successful then
         return "Get_path_successful"
      Else
         return "Get_path_unsuccessful_retry"
   Else
      return "Get_path_unsuccessful_retry"
Else
   If the partner node is mobile then
      return "Get_path_unsuccessful_retry" (the partner may be in the
       process of moving)
   Else
      return "Get_path_unsuccessful_no_retry"
```

- The initiator node is an EN and the partner node is either an EN or NN

  The initiator sends a directory search to its network node server requesting an HPR only route. An HPR only route is requested via a new indicator in the

CD-Initiate GDS variable. If the partner node is an EN, the NN server will send either a broadcast or directed search to verify the EN location and obtain it's TG vectors. If the partner node is a NN, the NN server either finds the resource locally (by searching its topology database) or does a broadcast or directed search. After receiving the search reply, the NN server calculates a new RSCV and returns it to the EN.

```
Do directory search (i.e. send search request to NN server)
If successful then (RSCV returned when successful)
   Do Route Setup protocol
   If successful then
      return "Get_path_successful"
   Else
      return "Get_path_unsuccessful_retry"
Else
   If the partner is a path switch controller (e.g. mobile node) then
      return "Get_path_unsuccessful_retry" (the partner may be in the
       process of moving)
   Else
      return "Get_path_unsuccessful_no_retry"
```

9.6.10.1 Limited Resource Link Considerations when Obtaining New Path

The route calculation algorithm does not take into consideration limited resource links since that link property is not part of topology information. Therefore, it's possible to obtain a new path that contains a limited resource link when the original path did not contain one. In this case it's impossible to communicate that information to APPN session end points (because it is communicated on the BIND request and response and there is no new BIND on path switch). The result may be the limited resource link on the new path is never deactivated because the session end points don't deactivate the sessions (because they were never told that a limited resource link exists along the path). Note that if the LU session end points reside in the same node as the RTP connection end point, the LUs may be notified of limited resource changes on new paths.

One possible solution to this problem is to not do the path switch in this case. However, it's very likely that customers will use switched links (limitied resource) as backup links just so they can be used for path switch when the primary link fails. So not doing a path switch is not acceptable.

Another possible solution is to allow the customer (operator) to maunually switch back to the original (primary) path when it comes back up. After the switch back occurs, the limited resource link will be deactivated normally. Manual switch back is a desireable function aside from the limited resource problem and so this solution is strongly recommended by architecture.

9.6.10.2 HPR EN obtaining a new path from APPN NN server

An HPR EN is connected to an APPN NN (the server) and an HPR NN (not the server). The first time the EN gets a route from the APPN NN server it happens (by chance) to be an HPR only route (i.e. consists of all HPR links) through the HPR NN and on to the destination HPR node. Later, that route fails causing the HPR EN to attempt a path switch. To obtain a new path, it sends a LOCATE/CDINIT to the APPN NN server. This time it specifically asks for an HPR only route. However, the APPN NN does not understand the HPR only

indicator (new for HPR) in the LOCATE/CDINIT. so it may or may not get an HPR only route (pure chance). So, when the route (RSCV) is returned to the HPR EN, it must be checked to make sure it consists of all HPR links. If it doesn't consist of all HPR links the RTP connection must be failed.

9.6.11 Sample Path Switch Sequences

The following are sequences illustrating various (and sometimes wierd) race/timing conditions involving path switch. Although the probability of some of these occuring is very low they are all handled by the HPR path switch protocol.

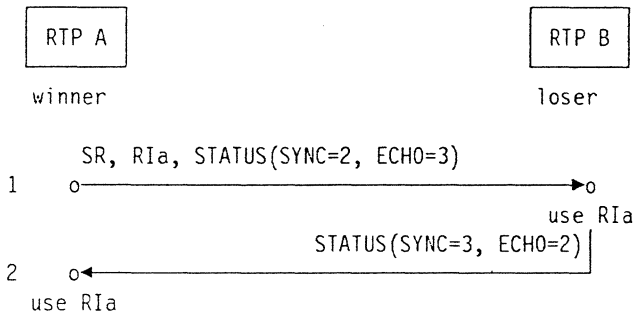


Figure 9-8. Winner initiates path switch - no race

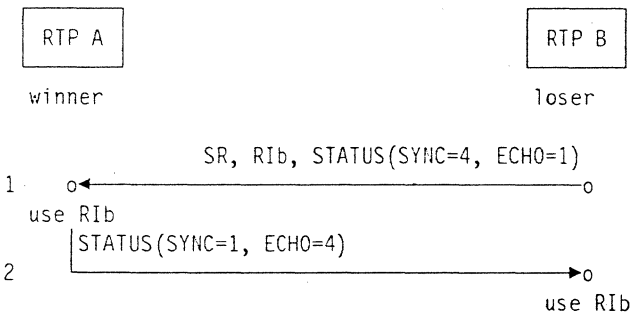


Figure 9-9. Loser initiates path switch - no race

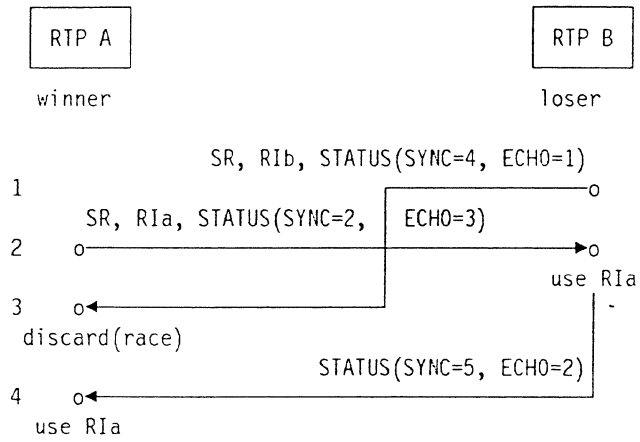


Figure 9-10. Loser's path switch message discarded because of race

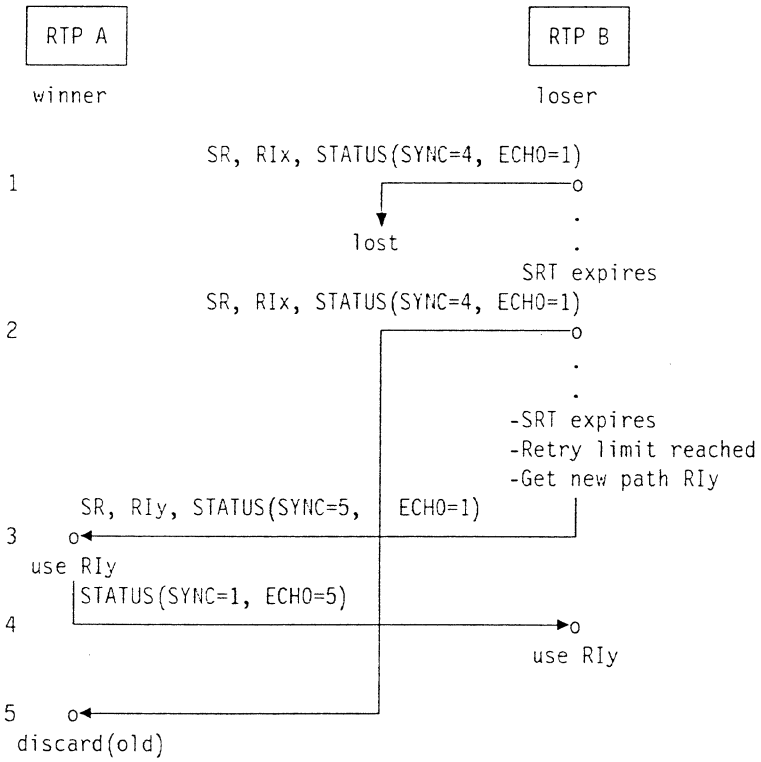


Figure 9-11. Path switch messages received out of order - old one discarded

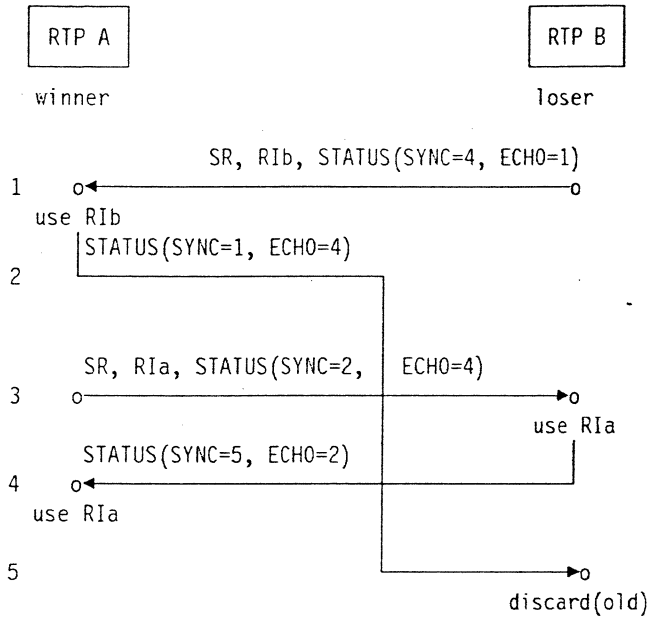


Figure 9-12. Old "acknowledgment" discarded at loser

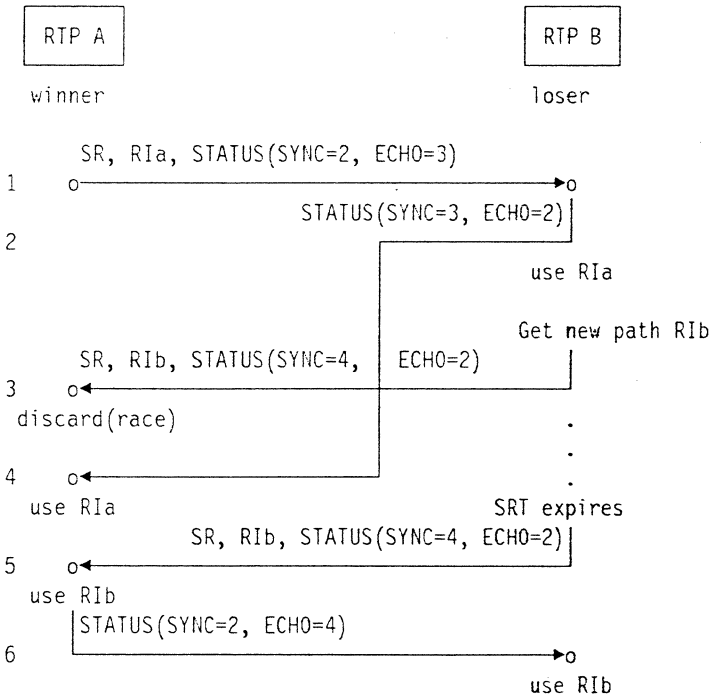


Figure 9-13. Race case where loser's route ends up being used

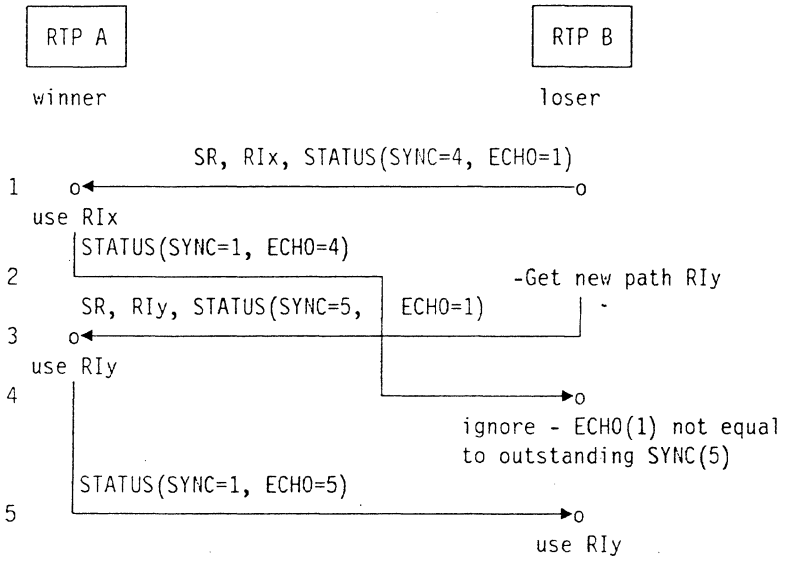


Figure 9-14. Ignoring old "acknowledgment"

Chapter 10. LU-LU Sessions involving APPN nodes (APPN/HPR Boundary Function)

10.1 Introduction

An **APPN/HPR boundary function (BF)** is performed in an HPR node when traffic passing through it needs to change protocols (i.e. go from APPN to HPR or HPR to APPN). See Figure 10-1.

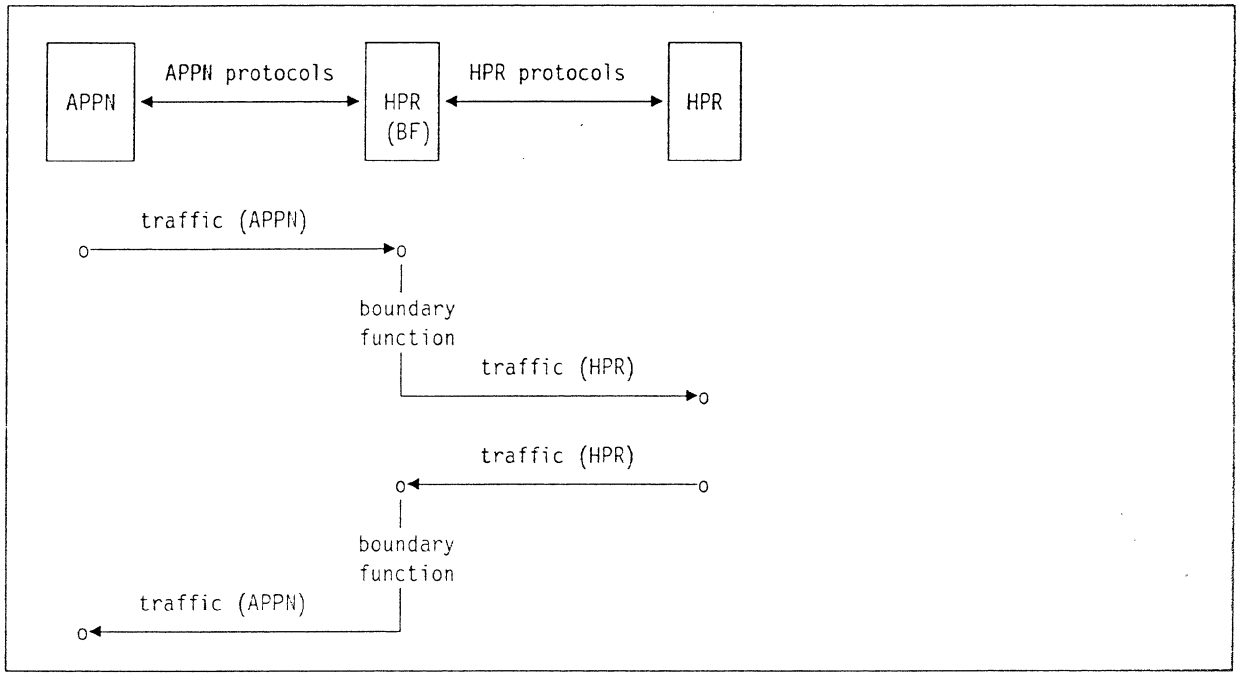


Figure 10-1. Boundary function (BF)

All protocols between the HPR BF and the APPN node are today's APPN protocols. The protocols between the HPR BF and the HPR node are HPR protocols as described in this document.

10.2 CP-CP Session Protocols

The CP-CP session protocols (directory and topology) are basically the same in HPR as in today's APPN. However, in HPR, the CP-CP PIUs are carried in Network Layer Packets (NLPs) which have headers for ANR routing (NHDR) and RTP connections (THDR). Also, the FID2 TH is replaced by a new FID5 TH to enhance session addressing (see 7.9, "Enhanced Session Addressing" on page 7-48).

For CP-CP session flows the APPN-HPR BF will convert between FID2 PIUs and NLPs. When a message is received on an APPN CP-CP session and needs to be forwarded over an HPR CP-CP session, the received FID2 PIU is converted to a

FID5 PIU and encapsulated in an NLP (i.e. the PIU is carried as data in the NLP). When a message is received on an HPR CP-CP session and needs to be forwarded over an APPN CP-CP session, the FID5 PIU in the received NLP is removed and converted to a FID2 PIU. This FID2/FID5 PIU conversion involves:

- TH

  All fields in the FID2 and FID5 TH are the same except for the session address fields. Therefore, the only conversion necessary is to change from the FID2 LFSID to the FID5 enhanced session address and vice-versa.

- RH and RU

  No conversion done - exactly the same in NLP or FID2 PIU.

10.3 LU-LU Sessions

When an HPR node receives a BIND from an APPN node that is destined to pass through an HPR subnet, the processing performed by the HPR node for the HPR subnet side is the same as the case where the BIND is originating from the HPR node. That is, a route setup is done, a new RTP connection is established, or an existing RTP connection is used. All subsequent session traffic (including UNBIND) flows over the RTP connection as normal RTP data.

10.3.1 Route Setup

Whenever an APPN node participates in an LU-LU session (either as the origin, destination, or intermediate node) that passes through an HPR subnet, an NCE must be obtained in order to properly route to the appropriate internal component at the last HPR node receiving the message. The Route Setup protocol is used to obtain this NCE. (See 9.2, "Route Setup protocol" on page 9-2 for a general description of this protocol). There are two cases.

- Case 1 - the final destination LU is within this HPR subnet

  In this case the destination LU resides in an HPR node (referred to as the destination HPR node). In order to properly route packets (using ANR routing) to this LU, the NCE of the LU must be obtained so it can be used as the final ANR label in the ANR routing field. This NCE is returned on the Route Setup reply. See 7.1.4.2, "LU NCEs" on page 7-2 for more information about LU NCEs.

- Case 2 - the final destination is beyond the HPR subnet

  In this case the destination LU resides somewhere beyond and outside of the HPR subnet. It may reside in an APPN node or an HPR node (in another disjoint HPR subnet). Either way, the route passes over an APPN link immediately after leaving this HPR subnet. (Remember, the definition of an HPR subnet is a group of contiguously connected HPR nodes over HPR links). This means the destination HPR node (the last HPR node to process the packet before it leaves the subnet) must execute the boundary function (i.e. transform the packet into APPN format) and forward the packet over the APPN link. In order to properly route packets (using ANR routing) to the destination HPR node, the NCE of the component that contains the BF processing (within the destination HPR node) must be obtained so it can be used as the final ANR

label in the ANR routing field. The Route Setup protocol is used to obtain this NCE. See 7.1.4.3, "Boundary Function NCE (NCE_BF)" on page 7-2 for more information about BF NCEs.

10.4 Sequences

The clearest way to describe the APPN/HPR BF is with sequences. There are 3 basic configurations.

1. APPN(EN or NN) -- HPR nodes(NNs) -- APPN(EN or NN)
2. APPN(EN or NN) -- HPR nodes(NNs) -- HPR(EN or NN)
3. HPR(EN or NN) -- HPR nodes(NNs) -- APPN(EN or NN)

The following sections describe these sequences.

10.4.1 APPN -- APPN

Figure 10-2 on page 10-4 shows an example of a session between two APPN nodes (a and e) that passes through an HPR subnet (nodes b, c, and d).

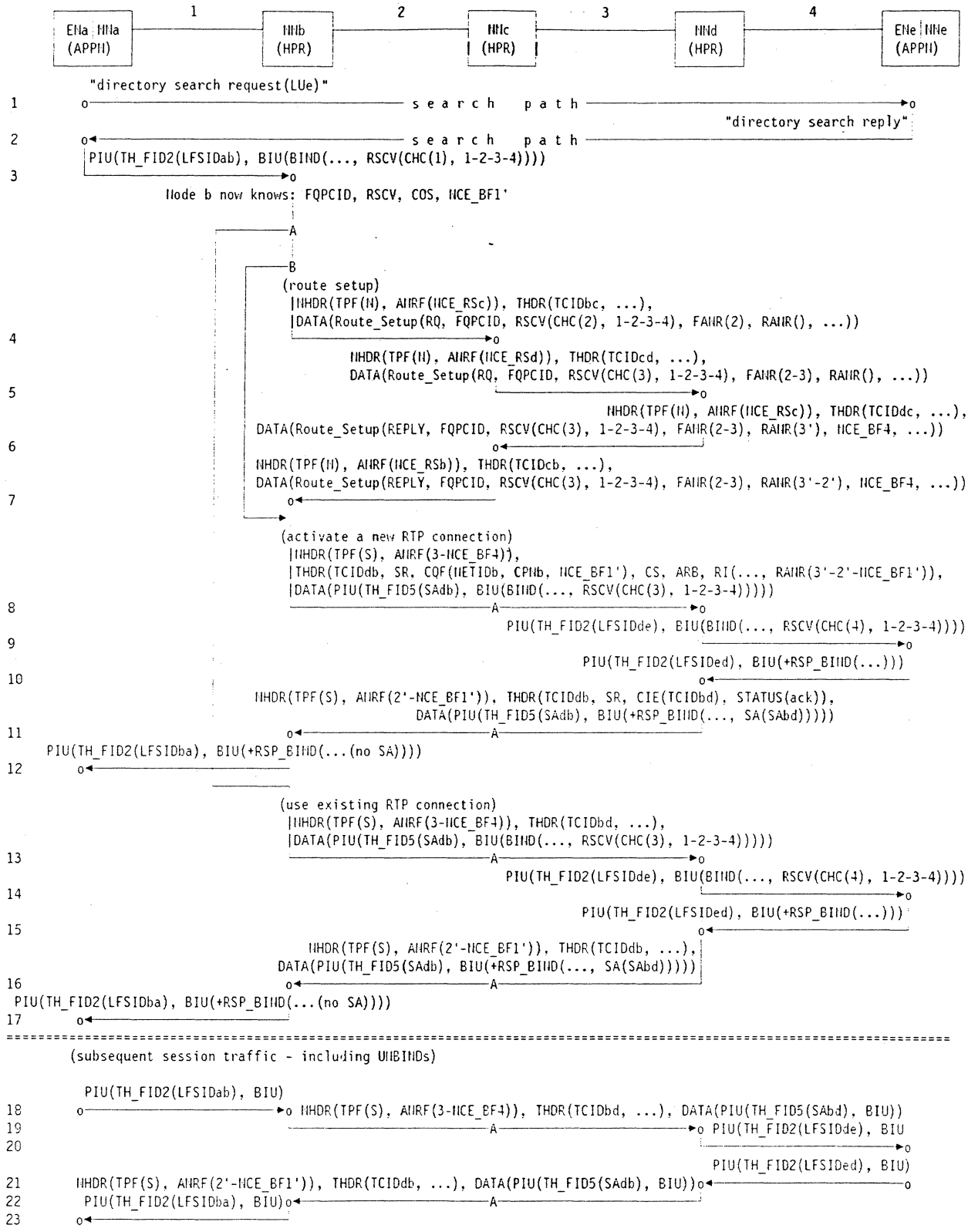


Figure 10-2. LU-LU Session : APPN -- APPN (through HPR subnet)

Conditions:

A If an RTP connection exists with matching RSCV(2-3-4), COS, and NCE_BF1' then use the existing RTP connection to transport this session (i.e. go to step 13). Notice that the first APPN TG (4) is compared along with the HPR links (2 and 3) in the RSCV compare. This is because there may be, at most, a single NCE associated with a BF for each APPN link. Therefore, if any RTP connection exists for that APPN link the BF NCE associated with the link is known.

B If an RTP connection exists with matching RSCV(2-3-4) then use routing information associated with the existing RTP connection to activate a new RTP connection (i.e. skip the route setup protocol).

Annotations:

1-2 A directory search is done exactly as in APPN.

3 Node a sends the BIND.

4 A Route_Setup request is sent to node c over the long-lived route setup RTP connection. Node b fills in the appropriate information for the forward ANR (2) and updates the RSCV hop count to 2.

5 Node c fills in the appropriate information for the forward ANR (3), updates the current hop count in the RSCV (to 3) and forwards the Route_Setup request on to node d.

6 Node d returns a Route_Setup reply. This reply now contains all the necessary information about the complete forward path. Node d adds information about the reverse ANR (4') to the reply to begin accumulating the reverse path information.

7 Node b now has all the information associated with the route between node b and node d.

8 Node b activates a new RTP connection to node d. The NHDR ANR field (ANRF) contains the forward route ending with the NCE associated with BF for APPN TG 4 (3-NCE_BF4). The THDR reverse ANR segment field (RANR) contains the reverse route and it ends with the NCE associated with BF for APPN TG 1' (3'-2'-NCE_BF1'). The BIND request is carried as data with the Connection Setup (CS) and the RSCV current hop count is updated appropriately.

9 The BIND is forwarded on as FID2.

10 The FID2 BIND response is received.

11 Node d responds with a Connection ID Exchange (CIE) to establish a TCID (TCIDbd) to be used for data sent from node b to node d over this RTP connection. In this particular example sequence, the CIE message also acknowledges that the Connection Setup was received successfully (i.e. the STATUS(ack) is piggy-backed with the + RSP_BIND). The BIND response is carried as data along with the Connection ID Exchange and it includes the HPR enhanced session address (SAbd) to be used on the session when sending data from node b to node d.

12 The BIND response is forwarded back to node a (as FID2). The enhanced session address SA(SAbd) is removed.

13 Node b sends the BIND on an existing RTP connection. The BIND is carried as data and the RSCV current hop count is updated appropriately.

14-15 The BIND and BIND response are exchanged as FID2 between node d and node e.

16 Node d assigns an enhanced session address (SAbd), adds it to the BIND response, and sends it back to node b on the existing RTP connection.

17 The BIND response is forwarded back to node a (as FID2). The enhanced session address SA(SAbd) is removed.

18-23 Session traffic flows over the established RTP connection and includes UNBINDs as well as session data. The appropriate TCIDs and SAs are used.

10.4.2 APPN -- HPR

Figure 10-3 on page 10-7 shows an example of a session between an APPN node (a) and an HPR node (e) that passes through a series of HPR NNs (nodes b, c, and d).

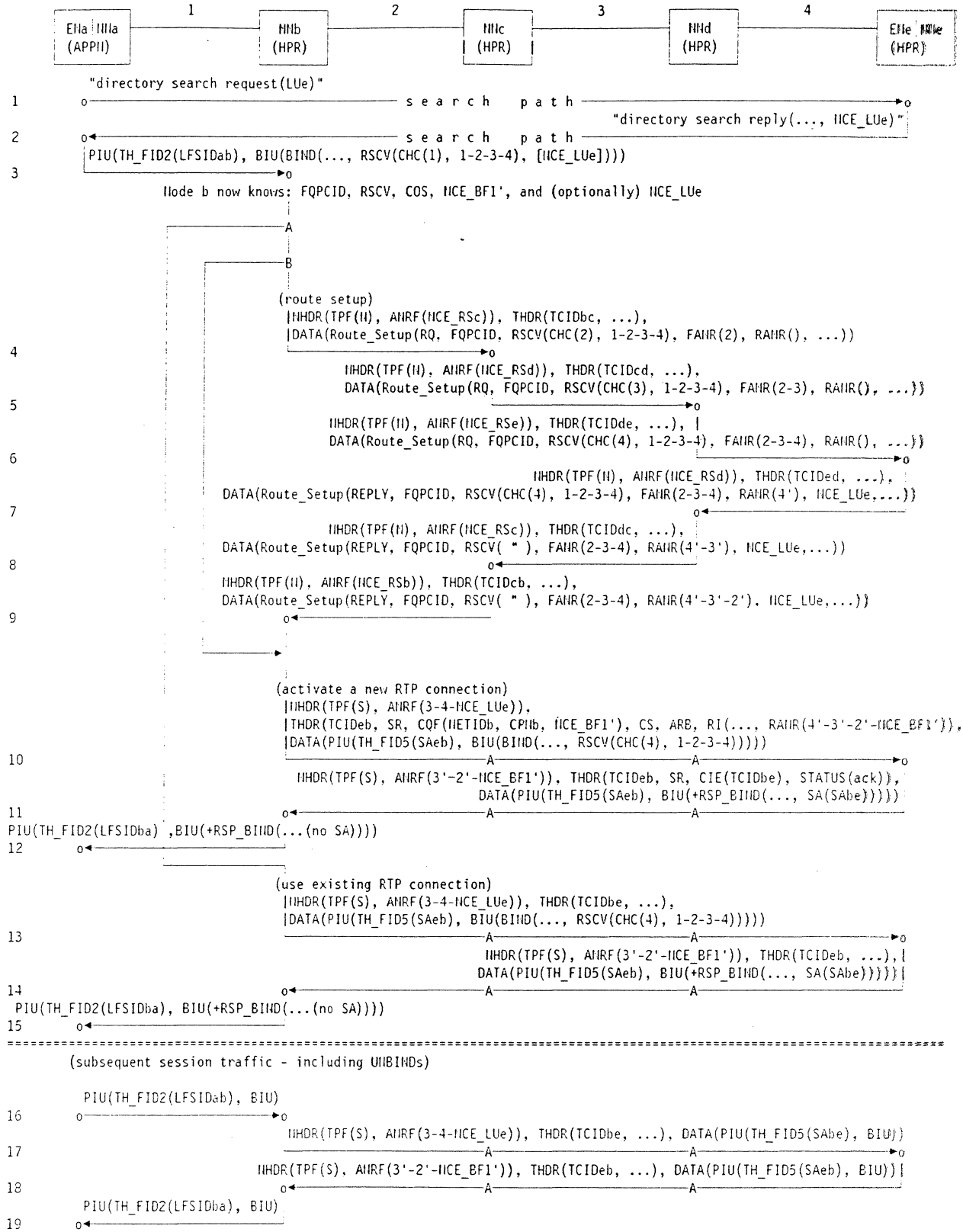


Figure 10-3. LU-LU Session : APPN -- HPR

Conditions:

A If the NCE_LUe is present on the BIND request and an RTP connection exists with matching RSCV(2-3-4), COS, NCE_BF1', and NCE_LUe then use that existing connection.

B If the NCE_LUe is present on the BIND request and an RTP connection exists with matching RSCV(2-3-4) then activate a new RTP connection (use routing information from the existing RTP connection and currently known COS, NCE_BF1', and NCE_LUe).

Annotations:

1-2 A directory search is done exactly as in APPN except that node e includes the NCE associated with the found LU (NCE_LUe).

3 Node a sends the BIND. If node a contains the logic to transfer the NCE from the directory search reply to the BIND then it does so. This "NCE transfer" update to old APPN nodes will improve session setup performance within HPR significantly because it eliminates having to do a route setup for every single session in order to obtain the NCE.

4-9 The route setup is done. Node e returns the NCE associated with LUe (NCE_LUe) in the Route Setup reply.

10-11 Node b activates a new RTP connection to node e using the forward route 2-3-4-NCE_LUe and the reverse route 4'-3'-2'-NCE_BF1'. The BIND request and response are carried as data along with the RTP connection setup.

12 The BIND response is forwarded back to node a (as FID2). The enhanced session address SA(SAbe) is removed.

13 Node b sends the BIND on an existing RTP connection. The BIND is carried as data and the RSCV current hop count is updated appropriately.

14 Node e assigns an enhanced session address (SAbe), adds it to the BIND response, and sends it back to node b on the existing RTP connection.

15 The BIND response is forwarded back to node a (as FID2). The enhanced session address SA(SAbe) is removed.

16-19 Session traffic flows over the established RTP connection and includes UNBINDs as well as session data. The appropriate TCIDs and SAs are used.

10.4.3 HPR -- APPN

Figure 10-4 on page 10-9 shows an example of a session between an HPR node (a) and an APPN node (e) that passes through a series of HPR NNs (nodes b, c, and d).

Figure 10-4. LU-LU Session : HPR -- APPN

Conditions:

A If an RTP connection exists with matching RSCV(1-2-3-4), COS, and NCE_LUa then use the existing RTP connection to transport this session (i.e. go to step 13). Notice that the first APPN TG (4) is compared along with the HPR links (2 and 3) in the RSCV compare. This is because there may be, at most, a single NCE associated with a BF for each APPN link. Therefore, if any RTP connection exists for that APPN link the BF NCE associated with the link is known.

B If an RTP connection exists with matching RSCV(1-2-3-4) then use routing information associated with the existing RTP connection to activate a new RTP connection (i.e. skip the route setup protocol).

Annotations:

1-2 A directory search is done exactly as in APPN.

3-8 The route setup protocol is done. Node d returns the NCE for the BF associated with APPN TG 4 (NCE_BF4).

9 Node a activates a new RTP connection to node d with forward route 1-2-3-NCE_BF4 and reverse route 3′-2′-1′-NCE_LUa.

10-11 The BIND request and response are exchanged (on FID2 PIUs) between nodes d and e.

12 Node d sends CIE back to node a.

13 Node a sends the BIND on an existing RTP connection. The BIND is carried as data and the RSCV current hop count is updated appropriately.

14-15 The BIND and BIND response are exchanged as FID2 between node d and node e.

16 Node d assigns an enhanced session address (SAad), adds it to the BIND response, and sends it back to node b on the existing RTP connection.

17-20 Session traffic flows over the established RTP connection and includes UNBINDs as well as session data. The appropriate TCIDs and SAs are used.

10.4.4 Segmenting/reassembling

The APPN intermediate reassembly function (as defined in today's APPN) is not done in the case where the outbound stage is over an RTP connection (however, it is still done when the outbound stage is over an APPN link). This is because segmenting is performed on each RTP connection rather than each session. See 7.2.4.2, "Segmentation and Reassembly" on page 7-17 for more information on RTP segmenting.

Appendix A. Formats

This appendix describes new and changed formats for HPR. Abbreviations and default values used in the sequences throughout this document are also described for each field under the "Sequence:" heading. A default value means that this is the value used unless a different value is explicitly specified in the sequence.

A.1 New for HPR

A.1.1 HPR Link Frame

Table A-1. HPR Link Frame

Byte	Bit	Content
0-q		DLC Header (format depends on DLC type)
q + 1-r		2.1 Packet Contains either an XID I-field or a Network Layer Packet (NLP).
r + 1-s		DLC Trailer (format depends on DLC type) The exact format depends on the DLC type but within every DLC Trailer there is a CRC that covers the DLC Header and Packet fields (bytes 0-r) This CRC is the ONLY data integrity check used by HPR therefore it is required for every link (i.e. exactly as in today's APPN).

A.1.2 2.1 Packet

Table A-2. 2.1. Packet

Byte	Bit	Content
0	0-3	Packet Type - indicates the format of the packet. 0010 FID2 PIU (TH(FID2)-RH-RU) used for APPN traffic 110x NLP (Network Layer Packet) used for HPR traffic
0	4 to byte n	Rest of FID2 PIU or NLP packet

A.1.3 Network Layer Packet (NLP)

Table A-3. Network Layer Packet (NLP)

Byte	Bit	Content
0-k		Network Layer Header (NHDR)
k+1-m		RTP Transport Header (THDR)
m+1-n		Data (DATA)

Note: Sequences: The NHDR, THDR, and DATA are always shown in the sequences.

A.1.4 Network Layer Header (NHDR)

Table A-4. Network Layer Header (NHDR)

Byte	Bit	Content
0	0-2	Routing Mode (RM): 110 ANR (only value currently being used by HPR) 001 permanently reserved (because this value is used for distinguishing FID2 PIU packets from NLP packets)
	3-4	reserved
	5-6	Transmission Priority Field (TPF) 00 Low (L) 01 Medium (M) 10 High (H) 11 Network (N) **Sequences**: This field is always specified. A value of "S" means session priority associated with RTP connection (i.e. the same as is specified in the BINDs for sessions being carried by this RTP connection).
	7	reserved
1		reserved
2-m		ANR Routing field (ANRF): AL1-AL2-...-ALn-X'FF' AL1, AL2, etc. are ANR labels associated with each TG (link) in the route (path). Each TG has two ANR labels (one for each direction). A string of these labels (AL1-AL2-...-ALn) represent a path through the network. Note that there are no delimiters separating the labels. *High-order bit*: The high-order bit of each label is reserved and always set to B'1'. **Sequences**: This field is always specified in the form ANRF(AL1-AL2-...-ALn). The X'FF' delimiter value following ALn is never explicitly shown.
m + 1		reserved

A.1.5 RTP Transport Header (THDR)

The length of the THDR is always an integer multiple of 4 (i.e., 4, 8, 12, etc.), to insure alignment of any subsequent GDS variables on a 4-byte boundary. Variable length data is placed in control vectors or RTP optional segments. RTP optional segments may include control vectors.

The length of an RTP optional segment is also an integer multiple of 4. Any padding required is included in the optional segment.

For control vectors, the length field counts the actual number of bytes in the control vector, including the header (length and key) and the data fields. This number is not necessarily an integer multiple of 4.

The first control vector embedded in the THDR or optional segment must begin on an alignment point. To ensure this, all preceeding fields in the enclosing structure must be fixed length with reserved fields if necessary.

Subsequent control vectors or optional segments also begin on alignment points. To achieve this, up to 3 bytes of non-significant padding (X'00') may be present after the control vector.

Table A-5 (Page 1 of 4). RTP Transport Header (THDR)

Byte	Bit	Content
0-3		Transport Connection Identifier (TCID)
	0	When the high-order bit is 0 this is the connection identifier chosen by the receiver, who can thus identify the connection without referring to the Connection Qualifier field. When the high-order bit is 1 this is a connection identifier that is further qualified by the Connection Qualifier field.
	1-31	A 31 bit field that along with the Connection Qualifier / Source Identifier field uniquely identifies an RTP connection. Sequences: This field is always specified in the format TCIDxyn where x and y are node identifiers and n is an optionally specified instance number. TCIDab means this is the TCID that is used when sending data from node a to node b. Multiple TCIDs used between the same two nodes are distinguished by the instance number (e.g. TCIDab1, TCIDab2).
4	0	reserved
	1	Connection Setup Indicator (SETUPI): 0 connection setup segment not present (¬SETUP) 1 connection setup segment present (SETUP) Sequences: If the Connection Setup segment is present, the value of this field defaults to SETUP; otherwise, it defaults to ¬SETUP. The CSI field is never explicitly shown in the sequences.
	2	Start of Message Indicator (SOMI) - used by RTP for segmenting/reassembling: 0 not start of message (¬SOM) 1 message starts with first byte of user's data (SOM) Sequences: The default value is SOM.
	3	End of Message Indicator (EOMI) - used by RTP for segmenting/reassembling: 0 not end of message (¬EOM) 1 message ends with last byte of user's data (EOM)

Table A-5 (Page 2 of 4). RTP Transport Header (THDR)

Byte	Bit	Content
		Sequences: The default value is EOM.
	4	Status Requested Indicator (SRI): 0 receiver need not reply with a status segment (¬SR) 1 receiver must reply with (at least) a status segment (SR) Sequences: This field is always explicitly shown as SR, ¬SR, or *SR. *SR means the field is set appropriately according to the protocol for obtaining status.
	5	Respond ASAP Indicator (RASAPI): 0 receiver need not transmit reply ASAP (¬RASAP) 1 receiver must transmit reply ASAP (RASAP) Sequences: The default value is ¬RASAP.
	6	Retry Indicator (RETRYI): 0 sender will retransmit this packet (RETRY) 1 sender will not retransmit this packet (¬RETRY). When ¬RETRY is specified in a THDR that also contains the Connection Setup segment then data will sent unreliable (i.e. no error recovery will be be performed) on the connection. Sequences: The default value is RETRY.
	7	reserved
5	0	Last Message Indicator (LMI): 0 not sender's last message on this connection (¬LM) 1 sender's last message on this connection (LM) Sequences: The default value is ¬LM.
	1-2	reserved
	3-4	Connection Qualifier Field Indicator (CQFI): 00 none present (NOCQF), 01 originator (ORIGIN) all other values reserved Sequences: If the CQF segment is present, the value of this field defaults to ORIGIN; otherwise, it defaults to NOCQF. The CQFI field is never explicitly shown in the sequences.
	5	Optional Segments Indicator (OSI): 0 no optional segments are present (¬OS) 1 one or more optional segments are present (OS) Sequences: If any optional segments are present the value of this field defaults to OS; otherwise, it defaults to ¬OS. This field is never shown in the sequences.
	6-7	reserved
6-7		DATA OFFSET/4: The position of the DATA relative to the beginning of the THDR. This position is always constrained to be a multiple of 4 bytes. The DATA OFFSET/4 field carries the DATA offset value divided by 4. Sequences: The value of this field always defaults to the appropriate value. It is never explicitly shown in the sequences.
8-11		DATA Length Field (DLF): The exact number of bytes carried in the DATA field. Sequences: The value of this field always defaults to the appropriate value. It is never explicilty shown in the sequences.

Table A-5 (Page 3 of 4). RTP Transport Header (THDR)

Byte	Bit	Content
12-15		Byte Sequence Number (BSN): sequence number of first byte of DATA Each DATA byte is (conceptually) assigned a sequence number. The BSN field carries the sequence number of the first byte of the DATA. (When the DATA field is empty, this is the sequence number that will be (or would be) assigned to the first byte of the next non-empty DATA field). Sequences: The value of this field always defaults to the appropriate value. It is never explicilty shown in the sequences.
16-k		Bytes 16-k may (optionally) contain any of the following: Connection Qualifier/Source Identifier Field (CQF): (present if CQFI = ORIGIN) This field contains the Transport Address control vector (X'05') of the originator of the connection setup request. This X'05' control vector must be on word boundary so that subsequent optional segments will also start on word boundary as well. Note that the subvectors (i.e., NETID, NODEID, NCEID) will also start on word boundary, but the length of these subvectors can contain the actual number of bytes of these fields. Sequences: The CQF field defaults to not present unless it is explicitly specified. If specified it contains a Transport Address Control Vector X'05'. For HPR, this control vector contains the network identifier, node identifier (CP name), and NCE identifier. They are specified in the sequences as: `CQF(network identifier, node identifier, NCE identifier).` For example CQF(NETIDa,a,NCE_LUa) means the Connection Qualifier field contains the network identifier for node a, the CP name of node a, and NCE identifier for an LU in node a). For further format details see the description of the Transport Address Control Vector in this chapter. The reason for including the NCE here is so that it can be used (by the receiver of this message) when a new route is obtained for path switch. Assume node A is sending this message to node B to establish a connection. If a link fails during the connection, node B must obtain a new path and resume sending messages to node A. In order to properly send these messages to node A, the NCE (e.g. NCE_LUa) at node A must be known so it can be included in the ANR routing field.

Table A-5 (Page 4 of 4). RTP Transport Header (THDR)

Byte	Bit	Content
k + 1-m		Optional Segment Field (OSF): (at least one present if OSI = OS). Each segment begins on a word boundary and has the following format: Byte Content 0 Segment Length/4: length (in 4-byte multiples) of segment type + segment data 1 Segment Type 2-t Segment Data Optional segments are included in the following order to maximize RTP performance. (The sender includes them in order but the receiver does not check to see that they are in order). X'02' Status Segment X'01' Connection Setup Segment X'04' Connection Identifier Exchange Segment X'06' Routing Information Segment X'07' Adaptive Rate Based Segment X'03' Client "Out of Band" Bits Segment Sequences: The OSF field defaults to empty (no segments present) unless one or more optional segments are explicitly specified.
m + 1-m + 4		reserved

A.1.6 RTP Optional Segments

A.1.6.1 Connection Setup (CS) Segment

Table A-6. Connection Setup (CS) Segment

Byte	Bit	Content
0		Length/4 of the Segment, including the Length field.
1		Key = X'01'
2-3		Version Values used by HPR are: X'0101' Version 1.1
4	0	Target resource identifier present 1 yes - only value used by HPR
	1	Source identifier field present 0 no - only value used by HPR
	2	Windowing flow control usage 0 no - Windowing not used by HPR (it only uses ARB)
	3	ARB flow/congestion control usage 1 yes - ARB is always used by HPR
	4-7	reserved
5-7		reserved
8-k		Subvectors: each subvector begins on a word boundary.
8-j		X'81' Topic Identifier Control Vector For HPR two types of Topic Identifiers are allowed: globally defined and user defined. **Globally defined** For connections that carry CP-CP session traffic the topic ID used is "CPSVCMG". For connections used to carry the Route Setup protocol traffic the topic ID is "RSETUP". **User defined** For connections carrying LU-LU session traffic, the topic ID is set to the user-defined COS name that is associated with the sessions.
j + 1-k		Target resource identifier This field is used to check that the first packet of a connection arrives at its intended target. It contains the following CVs. X'03' Network Identifier Control Vector X'00' Node Identifier Control Vector

Note: Sequences: The Connection Setup segment's presence is shown by the presence of the abbreviation CS in the THDR. The fields within the CS segment are never explicitly shown and default to the values as described above.

A.1.6.2 Status Segment

This segment is used to convey state information from one end of the logical connection to the other.

This segment should be sent as a reply to a packet with the Status Request bit set. It may also be sent as an unsolicited request for retransmission of lost packets, when the receiver detects a "gap" in Byte Sequence numbers of the message stream.

The status information always includes a Received Sequence number which serves to acknowledge Data Payload bytes received. The status "report" may optionally include *Acknowledged Byte Spans* - This is equivalent to an optional "selective repeat/reject" facility.

"STATUS(ack)" (as used in the sequences) means all the data has been successfully received and is being acknowledged as such.

Table A-7 (Page 1 of 2). Status Segment

Byte	Bit	Content
0		Length/4 of the Segment, including the Length field.
1		Key = X'02'
2		Status bits
		Indicates possible exceptional conditions or reasons for state exchange.
	0	GAPDETR - gap detected by the receiver:
		1 One or more packets have been lost. The missing bytes should be retransmitted as soon as possible.
		0 No packets have been lost.
	1	IDLE:
		1 No packets have been received on this connection for a while. This packet is a "heart beat" to let the receiving end know "I'm still alive."
		0 The connection has not been idle.
	2	CLOSED:
		1 No further packets will be received on this connection.
		0 The connection is not closed.
	3	IGNORE GAP/ADVANCING SEQUENCE NUMBER:
		1 The sender is advancing its sequence number with this packet. The receiver should ignore any previous missing bytes and set its RSEQ number for the receive path in the connection context to 1 plus the byte sequence number of the last byte of this packet.
		0 Normal sequence number processing in force.
	4-7	reserved
3		NABS: This is the number of Acknowledged Byte Span pairs that appear at the end of the Status Segment.
4-5		SYNC - Status Report Number: Each status report is numbered by the sender. This numbering can be used to distinguish new from stale data. Also see ECHO.
6-7		ECHO - Status Acknowledgment Number: This is the most recent SYNC number that was received by the partner that is sending this packet. Together, SYNC and ECHO can be used to determine when an exchange of state information has been effected.

Table A-7 (Page 2 of 2). Status Segment

Byte	Bit	Content
8-11		RSEQ - Received Sequence number: This is 1 plus the Byte Sequence number of the most recently received user message (or end of message) byte. In Retry mode, RSEQ refers to the most recently received byte without an earlier gap in the user data stream. Thus, in Retry mode RSEQ acknowledges all message (and end of message) bytes preceding Byte Sequence number RSEQ.
12-15		DSEQ - Delivered Sequence number: This is 1 plus the Byte Sequence number of the most recently *delivered* user message (or end of message) byte. A byte is deemed *delivered* when it is transferred to the buffer area specified by a transport user's Receive verb, and the user is notified that the buffer is ready for processing.
16-19		ASEQ - Allocation Sequence number: The reporting end (has allocated a "window" and) agrees to receive payload bytes with Bytes Sequence numbers up to, but not including, the Allocation Sequence number. In other words, this status report indicates that the receiver's window begins at Byte Sequence number RSEQ and ends at Byte Sequence number ASEQ-1. Additionally, Byte Sequence number ASEQ is permitted, if that is an End of Message Character. *Note* The remainder of the segment consists of NABS (zero or more) Acknowledged Byte Span Pairs (ABSP): Each ABSP represents a sequence of bytes in the receiver's window (between RSEQ and ASEQ) that are being held in the receiver's buffer(s) pending arrival of the "gaps." Whether to implement ABS processing and how many ABS pairs to support is an implementation choice. The receiver can choose how many "spans" to buffer. The sender is always free to retransmit any and all bytes starting at RSEQ. However on connections with large bandwidth-delay products it has been shown that keeping track of even one span beyond RSEQ can greatly improve the "link" efficiency.
20 + (i-1)*8- 23 + (i-1)*8		ABSBEG: The Byte Sequence Number of the first byte in the *ith* Acknowledged Span
24 + (i-1)*8- 27 + (i-1)*8		ABSEND: This is 1 plus the Byte Sequence Number of the last user message (or end of message) byte in the *ith* Acknowledged Span. The sender is no longer obligated to buffer the bytes that are in an Acknowledged Byte Span, from ABSBEG to ABSEND-1.

A.1.6.3 Client Out of Band Bits (COB) Segment

Table A-8. Client Out of Band Bits (COB) Segment

Byte	Bit	Content
0		Length/4 of the Segment, including the Length field.
1		Key = X'03'
2-3		Client bits Values defined for HPR: X'0001' Request Deactivation X'8000' Reply - OK X'8004' Reply - Reject

A.1.6.4 Connection Identifier Exchange (CIE) Segment

This segment is used by the partner that was originally the "listener" to offer an alternative TCID to the "calling" partner.

CIE(TCIDxyn) is used in the sequences to show that a CONNECTION IDENTIFIER EXCHANGE segment is present and contains an alternative connection identifier of TCIDxyn.

Table A-9. Connection Identifier Exchange (CIE) Segment

Byte	Bit	Content
0		Length/4 of the Segment, including the Length field.
1		Key = X'04'
2-3		reserved
4-7		Alternative Connection Identifier
	0	Sender/Receiver Chose TCID
		1 The high-order bit of this Alternative Connection Identifier field should be 1, indicating that the sender has chosen the identifier.
	1-31	The remaining 31 bits are the proper alternative connection identifier

A.1.6.5 RTP Routing Information (RI) Segment

This segment is present when the Connection Setup segment is present and when it's necessary to convey new path information to the partner on a path switch. It includes, among other things, the ANR that is to be used by the receiver when sending data on this RTP connection.

Table A-10. Routing Information (RI) Segment

Byte	Bit	Content
0		Length/4 of the Segment, including the Length field.
1		Key = X'06'
2-3		reserved
4-n		Routing Information For HPR the following control vectors are included (in LT form): X'83' HPR Routing Information CV X'85' HPR Return Route TG Descriptor CV

A.1.6.6 Adaptive Rate-Based (ARB) Segment

Newly defined segment for HPR. See 7.2, "HPR Usage of Rapid Transport Protocol (RTP)" on page 7-3 for details on how this field is used.

Table A-11. Adaptive Rate-Based (ARB) Segment

Byte	Bit	Content
0		Length/4 of segment including this length field
1		Key=X'07' - Adaptive Rate Based Information
2	0-1	Message Type
		00 ARB Connection Setup (SETUP): This value is used in conjunction with the Connection Setup and RTP Routing Information segments at connection setup time, and with Routing Information segment after a path switch has occurred. At the initial connection setup time, the forward and the return paths are symmetric, so the receiver of this ARB segment will utilize the information in all these fields (FIELD1 through FIELD4) to initialize the ARB algorithm. When a path switch occurs, the receiver will accept all information as in the case at connection set up time described above. FIELD1 contains the length of the measurement interval (in milliseconds). Currently, this field is always set to 256. It is included here for future flexibility. FIELD2 contains the maximum allowed network queueing time (in milliseconds) along the path for this connection. FIELD3 contains the initial rate information for this connection (Kbits/sec). FIELD4 contains the minimum incremental rate information (Kbits/sec).
		01 Rate Reply (REPLY): a reply to a Rate Request message FIELD1 is reserved. FIELD2 contains the receiving rate information for the sender of this message (Kbits/sec).
		10 Rate Request (RQ): the receive rate (for the receiver of this message) is requested. FIELD1 contains the sender's (sender of this message) measurement interval (milliseconds). FIELD2 is reserved.
		11 Rate Request and Rate Reply (RQ/REPLY): this message combines a Rate Request and Rate Reply into a single message. FIELD1 contains the sender's (sender of this message) measurement interval (Kbits/sec). FIELD2 contains the receive rate information for the sender of this message (Kbits/sec).
	2-7	reserved
3		reserved
4-7		FIELD1: content of this field depends on the Message Type field value
8-11		FIELD2: content of this field depends on the Message Type field value
12-15		FIELD3: content (and presence) of this field depends on the Message Type field value
16-19		FIELD4: content (and presence) of this field depends on the Message Type field value

Note: FIELD1 and FIELD2 are always present, FIELD3 and FIELD4 are only present when the message type is 00.

Sequences: The ARB segment is specified in the sequences as one of the following:

- "ARB(Message Type, Field1, Field2)" - fields are explicitly specified.
- "ARB" - Individual fields are not specified and are assumed to be set correctly.

A.1.7 RTP Control Vectors

This section contains control vectors that may be included in the THDR or optional segments in the THDR.

A.1.7.1 Node Identifier Control Vector (X'00')

This CV identifies a node and always contains the non-qualified CP name of the node.

Table A-12. Node Identifier Control Vector (X'00')

Byte	Bit	Content
0		Length, in binary, including the length field and any padding necessary to force embedded control vectors to start on 4-byte boundaries.
1		Key = X'00'
2-n		Node identifier: a 1 to 8-byte string of characters, the first byte being limited to uppercase letters, from character set 1134. The node identifier is unique when qualified by the X'03' Network Identifier control vector.

A.1.7.2 Network Identifier Control Vector (X'03')

This CV identifies a network.

Table A-13. Network Identifier Control Vector (X'03')

Byte	Bit	Content
0		Length, in binary, including the length field and any padding necessary to force embedded control vectors to start on 4-byte boundaries.
1		Key = X'03'
2-n		Network identifier: a unique 1- to 8-byte type 1134 string, the first byte being limited to uppercase letters.

A.1.7.3 Transport Address Control Vector (X'05')

This CV is used in the Connection Qualifier Field in the THDR.

Table A-14. Transport Address Control Vector (X'05')

Byte	Bit	Content
0		Length, in binary, including the length field and any padding necessary to force embedded control vectors to start on 4-byte boundaries.
1		Key = X'05'
2	0	Transport address type 1 Transport address is for point-to-point connection Sequences: This field is always set to 1 and is never explicitly shown.
	1-7	reserved
3		reserved
4-n		Subvectors: note that each of the subcontrol vectors start on a word boundary but, as indicated, their associated length field contains the exact number of bytes of each field. X'03' Network Identifier Control Vector X'00' Node Identifier (CP name) Control Vector X'26' NCE Identifier Control Vector

A.1.7.4 NCE Identifier Control Vector (X'26')

This CV identifies an NCE.

Table A-15. NCE Identifier Control Vector (X'26')

Byte	Bit	Content
0		Length, in binary, including the length field and any padding necessary to force embedded control vectors to start on 4-byte boundaries.
1		Key = X'26'
2-n		NCE identifier: a 1- to 8-byte EBCDIC string.

A.1.7.5 Topic Identifier Control Vector (X'81')

Identifies the intended "listening" application.

Table A-16. Topic Identifier Control Vector (X'81')

Byte	Bit	Content
0		Length, in binary, including the length field and any padding necessary to force embedded control vectors to start on 4-byte boundaries.
1		Key = X'81'
2	0	User defined 0 This topic identifier is globally unique 1 This topic identifier is user defined
	1-7	reserved
3-n		Topic identifier: a string of characters from character set 1134.

A.1.7.6 HPR Routing Information CV (X'83')

Table A-17. HPR Routing Information CV

Byte	Bit	Content
0		Length, in binary, including the length field and any padding necessary to force embedded control vectors to start on 4-byte boundaries.
1		Key = X'83'
2	0	REFIFO indicator 0 No - do not allow for normal operation REFIFOing 1 Yes - allow for normal operation REFIFOing
	1	Path switch controller indicator 0 The origin is not a path switch controller 1 The origin is a path switch controller
	2-7	reserved
3-5		reserved
6-7		Network header: bytes 0 and 1 described in A.1.4, "Network Layer Header (NHDR)" on page A-5. This field will be included in front of the ANR field below to form the complete return ANR route in the NHDR of packets sent on the return path. It indicates such things as the transmission priority associated with this connection.
8-11		Maximum packet size on the return path (in bytes)
12-15		Maximum path switching time for this connection (in milliseconds).
16-19		RTP Liveness (ALIVE) timer value (in seconds): For description of how this field is set and used see 7.2.4.1.3.1, "ALIVE Timer" on page 7-14.
20-21		Length of Return ANR
22-m		Return ANR This is the actual ANR field of the return path and it includes the appropriate NCE and is delimited by X'FF'. This field will be attached to bytes 6-7 field above to form the complete return ANR route.
m + 1		Length of Mode name field
m + 2-n		Mode name: 0 to 8 type-A symbol-string characters with optional (but not significant) trailing blanks.

A.1.7.7 HPR Return Route TG Descriptor CV (X'85')

This CV contains the description of the reverse route in terms of TG numbers and CP names (just like in an RSCV). It consists of a series of CV 46's. This CV describes the same route as specified in the Return ANR field in the HPR Routing Information CV. It is used by the remote RTP partner (receiver of this CV) to determine if sessions may be carried by this RTP connection.

Table A-18. HPR Return Route TG Descriptor CV

Byte	Bit	Content
0		Length, in binary, including the length field and any padding necessary to force embedded control vectors to start on 4-byte boundaries.
1		Key = X'85'
2-n		A series of CV 46's, in LT format, that describe the return route.

A.1.8 FID5 TH

Table A-19. FID5 TH

Byte	Bit	Content
0	0-3	FID5-Format Identification: 0101 Sequences: The FID5 TH is shown in the sequences as TH_FID5.
	4-5	MPF-Mapping Field (same as in FID2) 10 first segment of a BIU (BBIU, ¬EBIU) 00 middle segment of a BIU (¬BBIU, ¬EBIU) 01 last segment of a BIU (¬BBIU, EBIU) 11 whole BIU (BBIU, EBIU) Sequences: The default value is whole BIU.
	6	reserved (was ODAI field in FID2)
	7	EFI-Expedited Flow indicator 0 normal flow (NORMAL) 1 expedited flow (EXP) Sequences: The default value is NORMAL.
1		reserved
2-3		SNF-Sequence Number Field Sequences: The default value is whatever the appropriate value should be.
4-7		SA-Session Address (replaces OAF', DAF', and ODAI fields in FID2) There are 2 addresses associated with each session - one in each direction. The receiver node assigns the address used in the session traffic being received. Sequences: This field is always specified as SAxy where x and y identify the two nodes using the address. For example, SAab indicates a session address that is sent from node a to node b.

A.1.9 HPR capabilities control vector (CV61)

This CV is meaningful only on negotiation-proceeding XID3. Its presence indicates that it is desired that the link run HPR protocols. If it is received on a pre-negotiation XID3, it is ignored. It is used in KL form.

Table A-20. HPR capabilities control vector (CV61)

Byte	Bit	Content
0		Key = X'61'
1		Length of Control Vector Data (2-n)
2	0-1	Error recovery mode: this field indicates whether or not error recovery is required or preferred on this link (e.g. error recovery that is done by the LLC layer) for NLPs. Note: FID2 packets always require error recovery independent of the setting of this field. 00 Error recovery is required (ERP) 01 No error recovery is required (i.e., it is required NOT to do error recovery) (¬ERP) 10 Prefer no error recovery but will do it if partner wants it (*ERP) 11 reserved
	2-7	reserved
3		reserved
4-n		*Subfields* The following subfields may be present. They are parsed according to the subfield parsing rule LT (length followed by key). X'80' Token Ring LLC subfield (present only when exchanging XIDs over a token ring). X'81' HPR Transport Tower subfield (present only when the node sending the XID supports the HPR Transport Tower).

A.1.10 Token Ring 802.2 LLC subfield (X'80')

This subfield is included in CV 61 when XID3 is being exchanged over a token ring. The reason for making this a subfield (with variable length) is for future expandability.

Table A-21. Token Ring LLC subfield (X'80')

Byte	Bit	Content
0		Length of the subfield including this field
1		Key: X'80'
2-n		LLC SAP - this field contains the LLC SAP that is to be used by the adjacent node when sending NLPs (that do not require link-level error recovery) to this node (i.e., the adjacent nodes uses this field as the destination LLC SAP). The default value for this field is X'C8' which is an IBM reserved value assigned for HPR. Products/customers may use a different value if necessary. For NLPs (and FID2 PIUs) that use link-level error recovery, the LLC SAP values are as in today's APPN (see A.2.5, "Token Ring LLC 802.2 format for XID, FID2 PIUs, and HPR NLPs when using link level error recovery" on page A-37). Currently, all LLC SAPs are exactly one byte long. It is contained in this variable length subfield for possible future expansion.

Note: This subfield is not used for direct communication over a Frame Relay link even though Frame Relay uses LLC SAPs.

A.1.11 HPR Transport Tower subfield (X'81')

This subfield is included in CV 61 (on XID3) by nodes that support the HPR Transport Tower.

Nodes that do not support this tower do not send this subfield but, when received, do recognize it's presence (but not the content). If the subfield is present then the TG is reported as going to a node that supports the HPR Transport Tower.

The purpose of this subfield is to perform the "Route Setup" function for CP-CP session and long-lived Route Setup RTP connections (since a real Route Setup protocol is not done for these connections).

Table A-22. HPR Transport Tower subfield (X'81')

Byte	Bit	Content
0		Length of the subfield including this field
1		Key: X'81'
2-j		*Fields used for both CP-CP session and Route Setup RTP connections.*
2-3		Maximum send packet size
4		ANR Label length
5-j		ANR Label: The ANR label for this link (TG) in the direction from this node to the adjacent node.
j + 1-k		*Fields used only for CP-CP session RTP connections.*
j + 1		Length of Control Point NCE identifier
j + 2-k		Control Point NCE identifier: This NCE represents the CP in the node sending this CV and is used by the adjacent node when sending NLPs to this CP over a CP-CP session RTP connection. It is the only ANR label in the ANR Routing field of the NHDR (see A.1.4, "Network Layer Header (NHDR)" on page A-5). Sequences: It is specified as NCE_CPxy which means it is the NCE to be used by node x when sending to the CP in node y.
k + 1-n		*Fields used only for Route Setup RTP connections.*
k + 1		Length of Route Setup NCE identifier
k + 2-n		Route Setup NCE identifier: This field identifies the Route Setup component associated with this link for the sender of this XID. Sequences: It is specified as NCE_RSxy which means it is the NCE to be used by node x when sending to the Route Setup component in node y.

A.1.12 NCE identifier CV (CV62)

This CV is used in CD-Initiate on the Locate reply and the Route Setup reply. It uses LT form.

Table A-23. NCE Identifier CV (CV62)

Byte	Bit	Content
0		Length of the control vector including this field
1		Key = X'62'
2-n		NCE Identifier: indicates a component within a node that processes a received NLP with an ANR that has as its next hop this NCE.

A.1.13 Session Address Control Vector (CV65)

This CV is used on positive BIND responses in KL format. It is used to convey the session address to be used by the primary LU when sending data to the secondary LU. It is assigned by the secondary LU. See 7.9, "Enhanced Session Addressing" on page 7-48. It is included on BIND responses that are carried in FID5 PIUs over RTP connections. It is not included on BIND responses carried in FID2 PIUs.

Table A-24. Session Address CV (CV65)

Byte	Bit	Content
0		Key = X'65'
1		Length of Session Address
2-5		Session Address - the 4-byte, unique per RTP connection, session address used by HPR nodes. Sequences: The value of this field is specified in the form SAxy which indicates the session address used when sending data in the direction Node x to Node y.

A.1.14 FID2 Route Setup

This is the format of the Route Setup when sent between nodes where one or both do not support the HPR Transport tower. In this case the Route Setup is carried in a FID2 PIU.

Table A-25. FID2 Route Setup

Byte	Bit	Content
0-5		FID2 TH
0	0-3	FID2 Format Identification: 0010
	4-5	MPF - Mapping field 11 whole segment FID2 Route Setup is never segmented because it will always be less then 768 and 768 is the minimum link frame size allowed for HPR.
	6	ODAI: 0
	7	EFI: 1
1		reserved
2		DAF: X'00'
3		OAF: X'00'
4-5		SNF: X'0000'
6-8		RH
6	0	Request/Response Indicator: 0 (request)
	1-2	RU Category: 01 (network control) This is the field that distinguishes this FID2 from all the others.
	3	reserved
	4	Format Indicator: 1 (formatted)
	5	Sense Data Included: 0 (no sense data included)
	6	Begin Chain Indicator: 1 (begin chain)
	7	End Chain Indicator: 1 (end chain)
7	0	DR1: 0
	1	Length Checked Compression Indicator: 0
	2	DR2: 0
	3	ERI: 0
	4	reserved
	5	Request Larger Window Indicator: 0
	6	Queued Response Indicator: 0
	7	Pacing Indicator: 0
8	0	Begin Bracket Indicator: 0
	1	End Bracket Indicator: 0
	2	Change Direction Indicator: 0
	3	reserved
	4	Code Selection Indicator: 0
	5	Enciphered Data Indicator: 0
	6	Padded Data Indicator: 0
	7	Conditional End Bracket Indicator: 0
9-n		Route Setup RU
9		X'10' - Route Setup request code
10-n		Route Setup GDS variable X'12CE' (see A.1.15, "Route Setup GDS variable X'12CE'" on page A-31).

A.1.15 Route Setup GDS variable X'12CE'

Table A-26 (Page 1 of 2). Route Setup GDS variable X'12CE'

Byte	Bit	Content
0-1		Length of the GDS variable including this length field (0-n)
2-3		GDS ID X'12CE' (Route Setup)
4-n		GDS Variable Data
4	0	Type (TYPE): 0 Request (RQ) 1 Reply (REPLY) Sequences: The value of this field is always explicitly specified.
	1	Destination Node type: This field is only used on REPLY's and indicates the node type of the destination node (i.e. the node that originates the sending of the route setup reply). 0 EN 1 NN all other values are reserved
	2	REFIFO indicator: indicates whether or not the transport component (RTP) will, as part of normal operation (i.e. with no errors occurring), receive data traffic that arrives out of order. 0 No - do not allow for normal operation REFIFOing 1 Yes - allow for normal operation REFIFOing
	3	Route Setup required on path switch indicator. This indicator is set by a subnet (e.g. CNN) when it receives a Route Setup request and is used by the RTP end points to determine if a Route Setup is needed when doing a path switch. 0 Route Setup is not required on path switch 1 Route Setup is required on path switch
	4	Path switch controller indicator 0 The destination is not a path switch controller 1 The destination is a path switch controller
	5-7	reserved
5		reserved
6		Destination hop index - contains the index (integer) into the RSCV for the node that will send the Route Setup reply (and eventually become the partner RTP end point). When a node receives a Route Setup request it uses this field to determine if it is to be the final destination (i.e. send the Route Setup reply).
7		reserved
8-11		Limit resource liveness timer value (in seconds): Each limited resource link along the path has a liveness timer value associated with it. The purpose of this field is to obtain the smallest liveness timer value of all the limited resource links along the path.
12-n		Control vectors: all control vectors on Route Setup are parsed according to the subfield parsing rule LT. The following control vectors may be included on a Route Setup request. X'80' Forward Route Information - always present. This CV is used to accumulate information about the forward route. See A.1.16, "Route Information Control Vector (CV80)" on page A-33. X'0E' LU name (F3) - Contains the destination LU name. X'2B' RSCV - is used to direct the flow of the Route Setup request. Sequences: specified as RSCV(a-b-c...) where a, b, c. etc. are the TG numbers representing the links along the route.

Table A-26 (Page 2 of 2). Route Setup GDS variable X'12CE'

Byte	Bit	Content
		X'60' FQPCID - this is the fully qualified PCID associated with the session setup procedure that has caused this Route Setup request to flow. It is used by the originator and the nodes along the path of the setup to correlate the Route Setup reply to the Route Setup request. Sequences: assumed to be present and set correctly. X'2C' COS/TPF - indicates the COS and TPF for this route and is always present. It is used for routing through subnets like CNN. X'2D' Mode Control Vector - contains the mode name and is always present. This field is used by CNNs to map to the subarea COS. Control vectors for Route Setup reply. The control vectors included on a positive Route Setup reply are X'80' (forward route information), X'80' (reverse route information), X'2B', X'60', and X'62'. The control vectors included on a negative Route Setup reply are X'80' (for foward route information), X'60', and X'35'. X'80' Forward Route Information - always present. See A.1.16, "Route Information Control Vector (CV80)" on page A-33. X'80' Reverse Route Information. This CV is only present on a positive reply where it is used to accumulate information about the reverse route. See A.1.16, "Route Information Control Vector (CV80)" on page A-33. X'2B' RSCV - present only on a positive reply. Sequences: Specified as RSCV(a-b-c...) where a, b, c. etc. are the TG numbers representing the links along the route. X'60' FQPCID - The same as was received on the Route Setup request. This field is used to correlate the Route Setup reply to the request. X'62' NCE - Contains the NCE associated with the destination LU or, if the next hop is an APPN TG (i.e. not HPR), the NCE associated with the boundary function that performs the translation between HPR and APPN for this path. This CV is only present on a positive reply. X'35' Extended Sense Data - this field is included on a negative Route Setup reply to indicate that the Route Setup protocol was unsuccessful. The sense code indicates the type of error. If this CV is not present then the reply is positive (i.e. the Route Setup protocol was successful). The CV35 will also indicate which node detected the error so as to faciltate network mananegment.

A.1.16 Route Information Control Vector (CV80)

This CV is used in the Route Setup GDS variable in the LT format. It contains information about either the forward and/or reverse route that is to be used by an RTP connection. The term "Route Setup" is used to refer to either a Route Setup request or a Route Setup reply. The term "next hop" refers to the link that the Route Setup request or reply will be sent out on.

Table A-27. Route Information Control Vector (CV80)

Byte	Bit	Content
0		Length
1		Key = X'80'
2	0	Route direction 0 Forward: information about the forward route is collected on the Route Setup request. 1 Reverse: information about the reverse route is collected on the Route Setup reply.
	1-7	reserved
3		reserved
4-7		Maximum Packet Size If the maximum packet size for the next hop (TG) is less than the current value in the received Route Setup, then the next hop maximum packet size value is stored in the Route Setup.
8-11		Accumulated transmission time (in micro-seconds for 1200 bits) The transmission time for the next TG is added to the current value in the received Route Setup.
12-15		Minimum link capacity (in Kbits per second) If the link capacity for the next TG is less than the value in the received Route Setup then this lesser value is stored in this field on the Route Setup.
16		Length of Accumulated ANR String
17-n		Accumulated ANR String This field is updated at each hop by appending the next TG's ANR label to the end (right) of the received string value (in the received Route Setup). This field does not contain the destination's endpoint NCE. Sequences: Always specified as FANR|RANR(AL1-...-ALn) where FANR indicates forward information and RANR indicates reverse information.

A.2 Existing formats modified for HPR

Fields added or changed for HPR are indicated by an '*'.

A.2.1 Frame Relay format for XID3 and FID2 Route Setup

The format described here is documented in "Protocol Encapsulation over Frame Relay Implemention Agreements - TRF 92.32R2" by Rao Cherukuri. It is the same format as is used by today's APPN (FID2).

Table A-28. Frame Relay format for FID2 Route Setup

Byte	Bit	Content
0-1		T1.618 address (DLCI)
2		Control: X'03'
3		NLPID: X'08'
4-5		L2 Protocol Identifier: X'4C80' - indicates presense of 802.2 header
6-7		L3 Protocol Identifier: X'7083' - SNA-APPN(FID2)
8-11		802.2 header
8		DSAP: same as in today's APPN (e.g. X'04')
9		SSAP: same as in today's APPN (e.g. X'04')
10-11		Control fields: set as appropriate
12-n		Remainder of PDU: contains XID3 or Route Setup FID2 PIU
n + 1-n + 2		FCS

A.2.2 Frame Relay format for HPR NLPs when not using link level error recovery

The format described here will eventually be documented in "Protocol Encapsulation over Frame Relay Implemention Agreements - TRF 92.32R2" by Rao Cherukuri.

Table A-29. Frame Relay format for HPR NLPs when not using link level error recovery

	Byte	Bit	Content
	0-1		T1.618 address (DLCI)
	2		Control: X'03'
	3		NLPID: X'08'
*	4-5		L2 Protocol Identifier: X'7081' - indicates no L2 protocol
*	6-7		L3 Protocol Identifier: X'7085' - SNA-APPN/HPR(NLP)
*	8-n		Remainder of PDU: contains HPR NLP
	n + 1-n + 2		FCS

A.2.3 Frame Relay format for HPR NLPs when using link level error recovery

The format described here will eventually be documented in "Protocol Encapsulation over Frame Relay Implemention Agreements - TRF 92.32R2" by Rao Cherukuri.

Table A-30. Frame Relay format for HPR NLPs when using link level error recovery

	Byte	Bit	Content
	0-1		T1.618 address (DLCI)
	2		Control: X'03'
	3		NLPID: X'08'
	4-5		L2 Protocol Identifier: X'4C80' - indicates presense of 802.2 header
*	6-7		L3 Protocol Identifier: X'7085' - SNA-APPN/HPR(NLP)
	8-11		802.2 header
	8		DSAP: same as in today's APPN (e.g. X'04')
	9		SSAP: same as in today's APPN (e.g. X'04')
	10-11		Control fields: set as appropriate
*	12-n		Remainder of PDU: contains HPR NLP
	n+1-n+2		FCS

A.2.4 Token Ring LLC 802.2 format for HPR NLPs when not using link level error recovery

Table A-31. Token Ring LLC 802.2 format for HPR NLPs when not using link level error recovery

Byte	Bit	Content
0		DSAP: X'XX' - SNA-APPN/HPR(non-recoverable NLP) XX is the LLC destination SAP value that is used for transmitting HPR NLPs without performing link-level error recovery on them. The value used here is the one received on XID3 from the adjacent node during the XID3 link activation exchange (see A.1.10, "Token Ring 802.2 LLC subfield (X'80')" on page A-26).
1		SSAP: X'XX' - SNA-APPN/HPR(non-recoverable NLP) XX is the LLC source SAP value that is used for transmitting HPR NLPs without performing link-level error recovery on them. The value used here is the one sent in XID3 by this node during the XID3 link activation exchange (see A.1.10, "Token Ring 802.2 LLC subfield (X'80')" on page A-26).
2		Control field: X'03' unnumbered information

A.2.5 Token Ring LLC 802.2 format for XID, FID2 PIUs, and HPR NLPs when using link level error recovery

The format of this field is the same as used in today's APPN.

Table A-32. Token Ring LLC 802.2 format for HPR NLPs when using link level error recovery

Byte	Bit	Content
0		DSAP: same as in today's APPN (e.g. X'04') - SNA-APPN/HPR(recoverable NLP)
1		SSAP: same as in today's APPN (e.g. X'04') - SNA-APPN/HPR(recoverable NLP)
2-3		Control fields: set as appropriate

A.2.6 Node characteristics (CV4580)

Table A-33. Node characteristics (CV4580)

Byte	Bit	Content
0		Length of CV including this length field.
1		Key = X'80'
2-n		Control Vector Data
2-8		same as currently defined for APPN
9(=n)		Additional Node Support
	0	Adjacent subnet border node support
		0 The node lacks such support
		1 The node has such support
	1	Interchange node support:
		0 The node lacks such support
		1 The node has such support
	2	Intermediate border node support:
		0 The node lacks such support
		1 The node has such support
*	3-4	HPR support level: this field is defined solely for network management purposes.
*		00 No HPR support
*		01 Supports HPR but not the HPR Transport Tower
*		10 Supports HPR and also supports the HPR Transport Tower
*		11 reserved
	5-7	reserved

Note: These fields are newly defined for HPR and are there solely for network management purposes.

A.2.7 TG Identifier TG Descriptor Subfield (X'4680')

Table A-34. TG Identifier TG Descriptor Subfield (X'4680')

Byte	Bit	Content
0		Length, in binary, of TG Identifier subfield
1		Key: X'80'
2		TG number: the binary integer negotiated during XID exchange to represent the TG to the partner node on the TG.
3		Length, in binary, of TG-partner node's network-qualified CP name; values 0 to 17 are valid.
4-n		TG-partner node's network-qualified CP name: the name of the CP in the node at the opposite end of the TG.
n + 1	0	Link connection network indicator: 0 The TG-Partner Node's Network-Qualified CP Name field does not identify a link connection network (e.g. a local area network). 1 The TG-Partner Node's Network-Qualified CP Name field does identify a link connection network; in the case, bytes 4-n contain the CP name representing the virtual routing node.
*	1	Additional configuration information indicator
*		Additional configuration information, used for network management, may be associated
*		with this TG (e.g., subarea routing information within a composite network node)
*		regarding the session path described in the Route Selection Control Vector (X'2B'). This
*		setting is only valid when the subject TG Descriptor is contained within X'2B' CV.
*		0 Additional information is not associated with the TG
*		1 Additional information may be associated with the TG
*	2-4	TG type:
*		000 Boundary Function based TG or APPN TG
*		001 Interchange TG
*		010 Virtual Route based TG
*		011 HPR Transport TG: this TG goes to a node that supports the HPR Transport tower.
*		100 HPR Non-transport TG: this TG goes to a node that does not support the HPR Transport tower.
*		101 reserved
*		110 reserved
*		111 reserved
	5	Intersubnet link indicator: 0 This link is not an intersubnet link 1 This link is an intersubnet link (defines a border between subnetworks).
	6-7	reserved
n + 2-n + 5		Subarea number: In topology updates, this field contains the subarea number of the node identified in the TG-partner node's CP|PU Name field when this node does not have a subarea address, or the subarea number of this node if it has a subarea address. In the former case, the high-order bit of the subarea number is 0; in the latter, 1. If appended on XID3, this field contains the subarea address of the sending node if it has one, and the high-order bit of the address is always 0.

A.2.8 BIND

Table A-35. BIND

Byte	Bit	Content
0-r		Exactly as defined in today's APPN
r + 1-s		Control vectors: control vectors included for HPR are the same as those currently defined in today's APPN except for the addition of CV X'65' which is new for HPR. X'65' Session Address Control Vector: included on the BIND response for FID5 BINDs carried over RTP connections. See A.1.13, "Session Address Control Vector (CV65)" on page A-29.

A.3 Existing formats not modified for HPR

A.3.1 FID2 TH

HPR is not making any changes to FID2. It is included here just to show the default field settings used in the sequences.

Table A-36. FID2 TH

Byte	Bit	Content
0	0-3	FID2-Format Identification: 0010 Sequences: The FID2 TH is shown in the sequences as TH_FID2.
	4-5	MPF-Mapping Field (same as in FID2) 10 first segment of a BIU (BBIU, ¬EBIU) 00 middle segment of a BIU (¬BBIU, ¬EBIU) 01 last segment of a BIU (¬BBIU, EBIU) 11 whole BIU (BBIU, EBIU) Sequences: The default is whole BIU.
	6	ODAI field Sequences: An LFSID is always specified which includes ODAI, OAF', and DAF' fields.
	7	EFI-Expedited Flow indicator 0 normal flow (NORMAL) 1 expedited flow (EXP) Sequences: The default value is NORMAL.
1		reserved
2		DAF' Sequences: An LFSID is always specified which includes ODAI, OAF', and DAF' fields.
3		OAF' Sequences: An LFSID is always specified which includes ODAI, OAF', and DAF' fields.
4-5		SNF-Sequence Number Field Sequences: The value of this field always defaults to the appropriate value. It is never explicitly shown in the sequences.

Appendix B. Sequence Notation

This chapter describes the notation used in the sequences throughout this document.

B.1 Message fields

- "

  The double quote or ditto (") indicates that the field (or fields) within the message has not changed since the last time it was shown.

- ...

  The ellipses (...) indicates that the field value settings are not applicable to this particular sequence or are set as in today's APPN.

- **Unspecified fields**

  Unspecified fields always assume the default value (as specified in Appendix A, "Formats" on page A-1). This only applies to all new HPR fields and the APPN FID2 TH fields (default values for the FID2 TH fields are also specified in Appendix A, "Formats" on page A-1).

B.2 Links

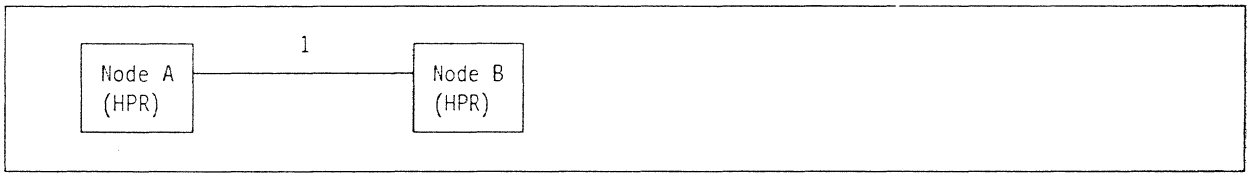


Figure B-1. Link representation

Figure B-1 shows an HPR link, with the number 1, between two HPR nodes, A and B. It has the following properties:

- The TG number associated with the link is 1 (TG numbers are assigned as defined in APPN).
- The ANR from A to B is 1 (left to right).
- The ANR from B to A is represented as 1′ (right to left).

B.3 Network Layer Header (NHDR) notation

See A.1.4, "Network Layer Header (NHDR)" on page A-5 for definitions, abbreviations, and default values used for fields in the Network Header.

B.4 Transport Header (THDR) notation

See A.1.5, "RTP Transport Header (THDR)" on page A-6 for definitions, abbreviations, and default values used for fields in the Transport Header.

B.5 THDR Optional Segments

See A.1.6, "RTP Optional Segments" on page A-10 for definitions, abbreviations, and default values used for fields in the THDR optional segments.

B.6 THs (TH_FID2 and TH_FID5)

See A.3.1, "FID2 TH" on page A-41 and A.1.8, "FID5 TH" on page A-24 for definitions, abbreviations, and default values used for fields in the FID2 and FID5 TH's.

B.7 PIU, BIU, TH, RH, and RU

PIU is the TH, RH, and RU.

BIU is the RH (on beginning segments only), and RU.

The TH is discussed above.

The RH bit settings are always exactly the same as in APPN and so are never explicitly shown in the sequences.

The RUs are described with as much detail as deemed necessary for the particular sequence. For example, the individual fields for a BIND request are not described in detail because they are exactly the same as in APPN.

B.8 ANR Routing

Whenever a packet is ANR routed through an HPR subnet, the intermediate HPR nodes strip off the first ANR label before sending the packet on. This means the packet's ANR Routing field (ANRF)in the NHDR changes at each intermediate node. Instead of explicitly showing this ANR label stripping at each hop (by updating ANRF at each intermediate node) a short-hand notation is used. Intermediate node ANR routing is indicated by the letter "A" under the intermediate node. See Figure B-2 on page B-3.

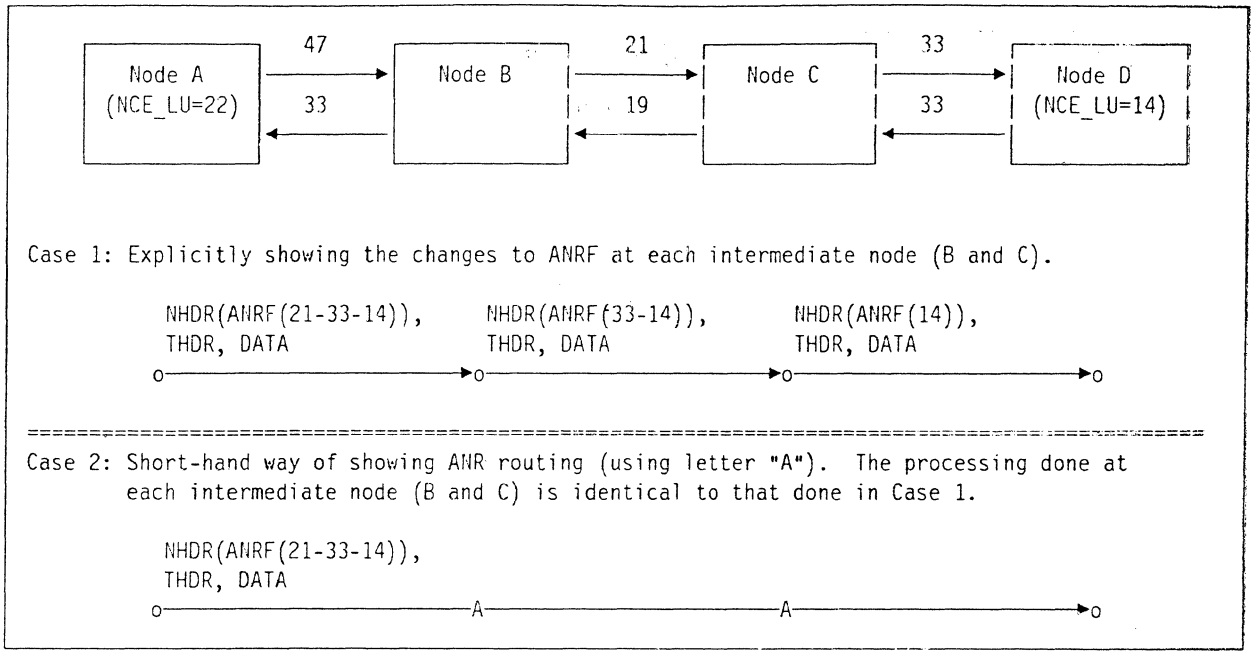


Figure B-2. ANR Routing Short-hand ("A")

B.9 BIUs

A BIU is the portion of the PIU that contains the RH and RU.

B.9.1 RHs

The RH portion of the BIU is never explicitly shown in the sequences. All BIUs used by HPR already exist and the RH is set as in today's APPN.

B.9.2 RUs

RUs may contain user data or control data. The control data is architected and may contain GDS variables and/or control vectors.

B.9.2.1 GDS variables

The following sections describe the notation for GDS variables that have changed for HPR.

- **Route_Setup**

  This GDS variable is new for HPR and indicates a Route Setup. See A.1.15, "Route Setup GDS variable X'12CE'" on page A-31 for a description of the notation used in the sequences.

B.9.2.2 Control Vectors

The following sections describe the notation for new or modified control vectors.

- SA(SAxy)

  This represents the new Session Address control vector (X'65') where SAxy specifies the FID5 session address to be used when sending session traffic from node x to node y.