IBM Library Server Print Preview

DOCNUM = SC30-3346-05
DATETIME = 09/27/93 17:10:48
BLDVERS = 1.2
TITLE = SNA/Management Services Reference
AUTHOR =
COPYR = © Copyright IBM Corp. 1986, 1993
PATH = /home/webapps/epubs/htdocs/book

TITLE Title Page

Systems Network Architecture
Management Services
Reference

Document Number SC30-3346-05

EDITION Edition Notice

```
 ___ Note! ____________________________________________________
|                                                              |
| Before using this information and the products it supports, be |
| sure to read the general information under "Notices" in      |
| topic LEGEND.                                                |
|                                                              |
|______________________________________________________________|
```

| **Sixth Edition (May 1993)**

| This edition, SC30-3346-5, is a major revision of the previous
| edition, SC30-3346-4, and obsoletes that edition; it applies until otherwise indicated in a new edition. Consult Part 3 of the latest edition of *IBM System/370, 30xx, 4300 and 9370 Processors -- Bibliography*, GC20-0370, for current information on this communication architecture. For a summary of the changes in this book, see "Summary of Changes" in topic CHANGES.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.

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

IBM Corporation
Department E70
P.O. Box 12195
Research Triangle Park, North Carolina 27709, U.S.A.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you or restricting your use of it.

| **© Copyright International Business Machines Corporation 1986, 1993.**
| **All rights reserved.**
Note to U.S. Government Users -- Documentation related to restricted rights -- Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

PREFACE Preface

Subtopics:

- PREFACE.1 About this Book
- PREFACE.2 Who Should Use this Book
- PREFACE.3 How This Book is Organized
- PREFACE.4 Prerequisite Publications
- PREFACE.5 Related Publications
- PREFACE.6 Relationship to Systems Application Architecture

PREFACE.1 About this Book

This book provides a comprehensive architectural description of the functions and services associated with Systems Network Architecture/Management Services (SNA/MS). When read in conjunction with the companion books listed below, it defines the protocols and operations associated with the management services component of an SNA type 2.1 node.

PREFACE.2 Who Should Use this Book

This book is intended for those who want to develop a product based on the architecture described herein and for systems programmers and program support personnel who need to understand the architecture at the implementation level of detail. It is intended to complement individual product publications. It does not describe product implementations of the architecture.

PREFACE.3 How This Book is Organized

This book is divided into three parts. It is assumed that the reader of this manual is familiar with the SNA concepts presented in *Systems Network Architecture Concepts and Products*, GC30-3072.

Part I: Introduction to Management Services is introductory information about SNA management services and its categories. An understanding of Chapters 1 and 2 is required before reading the other chapters. Chapters 1 through 7 are organized such that the material can be read straight through. The content is as follows:

° *Chapter 1* introduces:

- The processes required to plan, organize, and control an SNA network
- The management services components of a node
- The choices available to implementations of management services

° *Chapter 2* provides an overview of:

- The management services formats
- The generic management services flows
- Transport of management services data using
  - -- The SSCP-PU session
  - -- SNA/File Services and SNA/Distribution Services
  - -- Multiple-Domain Support (a technique for transporting management services data over LU 6.2 sessions)
- The processing associated with focal points and their spheres of control
- How management services identifies resources

° *Chapters 3, 4, 5, 6, and 7* describe the management services provided to assist with problem management, performance and accounting management, configuration management, change management, and operations management, respectively. Example flows are included.

Part II: Architectural Logic for Management Services is the detailed description of a model implementation of the management services function sets. It consists of chapters 8-10.

It is assumed that the reader of Part II is familiar with Part I. In addition, knowledge of LU 6.2 and SNA/DS is required. (Refer to the list of related publications for more information.)

These chapters provide a concise definition of the management services functions from an implementation perspective. The options and alternatives that implementations may choose are also described.

Part III: Detailed Reference Material contains the following appendixes, as well as the glossary and list of acronyms and abbreviations:

- Appendix A contains Alerts defined for specific environments.
- Appendix B contains management services protocol boundary verbs.
- Appendix C contains SNA/FS file names defined by management services.
- Appendix D contains names of service transaction programs defined by management services.

PREFACE.4 Prerequisite Publications

The following publications are general prerequisites for this book:

Systems Network Architecture Concepts and Products, GC30-3072

| *SNA APPN Architecture Reference*, SC30-3422 (formerly called *SNA Type*
| *2.1 Node Reference*)

The following publication is a preprequisite for the Multiple-Domain Support material:

SNA Transaction Programmer's Reference Manual for LU Type 6.2, GC30-3084

The following publications are prerequisites for the Change Management and SNA/File Services Support material:

SNA/Distribution Services Reference, SC30-3098

SNA/File Services Reference, SC31-6807

PREFACE.5 Related Publications

SNA Formats, GA27-3136

Systems Network Architecture Technical Overview, GC30-3073

SNA/MS Alert Implementation Guide, GC31-6809

Systems Application Architecture SystemView Concepts, SC23-0578

SNA LU 6.2 Reference: Peer Protocols, SC31-6808

SNA Format and Protocol Reference Manual: Architectural Logic, SC30-3112

Token-Ring Network Architecture Reference, SC30-3374

CCITT Recommendation X.21, 1984

| *The X.25 1984/1988 DTE/DCE and DTE/DTE Interface General Information*
| *Manual*, GA27-3761

IBM 5865/5866 Modems Models 2, 3 Maintenance Information and Parts Catalog, SY33-2048

Systems Application Architecture Common Communication Support Summary, GC31-6810

PREFACE.6 Relationship to Systems Application Architecture

Not all of the components of SNA/Management Services are included in Systems Application Architecture(*). The key components of SNA/Management Services that are currently included in Systems Application Architecture (SAA) Common Communications Support are:

° Problem management Alerts

Other features of the architecture are included in a range of products. Consult product specifications regarding the status of the feature for the product in which you are interested.

Subtopics:

- PREFACE.6.1 Relationship to SystemView

PREFACE.6.1 Relationship to SystemView

SystemView(*) is IBM's systems management strategy for planning, coordinating, and operating heterogeneous, enterprise-wide information systems. SystemView builds upon and extends the services and protocols defined by SNA/Management Services. Please consult your IBM representative or the IBM branch office serving your locality for announcements concerning SystemView products and documentation.

Subtopics:

- PREFACE.6.1.1 Relationship to Open Systems Interconnection (OSI)
- PREFACE.6.1.2 Relationship to Simple Network Management Protocol (SNMP)

PREFACE.6.1.1 Relationship to Open Systems Interconnection (OSI)

No OSI-based architecture is contained in this publication. Only SNA-based architecture for Management Services is included. Please consult your IBM representative or the IBM branch office serving your locality for announcements concerning future OSI-based systems management documentation.

| PREFACE.6.1.2 Relationship to Simple Network Management Protocol (SNMP)

| No SNMP-based architecture is contained in this publication. Only
| SNA-based architecture for Management Services is included. Please
| consult your IBM representative or the IBM branch office serving your
| locality for announcements concerning future SNMP-based systems management
| documentation.

CHANGES Summary of Changes

| The sixth edition includes new material for the following network
| management functions:

| ° Material in the MULTIPLE_DOMAIN_SUPPORT and MS_CAPS function sets that
| is implemented by APPN network nodes only. In previous editions, only
| the portions of these function sets applicable to APPN end nodes was
| published.

| ° APPN network node specific Alerts are now published in Appendix A.

| The sixth edition also includes revisions to the following network
| management functions:

| ° The Operations Management Function Set. Clarifications to some
| material in the Operations Management Function Set which was added in
| the previous edition.

| ° In Appendix A, twenty nine Alerts have been assigned their correct
| Alert ID value. The previously assigned Alert ID values continue to be

```
|     published for Alert receivers.


  New material is marked with a revision character ("|") in the left margin.
  Changes primarily editorial in nature, such as corrections or minor
  clarification to the text from the previous edition, are not marked.
```

CONTENTS Table of Contents

Summarize

TITLE Title Page

EDITION Edition Notice

PREFACE Preface
PREFACE.1 About this Book
PREFACE.2 Who Should Use this Book
PREFACE.3 How This Book is Organized
PREFACE.4 Prerequisite Publications
PREFACE.5 Related Publications
PREFACE.6 Relationship to Systems Application Architecture
PREFACE.6.1 Relationship to SystemView
PREFACE.6.1.1 Relationship to Open Systems Interconnection (OSI)
PREFACE.6.1.2 Relationship to Simple Network Management Protocol (SNMP)

CHANGES Summary of Changes

CONTENTS Table of Contents

FIGURES Figures

TABLES Tables

LEGEND Notices

1.0 Part 1. Introduction to Management Services

1.1 Chapter 1. Introduction to Management Services
1.1.1 Network Management and Its Major Categories
1.1.1.1 Problem Management
1.1.1.2 Performance and Accounting Management
1.1.1.3 Configuration Management
1.1.1.4 Change Management
1.1.1.5 Operations Management
1.1.1.5.1 Common Operations Services
1.1.1.5.2 Operations Management
1.1.1.6 Focal Point Services
1.1.2 Introduction to the Management Services Components of the Node
1.1.2.1 SNA Node Types
1.1.2.2 Management Services Roles
1.1.2.3 SNA Networks
1.1.2.3.1 Single-Domain Subarea Networks
1.1.2.3.2 Multiple-Domain APPN Networks
1.1.2.4 Introduction to Physical Unit Management Services (PUMS)
1.1.2.5 Introduction to Control Point Management Services (CPMS)
1.1.3 Overview of Management Services Communication
1.1.3.1 Communication Between CPMS and PUMS
1.1.3.1.1 SSCP-PU Sessions
1.1.3.1.2 SNA/Distribution Services
1.1.3.2 Communication Between CPMS in Different T2.1 Nodes
1.1.3.2.1 Multiple-Domain Support
1.1.3.2.2 SNA/Distribution Services
1.1.4 Implementation Choices
1.1.4.1 Introduction
1.1.4.2 Base and Optional Subsets of the Function Sets
1.1.4.2.1 Rules for Function Sets and Their Subsets
1.1.4.3 Role Requirements
1.1.4.4 Electives

1.2 Chapter 2. Management Services Concepts
1.2.1 Introduction to Management Services Formats
1.2.1.1 Management Services Formats
1.2.1.1.1 Overview of the Management Services Unit (MSU)
1.2.2 Generic Management Services Flows
1.2.2.1 Unsolicited Flows
1.2.2.2 Request Without Reply Flows
1.2.2.3 Request/Reply Flows for Non-Bulk Data
1.2.2.4 Request/Reply Flows for Bulk Data
1.2.3 Transport of Management Services Data
1.2.3.1 Transport of Management Services Data on the SSCP-PU Session
1.2.3.1.1 Overview of the SSCP-PU Session
1.2.3.1.2 Format Used on the SSCP-PU Session
1.2.3.2 Transport of Non-Bulk Management Services Data in APPN Networks
1.2.3.2.1 Overview of Multiple-Domain Support
1.2.3.2.2 Formats for Multiple-Domain Support
1.2.3.2.3 Routing Messages Between Nodes (Overview)
1.2.3.2.4 Routing Messages to Application Programs (Overview)
1.2.3.3 Transport of Bulk Management Services Data
1.2.3.3.1 Introduction to SNA/DS and SNA/FS
1.2.3.3.2 The SNA/FS Server
1.2.3.3.3 Formats for SNA/DS
1.2.3.3.4 How Management Services Uses SNA/FS
1.2.3.3.5 Choice of SNA/DS Roles, Electives, and Options
1.2.3.3.6 Security

1.2.4 **Focal Point Concepts**
1.2.4.1 Types of Focal Points
1.2.4.1.1 Primary Focal Point
1.2.4.1.2 Backup Focal Point
1.2.4.2 Properties of Focal Points
1.2.4.2.1 Sphere of Control
1.2.4.2.2 Takeover by Other Focal Points
1.2.4.2.3 Recovery Properties of Focal Points
1.2.5 **Identification of Resources and Application-Level Routing**
1.2.5.1 Identifying SNA-Addressable Resources
1.2.5.2 Identifying Resources That Are Not SNA-Addressable
1.2.5.2.1 Requests
1.2.5.2.2 Replies and Unsolicited Records
1.2.5.3 Routing a Request to a Specified Application Program

1.3 **Chapter 3. Problem Management**
1.3.1 **Problem Determination**
1.3.1.1 Overview
1.3.1.2 Problem Detection
1.3.1.3 Collection and Analysis of Problem Data
1.3.1.4 Reporting of Problem-Determination Data
1.3.1.5 Recording and Retrieval of Problem-Determination Data
1.3.1.6 Presentation of Problem-Determination Data
1.3.1.7 Processing of Requests for Problem-Determination Data
1.3.2 **Problem Diagnosis**
1.3.2.1 Overview
1.3.2.2 Collection of Diagnostic Data
1.3.2.3 Analysis of Diagnostic Data
1.3.2.4 Processing Requests for Diagnostic Data
1.3.3 **Problem Resolution**
1.3.3.1 Overview
1.3.3.2 Collection of Problem Recovery Data
1.3.3.3 Reporting of Problem Resolution Data
1.3.3.4 Recording and Retrieval of Problem Resolution Data
1.3.3.5 Presentation of Problem Resolution Data
1.3.3.6 Processing of Requests for Problem Resolution Data
1.3.4 **Problem Determination, Diagnosis, and Resolution via Alerts and Resolution Notifications**
1.3.4.1 Functions Provided
1.3.4.2 Relationship of Errors, Problems, and Alerts
1.3.4.3 Variations in Alert Data
1.3.4.4 Implementation
1.3.4.5 Format Usage
1.3.5 **Managing Problems in APPN Networks**
1.3.6 **Link Services**
1.3.6.1 Managing Links
1.3.6.1.1 Links Traversing X.21 Networks
1.3.6.1.2 SDLC Links
1.3.6.1.3 Links Traversing Local Area Networks
1.3.6.1.4 Links Traversing X.25 Packet-Switched Data Networks

1.4 **Chapter 4. Performance and Accounting Management**
1.4.1 **Response-Time Monitoring (RTM)**
1.4.1.1 Functions Provided
1.4.1.1.1 Collection and Analysis of Response-Time Data
1.4.1.1.2 Reporting of Response-Time Data
1.4.1.1.3 Recording and Retrieval of Response-Time Data
1.4.1.1.4 Presentation of Response-Time Data
1.4.1.1.5 Processing and Forwarding of Network Operator Requests
1.4.1.2 Implementation
1.4.1.3 Format Usage
1.4.1.3.1 Setting of RTM Parameters
1.4.1.3.2 Requesting RTM Data and Status
1.4.1.3.3 Sending Unsolicited RTM Data and Status

1.5 **Chapter 5. Configuration Management**
1.5.1 **Query Product ID (QPI)**
1.5.1.1 Functions Provided
1.5.1.1.1 Collection of Product-Identification Data
1.5.1.1.2 Forwarding Requests for Product-Identification data
1.5.1.1.3 Reporting of Product-Identification Data
1.5.1.1.4 Recording and Retrieval of Product-Identification Data
1.5.1.1.5 Presentation of Product-Identification Data
1.5.1.1.6 Processing and Forwarding of Network Operator Requests
1.5.1.2 Implementation
1.5.1.3 Format Usage

1.6 **Chapter 6. Change Management**
1.6.1 **Change Planning**
1.6.1.1 Overview
1.6.1.2 Change Preparation
1.6.1.3 Change Approval
1.6.1.4 Change Scheduling
1.6.2 **Change Control**
1.6.2.1 Overview
1.6.2.2 Change Distribution
1.6.2.3 Change Installation
1.6.2.4 Change Tracking
1.6.2.5 Functions Provided
1.6.2.5.1 Retrieve
1.6.2.5.2 Send
1.6.2.5.3 Delete
1.6.2.5.4 Install
1.6.2.5.5 Initiate
1.6.2.5.6 Send-and-Install
1.6.2.5.7 Send-and-Initiate
1.6.2.5.8 Remove
1.6.2.5.9 Accept
1.6.2.5.10 Cancel
1.6.2.5.11 Role Summary
1.6.2.5.12 Testing Features

1.6.2.6 Implementation
1.6.2.7 Format Usage
1.6.3 **Node Activation**
1.6.3.1 Functions Provided
1.6.3.1.1 Activate
1.6.3.2 Implementation
1.6.3.3 Format Usage
1.6.4 **Example Change Management Plans**
1.6.4.1 Centralized Customization of Microcode
1.6.4.2 Centralized Distribution and Control of Application Software

1.7 **Chapter 7. Operations Management**
1.7.1 **Operations Management**
1.7.1.1 Command Enveloping
1.7.1.2 Sending Messages to Operators
1.7.1.3 Routing to Second-level Application Programs
1.7.1.4 MSU Differences
1.7.2 **Common Operations Services Function Set**
1.7.2.1 Functions Provided
1.7.2.1.1 Enveloping of Data
1.7.2.1.2 Routing to Served Application Programs and Network Operators
1.7.2.1.3 Parameter Major Vectors
1.7.2.1.4 Commands
1.7.2.1.5 Messages to an Operator
1.7.2.1.6 Querying for Information on a Resource
1.7.2.2 Testing a Resource
1.7.2.2.1 Analysis of a Resource
1.7.2.3 Implementation
1.7.2.4 Format Usage
1.7.3 **Operations Management Function Set**
1.7.3.1 Functions Provided
1.7.3.1.1 Operations Management Requests, Replies, and Reports
1.7.3.1.2 Request Activation CP-MSU
1.7.3.1.3 Activation Acceptance CP-MSU
1.7.3.1.4 Activation CP-MSU
1.7.3.1.5 Request Deactivation CP-MSU
1.7.3.1.6 Deactivation Acceptance CP-MSU
1.7.3.1.7 Deactivation CP-MSU
1.7.3.1.8 Set Clock CP-MSU
1.7.3.1.9 Set Clock Report CP-MSU
1.7.3.1.10 Request Cancelation CP-MSU
1.7.3.1.11 Cancelation CP-MSU
1.7.3.1.12 Operate CP-MSU
1.7.3.1.13 Operate Report CP-MSU
1.7.3.1.14 Routing/Parsing Report CP-MSU
1.7.3.2 Implementation
1.7.3.3 Format Usage
1.7.3.3.1 Operations Management Flows

2.0 **Part 2. Architectural Logic for Management Services**

2.8 **Chapter 8. Role Requirements of Management Services Components**
2.8.1 **Role Requirements for Management Services Components**
2.8.1.1 Physical Unit Management Services (PUMS) in a Type 2.0 Node
2.8.1.2 Control Point Management Services (CPMS) in an APPN End Node or Network Node

2.9 **Chapter 9. General Management Services Function Sets**
2.9.1 **Protocol Boundaries in the Management Services Model**
2.9.1.1 Conventions Used in Describing Function Sets
2.9.1.2 Protocol Boundaries Between MS Function Set Groups
2.9.1.3 Protocol Boundary A--Send NMVT
2.9.1.4 Protocol Boundary B--Held Alert Processing
2.9.1.5 Protocol Boundary C--Delayed Alert Processing
2.9.1.6 Protocol Boundary D--NMVT Received
2.9.1.7 Protocol Boundary E--Send NMVT Response
2.9.1.8 Protocol Boundary F--Send MS Bulk Data
2.9.1.9 Protocol Boundary G--MS Bulk Data Received
2.9.1.10 Protocol Boundary N--MDS Error Message Information
2.9.1.11 Protocol Boundary O--Register Application Program Name
2.9.1.12 Protocol Boundary Q--MDS-MU Received
2.9.1.13 Protocol Boundary R--Send MDS-MU
2.9.1.14 Protocol Boundary S--Forward CP Status
2.9.1.15 Protocol Boundary T--Register MS Category
2.9.1.16 Protocol Boundary U--Begin Sending CP Status Information
2.9.1.17 Protocol Boundary V--Current Focal Point
2.9.1.18 Protocol Boundary BB--Cancelation From Operations Management
2.9.1.19 Protocol Boundary CC--Reporting Cancelation to Operations Management
2.9.2 **MULTIPLE_DOMAIN_SUPPORT Function Set**
2.9.2.1 Protocol Boundaries with Components Outside MULTIPLE_DOMAIN_SUPPORT
2.9.2.2 Prerequisite Function Sets
2.9.2.3 Overview of Subsets
2.9.2.3.1 End Node Role Requirements
2.9.2.3.2 Network Node Role Requirements
2.9.2.4 MULTIPLE_DOMAIN_SUPPORT Base Subset
2.9.2.4.1 Functions Provided
2.9.2.4.2 Implementation Requirements
2.9.2.4.3 Formats Supported
2.9.2.4.4 A Subset of Management Services
2.9.2.4.5 Subset of Control Point Session Services
2.9.2.4.6 Subset of Control Point Directory Services
2.9.2.4.7 Subset of Control Point and LU Presentation Services
2.9.2.4.8 MDS Router Processing
2.9.2.4.9 Overview of MDS Transaction Programs
2.9.2.4.10 MDS_SEND Processing
2.9.2.4.11 MDS_RECEIVE Processing
2.9.2.4.12 MSU_HANDLER Processing
2.9.2.4.13 MDS_SON Processing
2.9.2.5 MULTIPLE_DOMAIN_SUPPORT Optional Subset 1 (End Node Support)
2.9.2.5.1 Functions Provided
2.9.2.6 MULTIPLE_DOMAIN_SUPPORT Optional Subset 2 (Network Node Support)

2.9.2.6.1 Function Provided
2.9.2.6.2 Implementation Requirements
2.9.2.7 MULTIPLE_DOMAIN_SUPPORT Optional Subset 3 (High-Performance Option)
2.9.2.7.1 Function Provided
2.9.2.7.2 Implementation Requirements
2.9.2.8 MULTIPLE_DOMAIN_SUPPORT Optional Subset 4 (Transport Confirmation Option)
2.9.2.8.1 Function Provided
2.9.2.8.2 Implementation Requirements
2.9.2.9 Functions Common to MS Application Programs Using MDS
2.9.2.9.1 Registering with MDS
2.9.2.9.2 Common MDS-MU Parsing
2.9.2.9.3 Rejecting an MDS-MU with an Unsupported Application GDS Variable
2.9.2.9.4 Building Negative Replies for MDS-MUs
2.9.2.9.5 Canceling an Outstanding Unit of Work with an MDS Error Message
2.9.3 **MS_CAPS Function Set**
2.9.3.1 Protocol Boundaries with Components Outside MS_CAPS
2.9.3.2 Prerequisite Function Sets
2.9.3.3 Overview of Subsets
2.9.3.4 MS_CAPS Base Subset (APPN End Node Support)
2.9.3.4.1 Functions Provided
2.9.3.4.2 Formats Supported
2.9.3.4.3 Implementation Requirements
2.9.3.5 MS_CAPS Optional Subset 1 (Have a Backup or Implicit FP)
2.9.3.5.1 Functions Provided
2.9.3.5.2 Formats Supported
2.9.3.5.3 Implementation Requirements
2.9.3.6 MS_CAPS Optional Subset 2 (Be a Sphere-of-Control End Node)
2.9.3.6.1 Functions Provided
2.9.3.6.2 Formats Supported
2.9.3.6.3 Implementation Requirements
2.9.3.7 MS_CAPS Optional Subset 3 (Base Network Node Support)
2.9.3.7.1 Functions Provided
2.9.3.7.2 Formats Supported
2.9.3.7.3 Implementation Requirements
2.9.3.8 MS_CAPS Optional Subset 4 (Have a Subarea Focal Point)
2.9.3.8.1 Functions Provided
2.9.3.8.2 Formats Supported
2.9.3.8.3 Implementation Requirements
2.9.3.9 Summary of MS Capabilities Parsing
2.9.3.10 Application Program Registration with MS_CAPS
2.9.4 **FILE_SERVICES_SUPPORT Function Set**
2.9.4.1 Protocol Boundaries with Components Outside FILE_SERVICES_SUPPORT
2.9.4.2 Prerequisite Function Sets
2.9.4.3 Overview of Subsets
2.9.4.4 FILE_SERVICES_SUPPORT Base Subset
2.9.4.4.1 Functions Provided
2.9.4.4.2 Formats Supported
2.9.4.4.3 Implementation Requirements
2.9.4.5 FILE_SERVICES_SUPPORT Optional Subset 1 (Network Operator Support)
2.9.4.5.1 Functions Provided
2.9.4.5.2 Verbs Supported
2.9.4.5.3 Implementation Requirements
2.9.5 **SEND_DATA_SSCP_PU Function Set**
2.9.5.1 Protocol Boundaries with Components Outside SEND_DATA_SSCP_PU
2.9.5.2 Prerequisite Function Sets
2.9.5.3 Overview of Subsets
2.9.5.4 SEND_DATA_SSCP_PU Base Subset
2.9.5.4.1 Functions Provided
2.9.5.4.2 Formats Supported
2.9.5.4.3 Implementation Requirements
2.9.6 **RECEIVE_REQUEST_SSCP_PU Function Set**
2.9.6.1 Protocol Boundaries with Components Outside RECEIVE_REQUEST_SSCP_PU
2.9.6.2 Prerequisite Function Sets
2.9.6.3 Overview of Subsets
2.9.6.4 RECEIVE_REQUEST_SSCP_PU Base Subset
2.9.6.4.1 Functions Provided
2.9.6.4.2 Formats Supported
2.9.6.4.3 Implementation Requirements

2.10 **Chapter 10. Specialized Management Services Function Sets for Entry Points**
2.10.1 **EP_ALERT Function Set**
2.10.1.1 Protocol Boundaries with Components Outside EP_ALERT
2.10.1.2 Prerequisite Function Sets
2.10.1.3 Overview of Subsets
2.10.1.4 EP_ALERT Base Subset
2.10.1.4.1 Functions Provided
2.10.1.4.2 Formats Supported
2.10.1.4.3 Implementation Requirements
2.10.1.5 EP_ALERT Optional Subset 1 (Problem Diagnosis Data)
2.10.1.5.1 Functions Provided
2.10.1.5.2 Formats Supported
2.10.1.5.3 Implementation Requirements
2.10.1.6 EP_ALERT Optional Subset 2 (Delayed Alert)
2.10.1.6.1 Functions Provided
2.10.1.6.2 Formats Supported
2.10.1.6.3 Implementation Requirements
2.10.1.7 EP_ALERT Optional Subset 3 (Held Alert for PUMS)
2.10.1.7.1 Functions Provided
2.10.1.7.2 Formats Supported
2.10.1.7.3 Electives
2.10.1.7.4 Implementation Requirements
2.10.1.8 EP_ALERT Optional Subset 4 (Operator-Initiated Alert)
2.10.1.8.1 Functions Provided
2.10.1.8.2 Formats Supported
2.10.1.8.3 Implementation Requirements
2.10.1.9 EP_ALERT Optional Subset 5 (Qualified Message Data)
2.10.1.9.1 Functions Provided
2.10.1.9.2 Formats Supported
2.10.1.9.3 Implementation Requirements
2.10.1.10 EP_ALERT Optional Subset 6 (Text Message)

2.10.1.10.1 Functions Provided
2.10.1.10.2 Formats Supported
2.10.1.10.3 Implementation Requirements
2.10.1.11 EP_ALERT Optional Subset 7 (LAN Alert)
2.10.1.11.1 Functions Provided
2.10.1.11.2 Formats Supported
2.10.1.11.3 Implementation Requirements
2.10.1.12 EP_ALERT Optional Subset 8 (SDLC/LAN LLC Alert)
2.10.1.12.1 Functions Provided
2.10.1.12.2 Formats Supported
2.10.1.12.3 Implementation Requirements
2.10.1.13 EP_ALERT Optional Subset 9 (X.21 Alert)
2.10.1.13.1 Functions Provided
2.10.1.13.2 Formats Supported
2.10.1.13.3 Implementation Requirements
2.10.1.14 EP_ALERT Optional Subset 10 (Hybrid Alert)
2.10.1.14.1 Functions Provided
2.10.1.14.2 Formats Supported
2.10.1.14.3 Implementation Requirements
2.10.1.15 EP_ALERT Optional Subset 11 (X.25 Alert)
2.10.1.15.1 Functions Provided
2.10.1.15.2 Formats Supported
2.10.1.15.3 Implementation Requirements
2.10.1.16 EP_ALERT Optional Subset 12 (Held Alert for CPMS)
2.10.1.16.1 Functions Provided
2.10.1.16.2 Formats Supported
2.10.1.16.3 Electives
2.10.1.16.4 Implementation Requirements
2.10.1.17 EP_ALERT Optional Subset 13 (Resolution Notification Support)
2.10.1.17.1 Functions Provided
2.10.1.17.2 Formats Supported
2.10.1.17.3 Implementation Requirements
2.10.1.18 EP_ALERT Optional Subset 14 (Operations Management Support)
2.10.1.18.1 Functions Provided
2.10.1.18.2 Formats Supported
2.10.1.18.3 Implementation Requirements
2.10.1.19 Details of the Alert Encodings
2.10.1.19.1 Default/Replacement Code Points
2.10.1.19.2 Text in Alerts
2.10.1.19.3 Identification of Unique Alerts
2.10.1.19.4 Reserved Ranges of Alert Code Points
2.10.1.19.5 The Detailed Data Extended (X'85') and Detailed Data (X'82') Common Subfields
2.10.1.19.6 Product Set ID Indexing
2.10.1.19.7 Resource List Indexing
2.10.1.19.8 Associating Code Points with X'82', X'83', X'84', and X'85' Subfields
2.10.1.19.9 The Qualified Message Data (X'01') Subfield
2.10.1.19.10 Hierarchy Information in Alerts
2.10.1.19.11 Correlation in Alerts
2.10.1.19.12 Examples of Physical and Logical Identification of the Origin of an Alert Condition
2.10.2 **EP_RTM Function Set**
2.10.2.1 Protocol Boundaries with Components Outside EP_RTM
2.10.2.2 Prerequisite Function Sets
2.10.2.3 Overview of Subsets
2.10.2.4 EP_RTM Base Subset
2.10.2.4.1 Functions Provided
2.10.2.4.2 Formats Supported
2.10.2.4.3 Electives
2.10.2.4.4 Implementation Requirements
2.10.2.5 EP_RTM Optional Subset 1 (Local Display)
2.10.2.5.1 Functions Provided
2.10.2.5.2 Formats Supported
2.10.2.5.3 Implementation Requirements
2.10.3 **EP_OPI Function Set**
2.10.3.1 Protocol Boundaries with Components Outside EP_OPI
2.10.3.2 Prerequisite Function Sets
2.10.3.3 Overview of Subsets
2.10.3.4 EP_OPI Base Subset
2.10.3.4.1 Functions Provided
2.10.3.4.2 Formats Supported
2.10.3.4.3 Implementation Requirements
2.10.4 **EP_CHANGE_MGMT Function Set**
2.10.4.1 Protocol Boundaries with Components Outside EP_CHANGE_MGMT
2.10.4.2 Prerequisite Function Sets
2.10.4.3 Overview of Subsets
2.10.4.4 EP_CHANGE_MGMT Base Subset
2.10.4.4.1 Functions Provided
2.10.4.4.2 Formats Supported
2.10.4.4.3 Request Activation CP-MSU
2.10.4.4.4 Activation Acceptance CP-MSU
2.10.4.4.5 Reporting Activation CP-MSU
2.10.4.4.6 Implementation Requirements
2.10.4.5 EP_CHANGE_MGMT Optional Subset 1 (Production-Only Activation Support)
2.10.4.5.1 Functions Provided
2.10.4.5.2 Formats Supported
2.10.4.5.3 Implementation Requirements
2.10.4.6 EP_CHANGE_MGMT Optional Subset 2 (Execution Window Timing Support)
2.10.4.6.1 Functions Provided
2.10.4.6.2 Formats Supported
2.10.4.6.3 Implementation Requirements
2.10.4.7 EP_CHANGE_MGMT Optional Subset 3 (Activation Reporting Support)
2.10.4.7.1 Functions Provided
2.10.4.7.2 Formats Supported
2.10.4.7.3 Implementation Requirements
2.10.4.8 EP_CHANGE_MGMT Optional Subset 4 (Alter Active Support)
2.10.4.8.1 Functions Provided
2.10.4.8.2 Formats Supported
2.10.4.8.3 Implementation Requirements
2.10.4.9 EP_CHANGE_MGMT Optional Subset 5 (Object Disposition Support)
2.10.4.9.1 Functions Provided
2.10.4.9.2 Formats Supported

2.10.4.9.3 Implementation Requirements
2.10.4.10 EP_CHANGE_MGMT Optional Subset 6 (Initiate Command Support)
2.10.4.10.1 Functions Provided
2.10.4.10.2 Formats Supported
2.10.4.10.3 Implementation Requirements
2.10.4.11 EP_CHANGE_MGMT Optional Subset 7 (Cancel Command Support)
2.10.4.11.1 Functions Provided
2.10.4.11.2 Formats Supported
2.10.4.11.3 Request Cancelation CP-MSU
2.10.4.11.4 Cancelation CP-MSU
2.10.4.11.5 Implementation Requirements
2.10.4.12 EP_CHANGE_MGMT Optional Subset 8 (Last-Used Activation Support)
2.10.4.12.1 Functions Provided
2.10.4.12.2 Formats Supported
2.10.4.12.3 Implementation Requirements
2.10.4.13 Example Flows
2.10.4.13.1 Non-Destructive SEND_AND_INSTALL
2.10.4.13.2 Failing Non-Destructive SEND_AND_INSTALL
2.10.4.13.3 Destructive SEND_AND_INSTALL
2.10.4.13.4 Failing INSTALL
2.10.5 EP_COMMON_OPERATIONS_SERVICES Function Set
2.10.5.1 Protocol Boundaries with Components Outside EP_COMMON_OPERATIONS_SERVICES
2.10.5.2 Prerequisite Function Sets
2.10.5.3 Overview of Subsets
2.10.5.4 EP_COMMON_OPERATIONS_SERVICES Base Subset
2.10.5.4.1 Functions Provided
2.10.5.4.2 Formats Supported
2.10.5.4.3 Electives
2.10.5.4.4 Implementation Requirements
2.10.5.5 Common Operations Services Commands, Replies, and Unsolicited Traffic
2.10.5.5.1 Execute Command
2.10.5.5.2 Analyze Status
2.10.5.5.3 Query Resource Data
2.10.5.5.4 Test Resource
2.10.5.5.5 Send Message to Operator
2.10.6 EP_OPERATIONS_MGMT Function Set
2.10.6.1 Protocol Boundaries with Components Outside EP_OPERATIONS_MGMT
2.10.6.2 Prerequisite Function Sets
2.10.6.3 Overview of Subsets
2.10.6.4 EP_OPERATIONS_MGMT Base Subset
2.10.6.4.1 Functions Provided
2.10.6.4.2 Formats Supported
2.10.6.4.3 Implementation Requirements
2.10.6.5 Operations Management Replies, Reports, and Unsolicited Traffic
2.10.6.5.1 Activation Acceptance Reply and Activation Report
2.10.6.5.2 Deactivation Acceptance Reply and Deactivation Report
2.10.6.5.3 Cancelation
2.10.6.5.4 Set Clock Report
2.10.6.5.5 Operate Report
2.10.6.5.6 Routing/Parsing Report
2.10.7 Common EP_XXXX Functions
2.10.7.1 Building the Date/Time (X'01') and Relative Time (X'42') Subvectors
2.10.7.2 Building the SNA Address List (X'04') Subvector
2.10.7.3 Building the Product Set ID (X'10') Subvector
2.10.7.4 Building a Management Services Major Vector
2.10.7.5 Building an NMVT
2.10.7.6 Building an MDS-MU
2.10.7.7 Parsing of NMVTs
2.10.8 EP_XXXX Parsing of Individual Management Services Major Vectors
2.10.8.1 Subvector Formats
2.10.8.2 Parsing the Request RTM (X'8080') Major Vector
2.10.8.3 Parsing the Request Product Set ID (X'8090') Major Vector
2.10.8.4 Parsing the Change Management Major Vectors
2.10.8.5 Parsing the Operations Management Major Vectors

3.0 Part 3. Detailed Reference Material

A.0 Appendix A. Alerts Defined for Specific Environments
A.1 Alerts for Local Area Networks
A.1.1 Token-Ring LAN Alerts
A.1.1.1 Token-Ring LAN Alert 1
A.1.1.2 Token-Ring LAN Alert 2
A.1.1.3 Token-Ring LAN Alert 3
A.1.1.4 Token-Ring LAN Alert 4
A.1.1.5 Token-Ring LAN Alert 5
A.1.1.6 Token-Ring LAN Alert 6
A.1.1.7 Token-Ring LAN Alert 7
A.1.1.8 Token-Ring LAN Alert 8
A.1.1.9 Token-Ring LAN Alert 9
A.1.1.10 Token-Ring LAN Alert 10
A.1.1.11 Token-Ring LAN Alert 11
A.1.1.12 Token-Ring LAN Alert 12
A.1.1.13 Token-Ring LAN Alert 13
A.1.2 CSMA/CD LAN Alerts
A.1.2.1 CSMA/CD LAN Alert 1
A.1.2.2 CSMA/CD LAN Alert 2
A.1.2.3 CSMA/CD LAN Alert 3
A.1.2.4 CSMA/CD LAN Alert 4
A.1.2.5 CSMA/CD LAN Alert 5
A.1.2.6 CSMA/CD LAN Alert 6
A.1.2.7 CSMA/CD LAN Alert 7
A.1.2.8 CSMA/CD LAN Alert 8
A.1.2.9 CSMA/CD LAN Alert 9
A.1.2.10 CMSA/CD LAN Alert 10
A.1.2.11 CSMA/CD LAN Alert 11
A.1.3 Bridged LAN Alerts
A.1.3.1 Bridged LAN Alert 1
A.1.3.2 Bridged LAN Alert 2
A.1.3.3 Bridged LAN Alert 3
A.1.3.4 Bridged LAN Alert 4

A.1.3.5 Bridged LAN Alert 5
A.1.3.6 Bridged LAN Alert 6
A.1.3.7 Bridged LAN Alert 7
A.1.3.8 Bridged LAN Alert 8
A.1.3.9 Bridged LAN Alert 9
A.1.3.10 Bridged LAN Alert 10
A.1.3.11 Bridged LAN Alert 11
A.1.3.12 Bridged LAN Alert 12
A.1.3.13 Bridged LAN Alert 13
A.1.3.14 Bridged LAN Alert 14
A.1.3.15 Bridged LAN Alert 15
A.1.3.16 Bridged Alert 16
A.2 **SDLC/LAN LLC Alerts**
A.2.1 LAN LLC Alerts
A.2.1.1 LAN LLC Alert 1
A.2.1.2 LAN LLC Alert 2
A.2.1.3 LAN LLC Alert 3
A.2.1.4 LAN LLC Alert 4
A.2.1.5 LAN LLC Alert 5
A.2.1.6 LAN LLC Alert 6
A.2.1.7 LAN LLC Alert 7
A.2.1.8 LAN LLC Alert 8
A.2.1.9 LAN LLC Alert 9
A.2.1.10 LAN LLC Alert 10
A.2.1.11 LAN LLC Alert 11
A.2.2 SDLC Alerts
A.2.2.1 SDLC Alert 1
A.2.2.2 SDLC Alert 2
A.2.2.3 SDLC Alert 3
A.2.2.4 SDLC Alert 4
A.2.2.5 SDLC Alert 5
A.2.2.6 SDLC Alert 6
A.2.2.7 SDLC Alert 7
A.2.2.8 SDLC Alert 8
A.2.2.9 SDLC Alert 9
A.2.2.10 SDLC Alert 10
A.2.2.11 SDLC Alert 11
A.2.2.12 SDLC Alert 12
A.2.2.13 SDLC Alert 13
A.2.2.14 SDLC Alert 14
A.3 **Alerts for Switched Link Connections**
A.3.1 X.21 and X.21 Short-Hold Mode Alerts
A.3.1.1 X.21 Secondary Alert 1
A.3.1.2 X.21 Peer-to-Peer Alert 1
A.3.1.3 X.21 Secondary Alert 2
A.3.1.4 X.21 Peer-to-Peer Alert 2
A.3.1.5 X.21 Secondary Alert 3
A.3.1.6 X.21 Peer-to-Peer Alert 3
A.3.1.7 X.21 Secondary Alert 4
A.3.1.8 X.21 Peer-to-Peer Alert 4
A.3.1.9 X.21 Secondary Alert 5
A.3.1.10 X.21 Peer-to-Peer Alert 5
A.3.1.11 X.21 Secondary Alert 6
A.3.1.12 X.21 Peer-to-Peer Alert 6
A.3.1.13 X.21 Secondary Alert 7
A.3.1.14 X.21 Peer-to-Peer Alert 7
A.3.1.15 X.21 Secondary Alert 8
A.3.1.16 X.21 Peer-to-Peer Alert 8
A.3.1.17 X.21 Secondary Alert 9
A.3.1.18 X.21 Peer-to-Peer Alert 9
A.3.1.19 X.21 Secondary Alert 10
A.3.1.20 X.21 Peer-to-Peer Alert 10
A.3.1.21 X.21 Secondary Alert 11
A.3.1.22 X.21 Peer-to-Peer Alert 11
A.3.1.23 X.21 Secondary Alert 12
A.3.1.24 X.21 Peer-to-Peer Alert 12
A.3.1.25 X.21 Secondary Alert 13
A.3.1.26 X.21 Peer-to-Peer Alert 13
A.3.1.27 X.21 Secondary Alert 14
A.3.1.28 X.21 Peer-to-Peer Alert 14
A.3.1.29 X.21 Secondary Alert 15
A.3.1.30 X.21 Peer-to-Peer Alert 15
A.3.1.31 X.21 Secondary Alert 16
A.3.1.32 X.21 Peer-to-Peer Alert 16
A.3.1.33 X.21 Secondary Alert 17
A.3.1.34 X.21 Peer-to-Peer Alert 17
A.3.1.35 X.21 Secondary Alert 18
A.3.1.36 X.21 Peer-to-Peer Alert 18
A.3.1.37 X.21 Secondary Alert 19
A.3.1.38 X.21 Peer-to-Peer Alert 19
A.3.1.39 X.21 Secondary Alert 20
A.3.1.40 X.21 Peer-to-Peer Alert 20
A.3.1.41 X.21 Secondary Alert 21
A.3.1.42 X.21 Peer-to-Peer Alert 21
A.3.1.43 X.21 Secondary Alert 22
A.3.1.44 X.21 Peer-to-Peer Alert 22
A.3.1.45 X.21 Secondary Alert 23
A.3.1.46 X.21 Peer-to-Peer Alert 23
A.3.1.47 X.21 Secondary Alert 24
A.3.1.48 X.21 Peer-to-Peer Alert 24
A.3.1.49 X.21 Secondary Alert 25
A.3.1.50 X.21 Peer-to-Peer Alert 25
A.3.1.51 X.21 Secondary Alert 26
A.3.1.52 X.21 Peer-to-Peer Alert 26
A.3.1.53 X.21 Secondary Alert 27
A.3.1.54 X.21 Peer-to-Peer Alert 27
A.3.1.55 X.21 Secondary Alert 28
A.3.1.56 X.21 Peer-to-Peer Alert 28
A.3.1.57 X.21 Secondary Alert 29
A.3.1.58 X.21 Peer-to-Peer Alert 29

A.3.1.59 X.21 Secondary Alert 30
A.3.1.60 X.21 Peer-to-Peer Alert 30
A.3.1.61 X.21 Secondary Alert 31
A.3.1.62 X.21 Peer-to-Peer Alert 31
A.3.1.63 X.21 Secondary Alert 32
A.3.1.64 X.21 Peer-to-Peer Alert 32
A.3.1.65 X.21 Secondary Alert 33
A.3.1.66 X.21 Peer-to-Peer Alert 33
A.3.1.67 X.21 Secondary Alert 34
A.3.1.68 X.21 Peer-to-Peer Alert 34
A.4 **Alerts for X.25 Link Connections**
A.4.1 Packet Layer Control (PLC)
A.4.1.1 X.25 PLC Alert 1
A.4.1.2 X.25 PLC Alert 2
A.4.1.3 X.25 PLC Alert 3
A.4.1.4 X.25 PLC Alert 4
A.4.1.5 X.25 PLC Alert 5
A.4.1.6 X.25 PLC Alert 6
A.4.1.7 X.25 PLC Alert 7
A.4.1.8 X.25 PLC Alert 8
A.4.1.9 X.25 PLC Alert 9
A.4.1.10 X.25 PLC Alert 10
A.4.1.11 X.25 PLC Alert 11
A.4.2 Link Access Protocol Balanced (LAPB)
A.4.2.1 X.25 LAPB Alert 1
A.4.2.2 X.25 LAPB Alert 2
A.4.2.3 X.25 LAPB Alert 3
A.4.2.4 X.25 LAPB Alert 4
A.4.2.5 X.25 LAPB Alert 5
A.4.2.6 X.25 LAPB Alert 6
A.4.2.7 X.25 LAPB Alert 7
A.4.2.8 X.25 LAPB Alert 8
A.4.2.9 X.25 LAPB Alert 9
A.4.2.10 X.25 LAPB Alert 10
A.4.2.11 X.25 LAPB Alert 11
A.4.3 Logical Link Control (LLC)
A.4.3.1 X.25 LLC Alert 1
A.4.3.2 X.25 LLC Alert 2
A.4.3.3 X.25 LLC Alert 3
A.4.3.4 X.25 LLC Alert 4
A.4.3.5 X.25 LLC Alert 5
A.4.3.6 X.25 LLC Alert 6
A.4.3.7 X.25 LLC Alert 7
A.4.3.8 X.25 LLC Alert 8
A.4.3.9 X.25 LLC Alert 9
A.4.4 ISDN Alerts
A.4.4.1 B- and D-Channel Names in ISDN Alerts
A.4.4.2 Channel Names in the Hierarchy Name List (X'10') Subfield
A.4.4.3 Channel Names in the Associated Resources (X'11') Subfield
A.4.4.4 ISDN Basic and Primary Rate Physical Layer Alerts
A.4.4.5 ISDN Physical Layer Alert 1
A.4.4.6 ISDN Physical Layer Alert 2
A.4.4.7 ISDN Physical Layer Alert 3
A.4.4.8 ISDN Physical Layer Alert 4
A.4.4.9 ISDN Physical Layer Alert 5
A.4.4.10 ISDN Physical Layer Alert 6
A.4.4.11 ISDN Physical Layer Alert 7
A.4.4.12 ISDN Physical Layer Alert 8
A.4.4.13 ISDN Physical Layer Alert 9
A.4.4.14 ISDN Physical Layer Alert 10
A.4.4.15 ISDN Physical Layer Alert 11
A.4.4.16 ISDN Physical Layer Alert 12
A.4.4.17 ISDN Physical Layer Alert 13
A.4.4.18 ISDN R-Interface Alerts
A.4.4.19 ISDN R-interface Alert 1
A.4.4.20 ISDN Layer 2 B-Channel DLC Alerts
A.4.4.21 ISDN B-Channel DLC Alert 1
A.4.4.22 ISDN B-Channel DLC Alert 2
A.4.4.23 ISDN B-Channel DLC Alert 3
A.4.4.24 ISDN B-Channel DLC Alert 4
A.4.4.25 ISDN B-Channel DLC Alert 5
A.4.4.26 ISDN B-Channel DLC Alert 6
A.4.4.27 ISDN B-Channel DLC Alert 7
A.4.4.28 ISDN B-Channel DLC Alert 8
A.4.4.29 ISDN B-Channel DLC Alert 9
A.4.4.30 ISDN B-Channel DLC Alert 10
A.4.4.31 ISDN B-Channel DLC Alert 11
A.4.4.32 ISDN B-Channel DLC Alert 12
A.4.4.33 ISDN B-Channel DLC Alert 13
A.4.4.34 ISDN B-Channel DLC Alert 14
A.4.4.35 ISDN B-Channel DLC Alert 15
A.4.4.36 ISDN B-Channel DLC Alert 16
A.4.4.37 ISDN B-Channel DLC Alert 17
A.4.4.38 ISDN Layer 2 D-Channel LAPD Alerts
A.4.4.39 ISDN D-Channel LAPD Alert 1
A.4.4.40 ISDN D-Channel LAPD Alert 2
A.4.4.41 ISDN D-Channel LAPD Alert 3
A.4.4.42 ISDN D-Channel LAPD Alert 4
A.4.4.43 ISDN D-Channel LAPD Alert 5
A.4.4.44 ISDN D-Channel LAPD Alert 6
A.4.4.45 ISDN D-Channel LAPD Alert 7
A.4.4.46 ISDN D-Channel LAPD Alert 8
A.4.4.47 ISDN D-Channel LAPD Alert 9
A.4.4.48 ISDN D-Channel LAPD Alert 10
A.4.4.49 ISDN D-Channel LAPD Alert 11
A.4.4.50 ISDN D-Channel LAPD Alert 12
A.4.4.51 ISDN D-Channel LAPD Alert 13
A.4.4.52 ISDN D-Channel LAPD Alert 14
A.4.4.53 ISDN D-Channel LAPD Alert 15
A.4.4.54 ISDN D-Channel LAPD Alert 16

A.4.4.55 ISDN D-Channel LAP-D Alert 17
A.4.4.56 ISDN Layer 3 Q.931 Call Control Alerts
A.4.4.57 Q.931 Alert 1
A.4.4.58 Q.931 Alert 2
A.4.4.59 Q.931 Alert 3
A.4.4.60 Q.931 Alert 4
A.4.4.61 Q.931 Alert 5
A.4.4.62 Q.931 Alert 6
A.4.4.63 Q.931 Alert 7
A.4.4.64 Q.931 Alert 8
A.4.4.65 Q.931 Alert 9
A.4.4.66 Q.931 Alert 10
A.4.4.67 Q.931 Alert 11
A.4.4.68 Q.931 Alert 12
A.4.4.69 Q.931 Alert 13
A.4.4.70 Q.931 Alert 14
A.4.4.71 Q.931 Alert 15
A.4.4.72 Q.931 Alert 16
A.4.4.73 Q.931 Alert 17
A.4.4.74 Q.931 Alert 18
A.4.4.75 Q.931 Alert 19
A.4.4.76 Q.931 Alert 20
A.5 **Alerts for APPN Nodes**
A.5.1 Address Space Manager
A.5.1.1 Address Space Manager Alert CPAM001
A.5.2 Configuration Services
A.5.2.1 Configuration Services Alert CPCS001
A.5.2.2 Configuration Services Alert CPCS002
A.5.2.3 Configuration Services Alert CPCS003
A.5.2.4 Configuration Services Alert CPCS004
A.5.2.5 Configuration Services Alert CPCS005
A.5.3 Topology Database Manager
A.5.3.1 Topology and Routing Service Alert CPDB001
A.5.3.2 Topology and Routing Services Alert CPDB002
A.5.3.3 Topology and Routing Service Alert CPDB003
A.5.4 Directory Services
A.5.4.1 Directory Services Alert CPDN001
A.5.4.2 Directory Services Alert CPDN002
A.5.4.3 Directory Services Alert CPDN003
A.5.5 Session Connector Manager
A.5.5.1 Session Connector Manager Alert CPIM001
A.5.5.2 Session Connector Manager Alert CPIM002
A.5.5.3 Session Connector Manager Alert CPIM003
A.5.6 Management Services
A.5.6.1 Management Services Alert CPMS002
A.5.7 Path Control
A.5.7.1 Path Control Alert CPPL001
A.5.8 Session Services
A.5.8.1 Session Services Alert CPSS001
A.5.8.2 Session Services Alert CPSS002
A.5.8.3 Session Services Alert CPSS003
A.6 **LU6.2 Alerts**
A.6.1 LU6.2 Alert LU62001
A.6.2 LU6.2 Alert LU62002
A.6.3 LU6.2 Alert LU62003
A.6.4 LU6.2 Alert LU62004
A.6.5 LU6.2 Alert LU62005

B.0 **Appendix B. Management Services Protocol Boundary Verbs**
B.1 **Introduction**
B.2 **Verb Description Table**
B.2.1 Column Descriptions
B.2.1.1 Supplied Parameter Name
B.2.1.2 Returned Parameter Name
B.2.1.3 Parameter Reference Page (Parm Ref Page)
B.2.1.4 Length
B.2.1.5 Occurrences
B.2.1.6 Children
B.3 **Parameter Description**
B.4 **Protocol Boundary Verbs for File Services**
B.4.1 VERB: Retrieve
B.4.2 VERB: Send
B.4.3 VERB: Delete
B.4.4 VERB: Reply_To_Retrieve
B.4.5 VERB: Reply_To_Send
B.4.6 VERB: Reply_To_Delete
B.4.7 VERB: Notification_Of_Arrival
B.5 **Protocol Boundary Verbs for Change Management**
B.5.1 VERB: Send_and_Install
B.5.2 VERB: Send_and_Initiate
B.5.3 VERB: Install
B.5.4 VERB: Initiate
B.5.5 VERB: Remove
B.5.6 VERB: Accept
B.5.7 VERB: Activate
B.5.8 VERB: Reporting_Installation
B.5.9 VERB: Reporting_Initiation
B.5.10 VERB: Reporting_Removal
B.5.11 VERB: Reporting_Acceptance
B.5.12 VERB: Activation_Acceptance
B.5.13 VERB: Reporting_Activation
B.6 **Subtables**
B.6.1 SUBTABLE: Automatic_Acceptance
B.6.2 SUBTABLE: Back-Level_Change_Name_List
B.6.3 SUBTABLE: Change_Management_Activation_Use_Status
B.6.4 SUBTABLE: Corequisite_Change_Name_List
B.6.5 SUBTABLE: Deleted_Change_Name_List
B.6.6 SUBTABLE: DS_Security
B.6.7 SUBTABLE: Execution_Window_Timing
B.6.8 SUBTABLE: Force

B.6.9 SUBTABLE: Reported_Change_Name_List
B.6.10 SUBTABLE: Secondary_Installation_Change_Name_List
B.6.11 SUBTABLE: Source_Location
B.6.12 SUBTABLE: Target_List
B.6.13 SUBTABLE: Target_Location
B.6.14 SUBTABLE: Time_Stamp
B.7 **Parameter descriptions**

C.0 **Appendix C. SNA/FS File Names Defined by SNA/MS**

D.0 **Appendix D. MS Transaction Program Names**

E.0 **Appendix E. FSM Notation**
E.1 **FSM Notation**

ABBREVIATIONS **Acronyms and Abbreviations**

GLOSSARY **Glossary**

INDEX **Index**

FIGURES Figures

1-1. Model of an SNA Node (T2.0) 1.1.2.1
1-2. Model of an SNA APPN Node 1.1.2.1
1-3. Network Example 1.1.2.3.1
1-4. APPN Network Example 1.1.2.3.2
1-5. PUMS Protocol Boundaries with Other Components 1.1.2.4
1-6. CPMS (in an APPN Node) Protocol Boundaries with Other Components 1.1.2.5
1-7. CPMS to PUMS Communication on the SSCP-PU Session 1.1.3.1.1
1-8. Communication between CPMS and PUMS Using SNA/DS over LU-LU Session 1.1.3.1.2
1-9. Communication between CPMS in T2.1 Nodes 1.1.3.2.1
1-10. Communication between CPMS in Two T2.1 Nodes using SNA/DS over an LU-LU Session 1.1.3.2.2
1-11. Composition of a Role and a Function Set 1.1.4.3
2-1. Format of the NMVT Management Services RU 1.2.1.1
2-2. Format of the CP-MSU Management Services GDS Variable 1.2.1.1
2-3. Overview of a Management Services Major Vector 1.2.1.1.1
2-4. Single Reply Spanning Multiple NMVTs 1.2.1.1.1
2-5. The Unsolicited Flow for a Type 2.0 Node or Boundary-Function-Attached Type 2.1 Node 1.2.2.1
2-6. The Unsolicited Flow for an APPN Network Node or End Node 1.2.2.1
2-7. The Request without Reply Flow a Type 2.0 Node or a Boundary-Function-Attached Type 2.1 Node 1.2.2.2
2-8. The Request without Reply Flow for an APPN Network Node or End Node 1.2.2.2
2-9. The Request/Reply Flow for Non-Bulk Data for a Type 2.0 Node or a Boundary-Function-Attached Type 2.1 Node 1.2.2.3
2-10. The Request/Reply Flow for Non-Bulk Data (Multiple Resources) for a Type 2.0 Node or a Boundary-Function-Attached Type 2.1 Node 1.2.2.3
2-11. The Request/Reply Flow for Non-Bulk Data from an APPN Network Node or End Node 1.2.2.3
2-12. The Request/Reply Flow for Retrieving Bulk Data from an Entry Point 1.2.2.4
2-13. The Request/Reply Flow for Sending Bulk Data to an Entry Point 1.2.2.4
2-14. Relationship of MS to Control Point (CP) and Logical Unit (LU) Components for Multiple-Domain Support 1.2.3.2.1
2-15. Position of Multiple-Domain Support in the APPN Node 1.2.3.2.1
2-16. Structure of the Multiple-Domain Support Message Unit (MDS-MU) 1.2.3.2.2
2-17. Example of the Agent Unit-of-Work Correlator (X'1549') GDS variable 1.2.3.2.2
2-18. The MDS Error Message 1.2.3.2.2
2-19. Sessions Used by Multiple-Domain Support in an APPN Network 1.2.3.2.3
2-20. Sessions Used by Multiple-Domain Support in a Mixed Network 1.2.3.2.3
2-21. Normal Request/Reply Routing from a Network Node to an End Node 1.2.3.2.3
2-22. Normal Routing between Network Nodes for Request/Reply 1.2.3.2.3
2-23. Direct LU-LU Routing to an End Node for Request/Reply 1.2.3.2.3
2-24. Normal Network Node to End Node Routing for Request without Reply 1.2.3.2.3
2-25. Normal Network Node to Network Node Routing for Request without Reply 1.2.3.2.3
2-26. Direct Routing from End Node to a Network Node for a Request without Reply 1.2.3.2.3
2-27. Default End Node to Network Routing for MDS Request and Reply 1.2.3.2.3
2-28. Default End Node to Network Node Routing for an MDS Request without Reply 1.2.3.2.3
2-29. Default Routing of End Node to End Node Request without Reply 1.2.3.2.3
2-30. Default Routing of Request/Reply from End Node to End Node 1.2.3.2.3
2-31. Routing by MS Application names 1.2.3.2.4
2-32. SNA/DS View of Agent and Server 1.2.3.3.1
2-33. Structure of the SNA/DS Message Unit Used by SNA/MS 1.2.3.3.3
2-34. Example Flow: User Issues Request 1.2.3.3.4
2-35. Example Flow: SNA/MS Builds Agent Object and Issues

SEND_DISTRIBUTION 1.2.3.3.4
2-36. Example Flow: SNA/DS Builds a Message Unit and Sends It to the Entry Point 1.2.3.3.4
2-37. Example Flow: SNA/DS at the Entry Point Receives the MU and Invokes the SNA/FS Server 1.2.3.3.4
2-38. Example Flow: SNA/MS at the Entry Point is Returned RECEIVE_DISTRIBUTION Parameters 1.2.3.3.4
2-39. Focal Point Notification from APPN Network Node to Client End Nodes 1.2.4.1.1
2-40. MS Capabilities Exchanges for Explicit SOC 1.2.4.1.1
2-41. MS Capabilities Exchanges for Implicit SOC 1.2.4.1.1
2-42. MS Capabilities Exchanges for Default SOC 1.2.4.1.1
2-43. Switch to Backup FP, Then Recovery of Primary 1.2.4.1.2
2-44. Initialization of an Explicit Sphere of Control 1.2.4.2.1
2-45. A SOC Node Requests to be Included in the Implicit Sphere of Control 1.2.4.2.1
2-46. Focal Point Notification From a Network Node to an End Node 1.2.4.2.1
2-47. Switching to a Backup Focal Point 1.2.4.2.3
3-1. Overview of Flow Between Problem Management Elements 1.3
3-2. Overview of Problem-Determination Steps 1.3.1
3-3. Overview of Problem Diagnosis Steps 1.3.2
3-4. Overview of Problem Resolution Steps 1.3.3
3-5. Minimum Function Supplied by the Alert and Resolution Notification 1.3.4.3
3-6. Maximum Function Supplied by the Alert and Resolution Notification 1.3.4.3
3-7. Example Flow Showing an Alert 1.3.4.5
3-8. Example Flow Showing Alert Originating in an APPN Node 1.3.4.5
3-9. Example Flow Showing Delayed Alert 1.3.4.5
3-10. Example Flow Showing a Resolution Notification 1.3.4.5
3-11. Example Flow Showing a Resolution Notification Originating in an APPN Node 1.3.4.5
3-12. Example Flow Showing Alerts and Resolution Notifications Originating in an APPN Node 1.3.4.5
3-13. Boundary Function Attachment Over an X.21 Network in a Hierarchical Environment 1.3.6.1.1
3-14. Nodes Communicating Over an X.21 Network in a Peer Environment 1.3.6.1.1
3-15. Peer-to-Peer SDLC Link Between Type 2.1 Nodes 1.3.6.1.2
3-16. Nodes Communicating Over a Token-Ring LAN 1.3.6.1.3
3-17. Nodes Communicating Over a CSMA/CD LAN 1.3.6.1.3
3-18. Nodes Communicating Over a Bridged Token-Ring LAN 1.3.6.1.3
3-19. Logical Links Traversing a Token-Ring LAN 1.3.6.1.3
3-20. Boundary Function Attachment Over an X.25 Network 1.3.6.1.4
4-1. Overview and Placement of Response-Time Monitoring Steps 1.4.1
4-2. Example Flow Showing Request to Set RTM Parameters for a Single LU 1.4.1.3.1
4-3. Example Flow Showing Request to Set RTM Parameters for All LUs 1.4.1.3.1
4-4. Example Flow Showing Request for RTM Data and Status for a Single LU, and the Resulting Reply 1.4.1.3.2
4-5. Example Flow Showing Request for RTM Data and Status for All LUs, and the Resulting Replies 1.4.1.3.2
4-6. Example Flow Showing Request for RTM Data and Status for All LUs, When No LUs Have Accumulated Data 1.4.1.3.2
4-7. Example Flow Showing Unsolicited Return of RTM Data and Status 1.4.1.3.3
5-1. Overview and Placement of Query Product Identification (QPI) Steps 1.5.1
5-2. Example Flow Showing QPI Request to a PU and the Subsequent Replies 1.5.1.3
6-1. Overview of Flow Between Change Management Elements 1.6
6-2. Overview of Change Planning Steps 1.6.1
6-3. Overview of Change Control Steps 1.6.2
6-4. The Request/Reply Flow for Retrieving Change Files 1.6.2.7
6-5. The Request/Reply Flow for Send-and-Install 1.6.2.7
6-6. The Request/Reply Flow for Remove 1.6.2.7
6-7. The Request/Reply Flow for Activate 1.6.3.3
6-8. Centralized Customization of Microcode 1.6.4.1
6-9. Centralized Distribution and Control of Application Software 1.6.4.2
7-1. Parameter Major Vectors in Common Operations Services Encodings 1.7.2.1.3
7-2. Example Flow Showing Common Operations Services Request and Reply 1.7.2.4
7-3. Example Flow Showing Common Operations Services Request and Reply for an APPN Node 1.7.2.4
7-4. Example Flow Showing Common Operations Services Send Message to Operator 1.7.2.4
7-5. Example Flow Showing Common Operations Services Send Message to Operator for an APPN Node 1.7.2.4
7-6. Time Windows: Execute Not After Time and Time Interval 1.7.3.1
7-7. Time Windows: Execute Not Before Time and Time Interval 1.7.3.1
7-8. Time Windows: Execute Not Before/Execute Not After/Time Interval 1.7.3.1
7-9. Example Activation Scenario 1.7.3.3.1
7-10. Second-level Routing Using the Routing and Targeting Instructions GDS variable 1.7.3.3.1
7-11. The Request/Reply Flow for Non-Bulk Data from an APPN Network Node or End Node 1.7.3.3.1
7-12. The Unsolicited Flow for an APPN End Node 1.7.3.3.1
8-1. Function Sets for the PUMS Type 2.0 Node Role 2.8.1.1
8-2. Function Sets for the CPMS Role for APPN End Nodes or Network Nodes 2.8.1.2
9-1. Conventions Used In Describing MS Function Set Groups 2.9.1.1
9-2. MULTIPLE_DOMAIN_SUPPORT Function Set Group 2.9.2
9-3. Base and Optional Subsets of MULTIPLE_DOMAIN_SUPPORT Function Set 2.9.2.3
9-4. Subset of Management Services and its Service Transaction Programs

Implementing MDS 2.9.2.4.4
9-5. MDS External Protocol Boundaries 2.9.2.4.4
9-6. Model used to describe MDS_ROUTER 2.9.2.4.8
9-7. Processing Register Application Program Name 2.9.2.4.8
9-8. Processing Send MDS-MU 2.9.2.4.8
9-9. Processing Send Current Focal Point 2.9.2.4.8
9-10. Processing Begin Sending CP Status Information 2.9.2.4.8
9-11. Processing CP_STATUS Signal 2.9.2.4.8
9-12. Processing MDS Routing Unit (Inbound) 2.9.2.4.8
9-13. Processing SNASVCMG SON 2.9.2.4.8
9-14. Processing MDS Routing Unit (REPLY TO OUTBOUND) 2.9.2.4.8
9-15. MDS Routing Algorithm (Part 1 of 2) 2.9.2.4.8
9-16. MDS Routing Algorithm (Part 2 of 2) 2.9.2.4.8
9-17. Processing of Local Application Routing List (Part 1 of 3) 2.9.2.4.8
9-18. Processing of Local Application Routing List (Part 2 of 3) 2.9.2.4.8
9-19. Processing of Local Application Routing List (Part 3 of 3) 2.9.2.4.8
9-20. Processing of Active Transaction List (Part 1 of 6) 2.9.2.4.8
9-21. Processing of Active Transaction List (Part 2 of 6) 2.9.2.4.8
9-22. Processing of Active Transaction List (Part 3 of 6) 2.9.2.4.8
9-23. Processing of Active Transaction List (Part 4 of 6) 2.9.2.4.8
9-24. Processing of Active Transaction List (Part 5 of 6) 2.9.2.4.8
9-25. Processing of Active Transaction List (Part 6 of 6) 2.9.2.4.8
9-26. Processing of CPSVCMG Session List (Part 1 of 5) 2.9.2.4.8
9-27. Processing of CPSVCMG Session List (Part 2 of 5) 2.9.2.4.8
9-28. Processing of CPSVCMG Session List (Part 3 of 5) 2.9.2.4.8
9-29. Processing of CPSVCMG Session List (Part 4 of 5) 2.9.2.4.8
9-30. Processing of CPSVCMG Session List (Part 5 of 5) 2.9.2.4.8
9-31. Processing of SNASVCMG Session List (Part 1 of 4) 2.9.2.4.8
9-32. Processing of SNASVCMG Session List (Part 2 of 4) 2.9.2.4.8
9-33. Processing of SNASVCMG Session List (Part 3 of 4) 2.9.2.4.8
9-34. Processing of SNASVCMG Session List (Part 4 of 4) 2.9.2.4.8
9-35. MDS Header Parsing 2.9.2.4.8
9-36. Building MDS Error Messages and Alerts 2.9.2.4.8
9-37. MDS_SEND Manager (Part 1 of 2) 2.9.2.4.8
9-38. MDS_SEND Manager (Part 2 of 2) 2.9.2.4.8
9-39. Back-Level Node Support (Part 1 of 4) 2.9.2.4.8
9-40. Back-Level Node Support (Part 2 of 4) 2.9.2.4.8
9-41. Back-Level Node Support (Part 3 of 4) 2.9.2.4.8
9-42. Back-level Node Support (Part 4 of 4) 2.9.2.4.8
9-43. Sources of MDS_ROUTER Error Notification 2.9.2.4.8
9-44. Mulitiple-Domain Support Transaction Programs 2.9.2.4.9
9-45. Verb Sequence: Single MDS-MU on a Conversation 2.9.2.4.9
9-46. Verb Sequence: Multiple MDS-MUs on a Single Conversation 2.9.2.4.9
9-47. Verb Sequence: MDS_RECEIVE TP Failure (Part 1 of 2) 2.9.2.4.9
9-48. Verb Sequence: MDS_RECEIVE TP Failure (Part 2 of 2) 2.9.2.4.9
9-49. Verb Sequence: Session Outage (Part 1 of 2) 2.9.2.4.9
9-50. Verb Sequence: Session Outage (Part 2 of 2) 2.9.2.4.9
9-51. Verb Sequence: Initial Contact With Back-level Node 2.9.2.4.9
9-52. Verb Sequence: Null Message From MDS Router 2.9.2.4.9
9-53. Verb Sequence: MDS SEND Unable To Allocate Conversation 2.9.2.4.9
9-54. MDS_SEND Processing (Part 1 of 4) 2.9.2.4.10
9-55. MDS_SEND Processing (Part 2 of 4) 2.9.2.4.10
9-56. MDS_SEND Processing (Part 3 of 4) 2.9.2.4.10
9-57. MDS_SEND Processing (Part 4 of 4) 2.9.2.4.10
9-58. MDS_RECEIVE Processing (Part 1 of 2) 2.9.2.4.11
9-59. MDS_RECEIVE Processing (Part 2 of 2) 2.9.2.4.11
9-60. MSU_HANDLER Processing (Part 1 of 2) 2.9.2.4.12
9-61. MSU_HANDLER Processing (Part 2 of 2) 2.9.2.4.12
9-62. Processing for MDS_SON 2.9.2.4.13
9-63. Directory Services Support (Part 1 of 2) 2.9.2.6.2
9-64. Directory Services Support (Part 2 of 2) 2.9.2.6.2
9-65. MDS Router Chooses Outbound Session Based on Local MDS Information 2.9.2.6.2
9-66. MDS Router Chooses Outbound Session Based on Directory Services Local Information 2.9.2.6.2
9-67. MDS Router Chooses Outbound Session Based on Directory Services Network Search 2.9.2.6.2
9-68. Error Case: Destination Resource Not Known to CP.DS 2.9.2.6.2
9-69. Error Case: CP.DS Provides Incorrect Directory Information from Cache (Part 1 of 2) 2.9.2.6.2
9-70. Error Case: CP.DS Provides Incorrect Directory Information from Cache (Part 2 of 2) 2.9.2.6.2
9-71. Error Case: Routing Failure after the First Hop. 2.9.2.6.2
9-72. Service Transaction Programs for MULTIPLE_DOMAIN_SUPPORT Optional Subset 1 (High-Performance Option) 2.9.2.7.2
9-73. Verb Sequence: Sending MDS-MUs 2.9.2.7.2
9-74. Verb Sequence: MDS_HP_RECEIVE TP Failure 2.9.2.7.2
9-75. Verb Sequence: Session Outage (Part 1 of 2) 2.9.2.7.2
9-76. Verb Sequence: Session Outage (Part 2 of 2) 2.9.2.7.2
9-77. Verb Sequence: MDS_HP_SEND Unable To Allocate Conversation 2.9.2.7.2
9-78. Normal Sequence: Establishment of Parallel Conversations Between Two Nodes 2.9.2.7.2
9-79. Error Scenario: Failure Affecting only one of the Sessions 2.9.2.7.2
9-80. Error Scenario: Failure Affecting both Contention-winner and Contention-loser Sessions 2.9.2.7.2
9-81. Error Scenario: Failure of MDS_HP_RECEIVE_TP 2.9.2.7.2
9-82. MDS_HP_SEND Processing (Part 1 of 4) 2.9.2.7.2
9-83. MDS_HP_SEND Processing (Part 2 of 4) 2.9.2.7.2
9-84. MDS_HP_SEND Processing (part 3 of 4) 2.9.2.7.2
9-85. MDS_HP_SEND Processing (Part 4 of 4) 2.9.2.7.2
9-86. MDS_HP_RECEIVE Processing (Part 1 of 2) 2.9.2.7.2
9-87. MDS_HP_RECEIVE Processing (Part 2 of 2) 2.9.2.7.2
9-88. MS_CAPS Function Set 2.9.3

9-89. Base and Optional Subsets of MS_CAPS Function Set 2.9.3.3
9-90. MS Capabilities Flow Supported by MS_CAPS Base Subset 2.9.3.4.2
9-91. MS Capabilities Flows Supported by MS_CAPS Optional Subset 1 2.9.3.5.2
9-92. MS Capabilities Flows Supported by MS_CAPS Optional Subset 2 2.9.3.6.2
9-93. MS Capabilities Flows Supported by MS_CAPS Optional Subset 3 2.9.3.7.2
9-94. MS Capabilities Flow Supported by MS_CAPS Optional Subset 4 2.9.3.8.2
9-95. FILE_SERVICES_SUPPORT Function Set Group 2.9.4
9-96. Base and Optional Subsets of FILE_SERVICES_SUPPORT Function Set 2.9.4.3
9-97. SEND_DATA_SSCP_PU Function Set Group 2.9.5
9-98. Base and Optional Subsets of SEND_DATA_SSCP_PU Function Set 2.9.5.3
9-99. RECEIVE_REQUEST_SSCP_PU Function Set Group 2.9.6
9-100. Base and Optional Subsets of RECEIVE_REQUEST_SSCP_PU Function Set 2.9.6.3
10-1. EP_ALERT Function Set Group 2.10.1
10-2. Base and Optional Subsets of EP_ALERT Function Set 2.10.1.3
10-3. Delayed Alert Control Block 2.10.1.6.3
10-4. Held Alert Control Block 2.10.1.7.4
10-5. Overview of Delayed and Held Alert Processing 2.10.1.7.4
10-6. Overview of Delayed Alert Processing in a Control Point 2.10.1.16.4
10-7. Held Alert Processing with Multiple-Domain Support 2.10.1.16.4
10-8. Format for Default/Replacement Alert Code Points 2.10.1.19.1
10-9. The Unique Alert Identifier 2.10.1.19.3
10-10. Two-Stage Procedure for Generating an Alert ID Number 2.10.1.19.3
10-11. Example of the Detailed Data Extended (X'85') Subfield and the Corresponding Unit of Display 2.10.1.19.5
10-12. Example Encoding of the User Causes (X'94') Subvector 2.10.1.19.5
10-13. Possible Unit of Display for the Specified X'94' Subvector 2.10.1.19.5
10-14. Examples Illustrating How X'11' Subvectors are Indexed 2.10.1.19.6
10-15. Structure of the Subfields within a 'Causes' (X'94'-X'96') Subvector 2.10.1.19.8
10-16. Structure of 'Empty' and 'Full' X'85' Subfields 2.10.1.19.8
10-17. Relationship Between Qualified Message Data and Detailed Data Subfields 2.10.1.19.9
10-18. Product Set ID (X'10') Subvectors 2.10.1.19.12
10-19. Product Identifier (X'11') Subvectors within a X'10' Subvector 2.10.1.19.12
10-20. Product Identifier (X'11') Subvector 2.10.1.19.12
10-21. Illustration of Example 1 2.10.1.19.12
10-22. Illustration of Example 2 2.10.1.19.12
10-23. Illustration of Example 3 2.10.1.19.12
10-24. Illustration of Example 4 2.10.1.19.12
10-25. Illustration of Example 5 2.10.1.19.12
10-26. EP_RTM Function Set Group 2.10.2
10-27. Base and Optional Subsets of EP_RTM Function Set 2.10.2.3
10-28. EP_RTM's Request/Reply Correlation Control Block 2.10.2.4.4
10-29. EP_OPI Function Set Group 2.10.3
10-30. Base and Optional Subsets of EP_OPI Function Set 2.10.3.3
10-31. EP_CHANGE_MGMT Function Set Group 2.10.4
10-32. Base and Optional Subsets of EP_CHANGE_MGMT Function Set 2.10.4.3
10-33. EP_CHANGE_MGMT Processing State Transitions and Output Codes 2.10.4.4.6
10-34. Entry Point Processing for Install 2.10.4.4.6
10-35. Entry Point Processing for Remove 2.10.4.4.6
10-36. Entry Point Processing for Accept 2.10.4.4.6
10-37. Sample configuration 2.10.4.13.1
10-38. Non-Destructive Send and Install (Part 1 of 6) 2.10.4.13.1
10-39. Non-Destructive Send and Install (Part 2 of 6) 2.10.4.13.1
10-40. Non-Destructive Send and Install (Part 3 of 6) 2.10.4.13.1
10-41. Non-Destructive Send and Install (Part 4 of 6) 2.10.4.13.1
10-42. Non-Destructive Send and Install (Part 5 of 6) 2.10.4.13.1
10-43. Non-Destructive Send and Install (Part 6 of 6) 2.10.4.13.1
10-44. Sample Configuration 2.10.4.13.2
10-45. Failing Non-Destructive Send and Install (Part 1 of 6) 2.10.4.13.2
10-46. Failing Non-Destructive Send and Install (Part 2 of 6) 2.10.4.13.2
10-47. Failing Non-Destructive Send and Install (Part 3 of 6) 2.10.4.13.2
10-48. Failing Non-Destructive Send and Install (Part 4 of 6) 2.10.4.13.2
10-49. Failing Non-Destructive Send and Install (Part 5 of 6) 2.10.4.13.2
10-50. Failing Non-Destructive Send and Install (Part 6 of 6) 2.10.4.13.2
10-51. Sample Configuration 2.10.4.13.3
10-52. Destructive Send and Install (Part 1 of 6) 2.10.4.13.3
10-53. Destructive Send and Install (Part 2 of 6) 2.10.4.13.3
10-54. Destructive Send and Install (Part 3 of 6) 2.10.4.13.3
10-55. Destructive Send and Install (Part 4 of 6) 2.10.4.13.3
10-56. Destructive Send and Install (Part 5 of 6) 2.10.4.13.3
10-57. Destructive Send and Install (Part 6 of 6) 2.10.4.13.3
10-58. Sample Configuration 2.10.4.13.4
10-59. Failing Install (Part 1 of 6) 2.10.4.13.4
10-60. Failing Install (Part 2 of 6) 2.10.4.13.4
10-61. Failing Install (Part 3 of 6) 2.10.4.13.4
10-62. Failing Install (Part 4 of 6) 2.10.4.13.4
10-63. Failing Install (Part 5 of 6) 2.10.4.13.4
10-64. Failing Install (Part 6 of 6) 2.10.4.13.4

10-65. EP_COMMON_OPERATIONS_SERVICES Function Set Group 2.10.5
10-66. Base and Optional Subsets of EP_COMMON_OPERATIONS_SERVICES Function Set 2.10.5.3
10-67. EP_OPERATIONS_MGMT Function Set Group 2.10.6
10-68. Base and Optional Subsets of EP_OPERATIONS_MGMT Function Set 2.10.6.3
10-69. Unformatted, Formatted, and Partially Formatted Subvectors 2.10.8.1
E-1. Syntax of an FSM State-Transition Matrix E.1

TABLES Tables

1-1. The Major Categories of Network Management 1.1.1
1-2. The Elements of Problem Management 1.1.1.1
1-3. The Elements of Performance and Accounting Management 1.1.1.2
2-1. Summary of Formats Used by Management Services 1.2.1.1
2-2. Choice of Transport by MS Function Sets for Unsolicited Data 1.2.3
2-3. Settings of MDS Message Type, First and Last MDS Message Flags 1.2.3.2.2
2-4. Contents of Agent Unit-of-Work Correlator (X'1549') GDS variable 1.2.3.2.2
2-5. Contents of Sequence Number Date Time (SEQNO DTM) Structure (part of Agent Unit-of-Work Correlator (X'1549') GDS variable) 1.2.3.2.2
2-6. Contents of an MDS Error Message Created by MDS_ROUTER 1.2.3.2.2
2-7. Contents of an MDS Error Message generated by an MS application 1.2.3.2.2
2-8. Base SNA/FS Commands and Instructions by Role 1.2.3.3.4
6-1. Functions and the Roles to Which They Apply 1.6.2.5.11
7-1. Example Context-Identified Values (X'1323') GDS variable 1.7.3.1
7-2. An Example SNACR Structure 1.7.3.1
7-3. The Request Activation CP-MSU 1.7.3.1.2
7-4. Example Request Activation CP-MSU 1.7.3.1.2
7-5. The Activation Acceptance CP-MSU 1.7.3.1.3
7-6. Example Activation Acceptance CP-MSU 1.7.3.1.3
7-7. The Activation CP-MSU 1.7.3.1.4
7-8. Example Activation CP-MSU 1.7.3.1.4
7-9. The Request Deactivation CP-MSU 1.7.3.1.5
7-10. Example Request Deactivation CP-MSU 1.7.3.1.5
7-11. The Deactivation Acceptance CP-MSU 1.7.3.1.6
7-12. Example Deactivation Acceptance CP-MSU 1.7.3.1.6
7-13. The Deactivation CP-MSU 1.7.3.1.7
7-14. Example Deactivation CP-MSU 1.7.3.1.7
7-15. The Set Clock CP-MSU 1.7.3.1.8
7-16. Example Set Clock CP-MSU 1.7.3.1.8
7-17. The Set Clock Report CP-MSU 1.7.3.1.9
7-18. Example Set Clock Report CP-MSU (Acceptance) 1.7.3.1.9
7-19. Example Set Clock Report CP-MSU (Success) 1.7.3.1.9
7-20. The Request Cancelation CP-MSU 1.7.3.1.10
7-21. Example Request Cancelation CP-MSU 1.7.3.1.10
7-22. The Cancelation CP-MSU 1.7.3.1.11
7-23. Example Cancelation CP-MSU (Acceptance) 1.7.3.1.11
7-24. Example Cancelation CP-MSU (Success) 1.7.3.1.11
7-25. The Operate CP-MSU 1.7.3.1.12
7-26. Example Operate CP-MSU 1.7.3.1.12
7-27. The Operate Report CP-MSU 1.7.3.1.13
7-28. Example Operate Report CP-MSU (Acceptance) 1.7.3.1.13
7-29. Example Operate Report CP-MSU (Success) 1.7.3.1.13
7-30. The Routing/Parsing Report CP-MSU 1.7.3.1.14
7-31. Example Routing/Parsing Report CP-MSU 1.7.3.1.14
7-32. Example MULTIPLE_DOMAIN_SUPPORT Header and Application Correlator Control Bit Settings 1.7.3.3.1
9-1. Table of General and Specialized Function Sets for the PUMS Role 2.9.1.1
9-2. Table of General and Specialized Function Sets for the CPMS Role (APPN End Node) 2.9.1.1
9-3. MDS_ROUTER Control Blocks 2.9.2.4.8
9-4. MDS_ROUTER High Level Processes 2.9.2.4.8
9-5. MDS_ROUTER Specialized Processes 2.9.2.4.8
9-6. Contents of Local Application Program Routing List 2.9.2.4.8
9-7. Contents of Active Transaction List 2.9.2.4.8
9-8. Contents of CPSVCMG Session List 2.9.2.4.8
9-9. Contents of SNASVCMG Session List 2.9.2.4.8
9-10. Required Format Checks for MDS-MUs 2.9.2.4.8
9-11. Contents of an MDS Error Message created by MDS_ROUTER 2.9.2.4.8
9-12. MDS Routing Exceptions 2.9.2.4.8
9-13. MDS Transaction Exceptions 2.9.2.4.8
9-14. Multiple Domain Support LU 6.2 Transaction Programs 2.9.2.4.9
9-15. LU 6.2 Transaction Programs for MULTIPLE_DOMAIN_SUPPORT Optional Subset 1 (High-Performance Option) 2.9.2.7.2
9-16. Sense Data Returned by an MS Application After Receipt of an MDS-MU 2.9.2.9.2
9-17. Error Report Formats used by MS Application Programs 2.9.2.9.4
9-18. Sense Data Returned by MS Application (transaction cancelation) 2.9.2.9.5
9-19. MS Capabilities Table for End Nodes 2.9.3.4.3
9-20. Description of MS Capabilities Table Information 2.9.3.4.3
9-21. Contents of MS Capabilities Request (X'E1') - Focal Point Notification 2.9.3.4.3
9-22. Contents of MS Capabilities Request (X'63') - Implicit 2.9.3.5.3
9-23. Contents of MS Capabilities Request (X'63') - Backup 2.9.3.5.3
9-24. Contents of MS Capabilities Reply (X'64') - Reject 2.9.3.5.3
9-25. Contents of MS Capabilities Reply (X'64') - Accept 2.9.3.5.3
9-26. MS Capabilities Table for End Nodes Supporting Backup and Implicit

Focal Points 2.9.3.5.3
9-27. Description of MS Capabilities Table Information 2.9.3.5.3
9-28. MS Capabilities Table for SOC End Nodes 2.9.3.6.3
9-29. Description of MS Capabilities Table Information 2.9.3.6.3
9-30. Contents of MS Capabilities Request (X'61') - Primary Focal Point 2.9.3.6.3
9-31. Contents of MS Capabilities Request (X'61') - Primary Focal Point (Back-level) 2.9.3.6.3
9-32. Contents of MS Capabilities Reply (X'62') - Entry Point Accepts Request 2.9.3.6.3
9-33. Contents of MS Capabilities Reply (X'62') - Entry Point Rejects Request 2.9.3.6.3
9-34. Contents of MS Capabilities Reply (X'62') - Revoke 2.9.3.6.3
9-35. Contents of MS Capabilities Request (X'61') - Default Focal Point 2.9.3.7.3
9-36. Contents of MS Capabilities Request (X'61') - Secondary Focal Point (Back-level) 2.9.3.7.3
9-37. MS Capabilities Table for Network Nodes 2.9.3.7.3
9-38. Description of MS Capabilities Table Information 2.9.3.7.3
9-39. Sense Data Returned by MS_Capabilities Parser 2.9.3.9
9-40. Choosing Agent Object Contents and Server Parameters from the Command Received 2.9.4.4.3
9-41. Choosing Command and Server Instructions from the Verb 2.9.4.5.3
10-1. LMSs That Initiate Alerts 2.10.1.4.3
10-2. Setting of the Held and Delayed Alert Indicators by Different Alert Senders 2.10.1.7.4
10-3. Default and Replacement Text for Recommended Action Code Points 2.10.1.19.1
10-4. One-byte Structure for Indexing a Product Identifier (X'11') Subvector 2.10.1.19.6
10-5. Subvector Conditions of Presence in the Request RTM (X'8080') Major Vector 2.10.2.4.4
10-6. Example Illustrating Use of EP_RTM's NMVT Buffer 2.10.2.4.4
10-7. Subvector Conditions of Presence in the RTM (X'0080') Major Vector 2.10.2.4.4
10-8. The Request Activation CP-MSU 2.10.4.4.3
10-9. The Activation Acceptance CP-MSU 2.10.4.4.4
10-10. The Reporting Activation CP-MSU 2.10.4.4.5
10-11. Choosing Major Vector/Subvector and Server Parameters from the Verb 2.10.4.4.6
10-12. Reporting Subvectors Used Based on State of Primary Change Files 2.10.4.4.6
10-13. Choosing SNA/FS Server Parameters in the Report Direction 2.10.4.4.6
10-14. Choosing Major Vector/GDS variable and Server Parameters from the Verb 2.10.4.10.3
10-15. Choosing SNA/FS Server Parameters in the Report Direction 2.10.4.10.3
10-16. The Request Cancelation CP-MSU 2.10.4.11.3
10-17. The Cancelation CP-MSU 2.10.4.11.4
10-18. Send_and_Install Supplied Parameters 2.10.4.13.1
10-19. Send_Distribution Supplied Parameters 2.10.4.13.1
10-20. Receive_Distribution Returned Parameters 2.10.4.13.1
10-21. Send_Distribution Supplied Parameters 2.10.4.13.1
10-22. Receive_Distribution Returned Parameters 2.10.4.13.1
10-23. Reporting_Installation Returned Parameters 2.10.4.13.1
10-24. Send_and_Install Supplied Parameters 2.10.4.13.2
10-25. Send_Distribution Supplied Parameters 2.10.4.13.2
10-26. Receive_Distribution Returned Parameters 2.10.4.13.2
10-27. Send_Distribution Supplied Parameters 2.10.4.13.2
10-28. Receive_Distribution Returned Parameters 2.10.4.13.2
10-29. Reply_to_Send Returned Parameters 2.10.4.13.2
10-30. Send_and_Install Supplied Parameters 2.10.4.13.3
10-31. Send_Distribution Supplied Parameters 2.10.4.13.3
10-32. Receive_Distribution Returned Parameters 2.10.4.13.3
10-33. Send_Distribution Supplied Parameters 2.10.4.13.3
10-34. Receive_Distribution Returned Parameters 2.10.4.13.3
10-35. Reporting_Installation Returned Parameters 2.10.4.13.3
10-36. Install Supplied Parameters 2.10.4.13.4
10-37. Send_Distribution Supplied Parameters 2.10.4.13.4
10-38. Receive_Distribution Returned Parameters 2.10.4.13.4
10-39. Send_Distribution Supplied Parameters 2.10.4.13.4
10-40. Receive_Distribution Returned Parameters 2.10.4.13.4
10-41. Reporting_Installation Returned Parameters 2.10.4.13.4
10-42. Management Services Major Vectors Supported by EP_COMMON_OPERATIONS_SERVICES 2.10.5.4.2
10-43. MS Application Protocol Boundary A-1 2.10.5.4.4
10-44. MS Application Protocol Boundary A-2 2.10.5.4.4
10-45. MS Application Protocol Boundary A-3 2.10.6.4.3
10-46. MS Application Protocol Boundary A-4 2.10.6.4.3
10-47. Sense Data Returned by PUMS After Receipt of an NMVT Request 2.10.7.7
10-48. Sense Data Returned by PUMS after Receipt of a Request RTM (X'8080') Major Vector 2.10.8.2
10-49. Sense Data Returned by PUMS After Receipt of a Request Product Set ID (X'8090') Major Vector 2.10.8.3
10-50. Sense Data Returned by MS After Receipt of a Request Change Control (X'8050') Major Vector 2.10.8.4
10-51. Sense Data Returned by MS After Receipt of a Request Activation (X'8066') Major Vector 2.10.8.4
10-52. Sense Data Returned by MS After Receipt of a Request Cancelation (X'8076') Major Vector 2.10.8.4
10-53. Sense Data Returned by PUMS after Receipt of a Common Operations Services Major Vector 2.10.8.4
A-1. Token-Ring Alert Sending Products A.1.1
A-2. CSMA/CD LAN Alert Sending Products A.1.2
C-1. Identification Tokens for Microcode Objects Using Change Management Structured Names C.0
C-2. Identification Tokens for Microcode Customizing Data Objects Using Change Management Structured Names C.0

C-3. Identification Tokens for All Objects Using Enterprise Structured Names C.0
C-4. Identification Tokens of the Architecturally Defined Subtree for Group Objects C.0
C-5. Identification Tokens of the Architecturally Defined Subtree for C.0
C-6. Identification Tokens of the Architecturally Defined Subtree for C.0
C-7. Identification Tokens of the Architecturally Defined Subtree for C.0
D-1. Architecturally-Defined Management Services Transaction Programs D.0

LEGEND Notices

Any reference to an IBM licensed program in this document is not intended to state or imply that only IBM's program may be used.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not of itself constitute or imply a grant of any license or immunity under any patents, patent applications, trademarks, copyrights, or other similar rights of IBM or of any third party, or any right to refer to IBM in any advertising or other promotional or marketing activities. IBM assumes no responsibility for any infringement of patents or other rights of third parties that may result from use of the subject matter disclosed in this document or for the manufacture, use, lease, or sale of machines or programs described herein, outside of any responsibilities assumed via the IBM Customer Agreement.

This publication may include references to microcode. Some IBM products contain microcode classified as Licensed Internal Code. Licensed Internal Code is provided under terms and conditions set forth in IBM agreements, such as the IBM Customer Agreement.

Licenses under IBM's utility patents are available on reasonable and nondiscriminatory terms and conditions. IBM does not grant licenses under its appearance design patents. Inquiries relative to licensing should be directed in writing to the IBM Director of Commercial Relations, International Business Machines Corporation, Armonk, New York, 10504.

It is possible that this material may contain reference to, or information about IBM products (machines and programs), programming, or services that are not announced in your country. Such references or information must not be construed to mean that IBM intends to announce such products, programs, or services in your country.

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

| ° Advanced Peer-to-Peer Networking
| ° APPN
° Systems Application Architecture
° SystemView

| This book is not intended for production use and is furnished as is. IBM
| assumes no responsibility for the use of the functions as described in
| this book in any production manner.

1.0 Part 1. Introduction to Management Services

Subtopics:

- 1.1 Chapter 1. Introduction to Management Services
- 1.2 Chapter 2. Management Services Concepts
- 1.3 Chapter 3. Problem Management
- 1.4 Chapter 4. Performance and Accounting Management
- 1.5 Chapter 5. Configuration Management
- 1.6 Chapter 6. Change Management
- 1.7 Chapter 7. Operations Management

1.1 Chapter 1. Introduction to Management Services

Subtopics:

- 1.1.1 Network Management and Its Major Categories
- 1.1.2 Introduction to the Management Services Components of the Node
- 1.1.3 Overview of Management Services Communication
- 1.1.4 Implementation Choices

1.1.1 Network Management and Its Major Categories

Network management is the process of planning, organizing, monitoring, and controlling a communication-oriented data processing or information system. The architecture provided to assist in network management of SNA systems is called *management services* and is implemented as a set of functions and services designed to capture and use the information needed for effective management.

This section provides an overall discussion of network management, including those processes for which management services are not provided at present, those that are currently performed manually, and those that are implemented in components outside the SNA node. This general discussion will help the reader understand the use of management services described later in this book.

Network management is divided into the major categories listed in Table 1-1.

Table 1-1. The Major Categories of Network Management
Problem Management
Performance and Accounting Management
Configuration Management
Change Management
Operations Management

Network management processes may be distributed across different nodes, and may require several iterations of data collection and analysis as new events occur. They may be automated system processes or manual processes carried out by network operations or vendor service personnel. Each of the major categories of network management makes use of several types of common services, i.e., services that are implemented once, and then used by each individual category:

- Security management: The controlling of safeguards established to protect hardware, software, and data from accidental or malicious modification, destruction, or disclosure.

- Focal point services: The maintenance of focal point information needed for the establishment, backup, and recovery of focal-point to entry-point relationships.

The individual major categories of network management are described in the following sections.

Subtopics:

- 1.1.1.1 Problem Management
- 1.1.1.2 Performance and Accounting Management
- 1.1.1.3 Configuration Management
- 1.1.1.4 Change Management
- 1.1.1.5 Operations Management
- 1.1.1.6 Focal Point Services

1.1.1.1 Problem Management

Problem management is the process of managing a problem from its detection through its final resolution.

Problem is used to describe an error condition resulting in a loss of availability of a system resource to an end user. Problems may originate in hardware, software (operating systems and application programs), microcode, (1) media, or because of external causes such as user procedures or environmental abnormalities.

Problem management includes the elements listed in Table 1-2.

Table 1-2. The Elements of Problem Management
Problem Determination
Problem Diagnosis
Problem Bypass and Recovery
Problem Resolution
Problem Tracking and Control

SNA management services are provided to assist in performing problem determination, problem diagnosis, and problem resolution.

Problem determination is the detection of the loss or impending loss of availability of a system resource to an end user, and completion of the steps necessary for problem diagnosis to begin. It is the process of isolating a problem to the failing hardware device, software product, microcode component, medium, or external cause, to identify the organization responsible for problem diagnosis.

Problem diagnosis is the process of determining the precise cause of a hardware, software, microcode, medium, or externally-caused problem and the precise action required to resolve the problem.

If problem diagnosis is carried out manually, it begins at the end of problem determination. Problem determination output is used to determine the system resource responsible for the problem and the general area where problem diagnosis should begin. If diagnostic data was gathered along with problem determination data, it will be used. If additional data is needed for problem determination or problem diagnosis, it must be gathered and analyzed manually or with the help of SNA management services. In cases of a complex problem, this process may be iterated several times, each time gaining more data and eliminating more components from the set of possible causes.

If problem diagnosis is carried out automatically by a system process, it is usually done in parallel with problem determination, such that the outcome of both processes can be reported together.

Problem bypass and recovery is the process of implementing partial or complete circumvention of a problem, usually before the final resolution of the original incident. This is normally temporary in nature, although it may be permanent. In a simple case, such as printer failure, the system may bypass the reported incident by directing printer output to an alternate printer. In a more complex case, such as controller failure, typically a spare controller must be acquired and placed into service or the existing controller must be repaired, thus substituting problem resolution for bypass and recovery. Problem bypass and recovery frequently takes place in parallel with other elements of problem management; e.g., output may be routed to an alternate printer while the problem with the primary printer is still being diagnosed.

Problem resolution is the process of taking action to correct the error condition detected as a problem or impending problem. This action starts when problem diagnosis is complete and often requires scheduling a repair action, carrying out the repair action, testing the repair, and subsequently reporting the problem as closed and the resource back in service. This procedure is normally the case when a permanent hardware failure occurs. Problem resolution may be simple; for instance, for an inadvertent power-off condition, problem resolution is to turn on the power.

Problem tracking and control is the process of tracking problems until their final resolution. A problem management record is created in the problem data base anytime external intervention is required to restore the system to its proper state of operation. The problem management record provides a repository for all data about a problem to allow correlation with other activities and failures related to the same problem. Some types of this data are problem resolution, status monitoring, and problem status reports.

---

(1) Microcode may be classified as IBM Licensed Internal Code. See "Notices" at the beginning of this document for more information.

1.1.1.2 Performance and Accounting Management

Performance and accounting management is the process of quantifying, measuring, reporting, and controlling the responsiveness, availability, utilization, and usage charges of a network component.

As users become more dependent on networks and network application programs, the attainment of acceptable and consistent performance objectives becomes more critical. A poorly performing (or erratically performing) system becomes, in effect, an unavailable system from the end-user perspective. Therefore, awareness of this condition by the party responsible for the management of the information system is required.

Many installation managers establish service agreements or objectives with end users. Typically, these agreements include some response time or availability criteria. Performance management data is required to determine if these agreements or objectives have been met.

Performance and accounting management includes the elements listed in Table 1-3.

Table 1-3. The Elements of Performance and Accounting Management
Response-Time Monitoring
Availability Monitoring
Utilization Monitoring
Component Delay Monitoring
Performance Tuning
Performance Tracking and Control
Accounting

SNA management services are provided to assist in performing response-time monitoring.

Response-time monitoring is the monitoring of end-user response times and the starting of problem determination if service levels are exceeded.

Availability monitoring is the monitoring of availability and the providing of appropriate data for accounting management.

Utilization monitoring is the monitoring of the utilization of network resources and the starting of problem determination if service levels are exceeded.

Component delay monitoring is the monitoring of the delay incurred at critical components and the starting of problem determination if service levels are exceeded.

Performance tuning is the process of taking action to improve performance. Data available from performance tracking and control is used to identify areas where performance tuning is required.

Performance tracking and control is the tracking and reporting of performance status and the controlling of the effects of tuning actions.

Accounting is the recording and tracking of usage charges at a system resource level. The goal of the accounting function is to provide data to permit proper distribution of the resource cost among the users according to the portion of the system used by each user. The types of accounting data to be collected include connect time, processor cycles used, data quantity transmitted, and class of service for each user session.

1.1.1.3 Configuration Management

Configuration management is the control of information necessary to identify both physical and logical information system resources and their relationship to one another. The information may include resource names, addresses, location, contacts and phone numbers, vendor or organizations responsible for service, product identification information, and other items. Configuration management assures that this information is updated whenever changes are made and always reflects the current configuration.

Configuration management aids the user in managing the inventory of information system components and assists the other management services categories as follows:

- Problem determination may use the data to determine resource physical identity, location, and possibly the organization responsible for service.

- Change management may use this data to analyze the effect of changes and to schedule changes. Refer to the discussion of change management for the relationship of configuration management and change management.

The following SNA management services assist in configuration management.

Query product identification is the process of retrieving physical identification information (on both hardware and software) from a specified node. The information retrieved is sometimes referred to as *vital product data*.

1.1.1.4 Change Management

Change management is the planning, control, and application program of additions, deletions, and modifications to the information system hardware, microcode and software components. Included are the following:

- Changes to software such as its installation or removal. Software includes system software or application program (user) software. Changes can be in the form of fixes, complete load module replacements, or customizing data.

- Changes to microcode, such as its installation or removal. A microcode change could be a patch, a fix, an engineering change (EC), a feature change or customizing data.

- Changes to hardware such as its installation or removal, the application program of engineering changes or miscellaneous equipment specification (MES) updates.

The following functions are provided by change management:

- Controlling changes -- sending, retrieving, installing, removing, and accepting change files at remote nodes

- Node activation

Change management and configuration management are closely related. Configuration management focuses on the existing configuration of hardware, software, and microcode at any given time. Change management focuses on the planning, application program, and tracking of changes to the existing configuration.

The change management and operations management function sets are also related in that both may be used to activate system resources. Although the parameters specified in the activation requests may be different, the parameters are chosen from the same major vector format. Another relationship between these function sets concerns the cancelation request. As with activation requests, each function set may tailor cancelation requests using various optional parameters. Again, the parameters are chosen from the same major vector format.

Changes occur for two reasons: (1) because user requirements have changed, e.g., for new or additional hardware or a new application program system; (2) because a problem requires bypassing or correction. For example, the latter is the case when a failing component is to be removed or replaced to correct a hardware failure, or when an application program module is to be modified to correct a detected program error.

Problems and changes interact with each other in the following manner. While problem bypass or resolution is one of the causes of change, change, in turn, is one of the causes of problems. Problems may be caused by the installation of programs, modifications, or permanent or temporary fixes that were not completely debugged; or by the addition of hardware or software that causes the existing system to perform in a different manner, resulting in a problem that had not been experienced before. Changes that cause problems cannot be completely eliminated, but change management attempts to bring them under control, handle them in an orderly manner, and track them so that problem management can be aware of change activity, if required.

1.1.1.5 Operations Management

The Operations Management category provides the means to query and control

distributed network resources from a central site. The two sub-categories that accomplish this are the Common Operations Services function set and the Operations Management function set. Depending on solution requirements, either (or both) of these function sets provide for skill centralization, network automation, and thus a reduced manual workload for network operators. The two functions sets are described below.

Subtopics:

- 1.1.1.5.1 Common Operations Services
- 1.1.1.5.2 Operations Management

1.1.1.5.1 Common Operations Services

Common operations services are a set of services that cross all of the major categories of network management. The architecture for common operations services provides a way to manage types of resources not explicitly addressed by the architecture defined for the individual categories. It does this by providing a general mechanism that allows a network operator to communicate with specialized network management application programs; these application programs, in turn, provide functions not currently provided by SNA management services.

As an aid in problem management a number of common operation services are provided. They are used in conjunction with any of the management services categories, but are especially useful for problem determination.

- The **Execute Command** service provides a transmission envelope for any character-coded message or command that is to be executed at a destination node. It provides a means of invoking remotely what would otherwise be a local management services function.

- **Resource Management** services are provided to transport information in an architecturally-defined structure without constraining the content of the information. The Query Resource Data and Test Resource services collect data and associated identifiers about one or more system or network elements. The Test Resource service is the more powerful since it requests active testing and returns summary data as well as the detailed data with labels. The Analyze Status service performs the same type of function but requests an architecturally-defined response rather than resource-dependent information.

1.1.1.5.2 Operations Management

Operations management provides the capability to control distributed network resources. Activating and deactivating resources, as well as setting network resource clocks are all functions that are included in this category. In addition, a cancelation function has been defined that enables previously sent commands (including those executing at the target) to be terminated.

As "Change Management" in topic 1.1.1.4 states, operations management and change management are related in that both make use of the encodings for the activate and cancel requests.

Finally, the Operate command (and corresponding report) provides a means to envelope product-specific commands and data within an SNA transport structure, extending the reach of the network operator. Multiple types of enveloping are supported, depending on the implementation requirements and the structure of the enveloped command itself.

As an implementation option, operations management commands may be initiated as a result of system problem notification forwarding, as described in the problem management architecture.

1.1.1.6 Focal Point Services

Focal point services are available to all the categories of management services. These services provide for the establishment of focal point-entry point relationships, the maintenance of those relationships, and error recovery for those relationships. This topic is introduced in "Focal Point Concepts" in topic 1.2.4.

1.1.2 Introduction to the Management Services Components of the Node

In an SNA network, the end point of a link or the junction of two or more links is referred to as a *node*. Nodes can be host processors, distributed processors, communication controllers, or workstations. Nodes can vary in routing and other functional capabilities depending on their role in the network. Physically, nodes include the hardware and software components of workstations, controllers, and processors.

Subtopics:

- 1.1.2.1 SNA Node Types
- 1.1.2.2 Management Services Roles
- 1.1.2.3 SNA Networks
- 1.1.2.4 Introduction to Physical Unit Management Services (PUMS)
- 1.1.2.5 Introduction to Control Point Management Services (CPMS)

1.1.2.1 SNA Node Types

The role requirements for management services components are categorized according to the type of node in which the component resides. Since the terminology for identifying the various node types within SNA has varied in the past, this introductory section will state the meanings of the terms used in the sections that follow. Refer to Chapter 8, "Role Requirements of Management Services Components" in topic 2.8 for additional information about each node type.

° **Type 2.0 node:** (2)

An SNA node that attaches to a subarea boundary function and receives its management services support via an SSCP-PU session. Also referred to as a *peripheral node*.

° **Type 2.1 node:**

- **boundary-function-attached type 2.1 node:**

  A type 2.1 node that is attached to the subarea boundary function, and receives its management services support via an SSCP-PU session in a manner identical to a type 2.0 node. Since the management services roles for the two node types are identical, this manual will generally not distinguish between the two.

- **Type 2.1 end node:**

  A type 2.1 node that provides full SNA end-user services, but no intermediate routing or network services to any other node; it is configured only as an endpoint in a network.

  -- **Low-entry networking (LEN) end node:**

  A type 2.1 end node that does not support sessions between its local control point (CP) and another CP.

  -- **Advanced Peer-to-Peer Networking(*) (APPN(*)) end node:**

  A type 2.1 end node that supports sessions between its local control point (CP) and the CP in an adjacent network node, to dynamically register its resources with the adjacent CP (its network services server), to send and receive directory search requests, and to obtain management services. Optionally, it may also attach to a subarea network as a peripheral node, presenting the appearance of a *boundary-function-attached type 2.1 node*.

- **Advanced Peer-to-Peer Networking (APPN) network node:**

  A type 2.1 node that, besides offering full SNA end-user services, provides intermediate routing services within a network of type 2.1 nodes, and network services to its local LUs and attached type 2.1 end nodes in its domain; optionally, it may also attach to a subarea network as a peripheral node, presenting the appearance of a *boundary-function-attached type 2.1 node*.

° **Subarea node:**

A node that uses network addresses for routing and whose routing tables are therefore affected by changes in the configuration of the

network. Subarea nodes can provide gateway function, subarea transport facilities, and boundary function support for peripheral nodes.

- **Type 4 node:**

  An SNA node that provides subarea path control networking with other type 4 nodes and type 5 nodes, and a boundary function for attachment of type 2.0 and type 2.1 nodes to the subarea network. It receives its management services support via an SSCP-PU session.

- **Type 5 node:**

  An SNA node that provides subarea path control networking with other type 4 nodes and type 5 nodes, and a boundary function for attachment of type 2.0 and type 2.1 nodes to the subarea network. A type 5 node, unlike a type 4 node, provides an SSCP.

  A **system services control point (SSCP)** is a control point within a subarea network for managing the configuration, coordinating network operator requests, and providing directory support and other session services for end users of the network.

| ° **Type 4/5 APPN and LEN nodes:**

| A type 4/5 node complex can also provide an APPN or LEN appearance to
| an attached T2.1 APPN or LEN node, using fully compatible protocols.

This manual describes management services for type 2.0 nodes,
| boundary-function-attached type 2.1 nodes, type 2.1 end nodes, and APPN
| network nodes. Management services for type 4 and type 5 nodes are not
described.

For boundary-function-attached type 2.1 nodes, the management services role is identical to that for a type 2.0 node, so this manual will generally not distinguish between the two node types.

```
          _____________________________________                          ___________
         |  physical  |  physical resources  |                         |    end    |
         |  resources |  (e.g., disks,       |                         |    user   |
         |  manager   |  tempes, storage)    |                         |           |
         |____________________________________|                         |___________|
              "                                                              "
              |____________________|                                          |
          _°°°°°°°°°°°°°°°°°°°°°°°|°°°°°°°°°°°°°°°°°°°
```

```
|           °°°°°°°°°°°°°°°°°°°°|°°°°°°°°°°°°°°°°°°°
```

(4) While type 4 and type 5 nodes are entry points, the specific management services support for them is not described in this manual.

1.1.2.3 SNA Networks

SNA management services provides operators, whether programmed or human, with the facilities by which an SNA node or a network of nodes may be managed. These include the facilities to manage problems with system resources and to manage system changes, system performance, and system configuration. This section introduces the management services components of a node.

Subtopics:

- 1.1.2.3.1 Single-Domain Subarea Networks
- 1.1.2.3.2 Multiple-Domain APPN Networks

1.1.2.3.1 Single-Domain Subarea Networks

```
  ..............................................
  .  network control center                    .
  .                 _________________          .
  .                | network operator |        .
  .                |__________________|        .
  .                       |                    .
  .                       |        ______      .
  .      ____________________    __| disk |    .
  .      | A                |_|  |______|      .
  .      |    processor 1   |     ______       .
  .      |          (SSCP)  |____| tape |      .
  .      |_ _________ ___ ___|    |______|      .
  .  ______|_______   |   |__________          .
  . | display (LU) |  |       _______|_____     .       ........................................
  . |______________|  |      | printer (LU) |  .       .                                      .
  .                   |      |______________|  .       .                                 _____ .
  .      ____________|_______                  .       .    ____________________    __| disk |  .
  .      | B   communication  |                .       .   | D                |_|  |_____|  .
  .      |     controller 1   |________________/_______|     processor 2    |     _____    .
  .      |               (PU) |                .       .   |               (PU) |____| tape |  .
  .      |___________ ________|                .       .   |___ ___________ __|    |_____|  .
  .                  |                         .       .       |           |____             .
  .                  |          location 1     .       .  _____|_______    _____|_________   .
  ...................|........................        . | printer (LU) |  | workstation (LU)|  .
                    /                                  . |______________|  |_________________|  .
          ............|............                   .                     location 3     .
          .    _______|_____      .                   ........................................
          .   | C             |    .
          .   |   printer     |    .
          .   |    (PU) (LU)  |    .
          .   |_______________|    .
          .          location 2    .
          ........................
```

Figure 1-3. Network Example

Figure 1-3 is an example SNA subarea network with one domain. In this example, the resources are interconnected with different transmission facilities and media, e.g., channel adapters, local coaxial cable, and links. The nodes are labeled with generic hardware terminology, with the SNA network addressable unit shown in parentheses; a letter is shown in the upper left corner for reference purposes. The term *end user* is used to describe the ultimate source or destination of application program data flowing through an SNA network. An end user may be an application program or a person. End users of the network have access to its functions through *logical units* (LUs). LUs allow communication with other end users by establishing sessions between them (i.e., LU-LU session). End users located at displays or workstations have access to network resources at all of the locations.

Each physical location contains at least one SNA node. The *physical unit* (PU) in the node is responsible for managing and monitoring the resources associated with the node. The PU is an *entry point*. The PU is also responsible for managing links when its node contains the primary link station. For example, the PU at node B (communication controller 1) is responsible for managing the SNA resources associated with its node (e.g., links as well as logical resources such as sessions and routes) and all of the non-SNA resources (e.g., communication controller storage). The PU at node D (processor 2) is responsible for managing the SNA resources associated with its node (e.g., physical resources such as printers and workstations as well as logical resources such as logical units associated with these printers and workstations) and all of the non-SNA resources (e.g., disk, tape, processor storage).

Node A contains a *system services control point* (SSCP). The role of an SSCP is broader than that of a PU: whereas the PU manages the resources

associated with the node in which it resides, the SSCP manages a set of PUs. The SSCP is fulfilling the role of a *focal point* for the entry points (PUs) in its domain. The term *domain* describes an SSCP and the physical units (PUs), logical units (LUs), links, link stations, and all the associated resources that the SSCP has the ability to control by means of SNA configuration services activation requests and deactivation requests. The SSCP is also responsible for managing the configuration, processing unsolicited data received from physical units, coordinating and routing network operator requests and physical unit replies, and for providing directory support and other session services for end users of the network.

The SSCP may interact with PUs, and a *network operator*, whose job is to manage the network.

Network operators issue commands to, and receive responses from, SSCPs. Many repetitive, supervisory functions use predetermined sequences of commands and responses that can be coded into a program, a technique sometimes described as *automated network operations*. The term *network operator* therefore refers also to any program that manages network operation.

1.1.2.3.2 Multiple-Domain APPN Networks

```
                                                   ooooooooooooooooooooooooooooooooooooooooooooooo
                                                   o  . . . . . . . . . . . . . . . . . . . . . . o
                                                   o  .                 _______________        .  o
                                                   o  .                | F             |       .  o
                                                   o  .          _____|   processor   |       .  o
                                                   o  .         |     | (LU) (EN CP)  |       .  o
                                                   o  .         |     |___ _________|        .  o
                                                   o  .         |     |          |      location 3 .  o
                                                   o  .         | . . . . . . | . . . . . . . .  o
                                                   o  . . . . . | . . . . . . | . . . . . . . .  o
                                                   o  .         |             |      location 4 .  o
                                                   o  .    _____|_____        |                .  o
                                                   o  .   | G           |    |                .  o
                                                   o  .   |   processor   |    |                .  o
                                                   o  .   | (LU) (EN CP)  |    |                .  o
                                                   o  .   |_____ _________|    |                .  o
                                                   o  .         |             |                .  o
                                                   o  .         |     _________|                .  o
                                                   o  .    _____|___|____      _____________  .  o
                                                   o  .   | H           |     | I           |  .  o
                                                   o  .   |   processor   |____| workstation |  .  o
                                                   o  .   |    (NN CP)    |    |   (EN CP)   |  .  o
                                                   o  .   |_______________|    |_____________|  .  o
                                                   o  . . . . . . . . . . . . . . . . . . . . . . o
                                                   o  NN H domain                                  o
                                                   ooooooooooooooooooooooooooooooooooooooooooooooo
          _________________
         | network operator |
         |__________________|
                   |
oooooooooooooooooooo|oooooooooooooooooooooooooooo
o . . . . . . . . . |. . . . . . . . . . . . . . o
o .     ___________________      network      . o
o .    | A   (focal point) |    control      . o
o .    |     processor     |    center       . o
o .    |      (NN CP)      |                 . o
o .    |___________ ___ __|                 . o
o .                |   |                     . o
o .                |   |        location 1 . o
o . . . . . . . . . | . | . . . . . . . . . . o
o NN A domain      |   |____________________________
oooooooooooooooooooo|oooooooooooooooooooooooooooo
                   |   ____________________________
oooooooooooooooooooo|ooo|oooooooooooooooooooooooo
o . . . . . . . . . | . | . . . . . . . . . . . . o
o .            _____|___|_____   location 2 . o
o .           | C             |              . o
o .           |   processor   |              . o
o .      _____|    (NN CP)    |____          . o
o .     |     |_______________|    |         . o
o .     |                          |         . o
o .  ___|__________      __________|____     . o
o .  | D            |    | E            |    . o
o .  |   processor  |____|   processor  |    . o
o .  |    (EN CP)   |    |    (EN CP)   |    . o
o .  |______________|    |______________|    . o
o .                                          . o
o . . . . . . . . . . . . . . . . . . . . . . o
o NN C domain                                 o
oooooooooooooooooooooooooooooooooooooooooooooo
```

Figure 1-4. APPN Network Example

Figure 1-4 is an example of an APPN network. Nodes A, C, and H are APPN network nodes; nodes D, E, F, G, and I are APPN end nodes.

The domain of an APPN end node control point is the CP, its LUs, links, link stations, and all resources that it activates and deactivates. An APPN end node acts only as an entry point for itself and the resources in its domain.

The domain of an APPN network node control point is the CP, its LUs, links, link stations, and all resources that it activates and deactivates, and the end node CPs and LUs for which it provides network services. When an APPN network node establishes a relationship with a focal point, it notifies all the end nodes in its domain of the focal point's identity. This allows the end nodes to report to the same focal point without having to participate directly in the exchanges of *MS Capabilities* messages associated with the establishment of a *sphere of control*. These focal point concepts are introduced in greater detail later.

Management information may be routed to a focal point from several domains, including the focal point's own domain. The set of domains for which a focal point is responsible is termed that focal point's *sphere of control*.

Information concerning resources in domains other than the focal point's is routed through those domains' CPs. A focal point receives unsolicited data from the domains in its sphere of control for the particular management services category for which the focal point is responsible. A focal point can send requests to and receive replies from any control points in the APPN network.

The technique of locating management services support at one or more focal points by distributing management services requests and data throughout a network is known as *distributed management services*.

In the example, the NN CP in node A is the focal point for certain categories of management services for the entire APPN network. The NN CPs in nodes C and H are in A's sphere of control, while the EN CPs are in the same sphere of control as their NN server CPs. Although this is not shown in the illustration, an end node implementation may *optionally* provide the capability to be directly in a focal point's sphere of control rather than simply reporting to the same focal point as its network node server.

Network owners can define multiple focal points in their networks. These focal points can have the responsibility for the same category of network management data or for different categories of network management data. This flexibility is obtained by allowing focal points to have defined spheres of control. The spheres of control for multiple focal points for the same category of management services may *not* be overlapping (i.e., the same domain may *not* be in the sphere of control of multiple focal points for the same category of management services). The spheres of control for focal points for different management services categories need not be the same.

Focal points may communicate with one another to allow for multiple levels of expertise by nesting focal points. (For example, a focal point may forward management services data it receives from a control point in its sphere of control to its focal point.)

1.1.2.4 Introduction to Physical Unit Management Services (PUMS)

Physical unit management services (PUMS) is the component of the PU responsible for providing general management services for the node and its associated resources. PUMS communicates with its controlling SSCP using management services RUs transported on an SSCP-PU session and over LU-LU sessions used by SNA/Distribution Services. In a boundary-function-attached type 2.1 node, the CP acts as a PU for the purposes of management services. In general, no distinction is made in this document between type 2.1 nodes in this role and type 2.0 nodes.

The services performed by PUMS are summarized as follows:

- Receiving management services request RUs from a control point with which it has an SSCP-PU session, converting the RUs into requests that are typically implementation-unique, and routing these requests to the appropriate component

- Building and sending RUs

  - Receiving unsolicited management data from other components associated with its node, building an RU, and sending the RU to the resource's controlling SSCPs

  - Receiving solicited management data from other components associated with its node, building an RU, and sending the RU to the control point that requested the data.

  In building the RU, PUMS may reformat implementation-unique data, add other data such as product identification, and assist in correlating requests with replies.

Figure 1-5 provides a model of the PU and illustrates the components with which PUMS has protocol boundaries. Each protocol boundary is numbered in the figure for subsequent ease of reference. Protocol boundaries exist between PUMS and the following components (numbered to correspond to the numbers in Figure 1-5). In those cases in which the protocol boundaries are the same, the numbers used are the same as those used to specify the protocol boundaries between CPMS and the components with which it interacts. (See Figure 1-6 in topic 1.1.2.5).

```
 ____________________________________________________________________________
|                                                                            |
| º º º º º º º º º º º º º º º º º º º º º º º º º º                              |
| º                                 ___________   º          ________________|__  |
| º  ________________             |           |  º (2)     |   ____          |
| º | session        |    (5)     |           |____________| LMS |      LU   |  |  |
| º | manager        |__________ÿ|           |  º         |  |_____|          |  |  |
```

```
 | ° |_______________|           |           |  °         |                   |  |  |
 | °                             |           |  °         |  _________________ |  |  |
 | ° | configuration  |    (6)    |           |  ° (7)     | |SNA/Distribution| |  |  |
 | ° | services       |_________ÿ|           |___________ÿ|Services        | |  |  |
 | ° |_______________|           |           |  °         | |_________________| |  |  |
 | °                             |  P U M S  |  °         |_____________________|  |  |
 | ° | SSCP-PU        |    (4)    |           |  °           |_____________________|  |
 | ° | half-session   |_________ÿ|           |  °                                    |
 | ° |_______________|           |           |  °                                    |
 | °                             |           |  °                                    |
 | °                             |           |  °          _________________         |
 | °                             |           |  ° (3)     | data link control  |__    |
 | °                             |           |_________ÿ| manager            |  |  |
 | °                             |           |  °         |_____________________|  |  |
 | °                             |___________|  °           |_____________________|  |
 | ° Physical Unit                  "    "     °                                    |
 | ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° | ° °|° ° ° °                                    |
 |   SNA node                     |    |                                            |
 |________________________________|____|____________________________________________|
 _________________                |    |                                  ___________
 | physical    _____  |           (1) |    | (8)                 |   _____  program   |
 | resources | LMS |_________________|    |_______________________ÿ| LMS | supervisor|
 | manager   |_____|  |                                          |  |_____|          |
 |___________________|                                          |____________________|
```

Figure 1-5. PUMS Protocol Boundaries with Other Components

(1) Physical Resources Manager Local Management Services (LMS)

- The physical resources manager LMS provides unsolicited notification of problems with the node's physical resources, e.g., tapes, disks, storage, microcode.

(2) Logical Unit Local Management Services (LMS)

- Upon request, the logical unit LMS sets response-time measurement parameters and provides response-time data.

- The logical unit LMS provides unsolicited notification of problems within the LU and unsolicited response-time data.

(3) Data Link Control Manager Local Management Services

- The DLC manager LMS provides unsolicited notification of problems with links.

(4) SSCP-PU Half-Session

- The half-session provides communication (over SSCP-PU sessions) with a resource's controlling CPMS.

(5) PU Session Manager

- Upon request, the PU session manager provides information about the currently active sessions managed by the PU.

(6) PU Configuration Services

- Upon request, PU configuration services provides information that uniquely identifies the hardware and software of the node and provides a list of active logical units.

- PU configuration services provides unsolicited notification when the SSCP-PU session becomes active.

(7) SNA/Distribution Services (SNA/DS)

- SNA/Distribution Services provides:

  - The capability to send and receive CP-MSUs, SNA/File Services (SNA/FS) agent objects, and SNA/FS files (bulk data) over LU-LU sessions.

  The change management category uses SNA/File Services and SNA/Distribution Services for distribution of potentially large files, requests to manipulate them, and reports to track the distribution and installation.

(8) Program Supervisor Local Management Services (LMS)

- Upon request, the program supervisor alters software and microcode

components.

1.1.2.5 Introduction to Control Point Management Services (CPMS)

Control point management services (CPMS) is a component of the control point that assists a network operator in management of the control point and its domain.

The services performed by CPMS in an APPN node are summarized as follows:

- *multiple-domain support*, or MDS. This transport provides for routing of data between management services application programs via CP-CP and LU-LU sessions. See "Transport of Non-Bulk Management Services Data in APPN Networks" in topic 1.2.3.2 for a discussion of communication between control points via MDS.

- Receiving management services requests from a focal point via MDS. Typically, these focal point requests are converted into implementation-unique formats and are routed to the appropriate component.

- Receiving unsolicited management data from other components associated with its node, building the appropriate SNA formats, and sending this data to the appropriate focal point.

CPMS in a LEN end node has the same network management capabilities as PUMS (see "Introduction to Physical Unit Management Services (PUMS)" in topic 1.1.2.4); its network management capabilities are currently limited to its role as a peripheral node attached to a subarea network.

```
 _____________________________________________________________________________
| ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° |
| °                                 __________    ° (22)    | node operator     |     |
| °                                |          |__________ÿ| facility          |     |
| °   ________________             |          |   °        |___________________|     |
| °   | session      |  (11)      |          |   °        ___________________     |
| °   | services     |_________ÿ|          |  ° (2)     |  ____          |__   |
| °   |______________|            |          |____________| LMS |      LU  |  |  |
| °   ________________             |          |   °        |  |____|          |  |  |
| °   | configuration |  (12)     |          |   °        |                  |  |  |
| °   | services     |_________ÿ|          |   °        |  _____________   |  |  |
| °   |______________|            |          |  ° (7)     | |SNA/Distribution| |  |  |
| °   ________________             |          |___________ÿ|Services        | |  |  |
| °   | address space |  (15)     |          |   °        | |_______________| |  |  |
| °   | manager      |_________ÿ|  C P M S |   °        |___________________|  |  |
| °   |______________|            |          |   °          |___________________|  |
|| °   ________________             |          |   °        ___________________     | "
|| °   | topology and |  (19)     |          |  ° (18)    | session connector |     | | NN
|| °   | routing services |_____ÿ|          |___________| manager           |     | | only
|| °   |______________|            |          |   °        |___________________|     |
| °   ________________             |          |   °        ___________________     |
| °   | directory    |  (20)     |          |  ° (9)     | path control      |__   |
| °   | services     |_________ÿ|          |___________| manager           |  |  |
| °   |______________|            |          |   °        |___________________|  |  |
| °   ________________             |          |   °          |___________________|  |
| °  _| presentation |  (21)     |          |   °        ___________________     |
| ° | | services     |_________ÿ|          |  ° (3)     | data link control |__   |
| ° | |______________|            |          |___________| manager           |  |  |
| ° |_________________|            |          |   °        |___________________|  |  |
| °                                |          |   °          |___________________|  |
| °   Control Point                |__________|   °                                |
| ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° " ° " ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° |
|     SNA node                          |   |                                     |
|_______________________________________|___|_____________________________________|
    ________________                  |   |                   ___________________
   | physical   _____ |            (1) |   | (8)              |   _____  program  |
   | resources | LMS |_______________|   |__________________ÿ| LMS | supervisor|
   | manager   |_____|  |                                     |  |____|           |
   |__________________|                                      |___________________|
```

Figure 1-6. CPMS (in an APPN Node) Protocol Boundaries with Other Components

Figure 1-6 illustrates the components with which CPMS in an APPN node exchanges data. The definition for the exchange of this data within a single node is referred to as a *protocol boundary*. Each protocol boundary is numbered in the figure for subsequent ease of reference. The same numbers will be used in the function set group diagrams in later chapters to refer to these protocol boundaries. The numbers appearing in Figure 1-6 are not necessarily consecutive, since some of the numbers are assigned in Figure 1-5 in topic 1.1.2.4. PUMS in a type 2.0 node has protocol boundaries with other components in the node which are described in "Introduction to Physical Unit Management Services (PUMS)" in topic 1.1.2.4.

(1) Physical Resources Manager Local Management Services (LMS)

- The physical resources manager LMS provides unsolicited notification of problems with the node's physical resources, e.g., tapes, disks, storage, microcode.

(2) Logical Unit Local Management Services (LU LMS)

- The logical unit LMS provides unsolicited notification of problems detected by the LU.

(3) Data Link Control Manager

- The DLC manager LMS provides unsolicited error and traffic statistics when problems associated with links and link stations are encountered, or when a counter threshold is exceeded.

(7) SNA/Distribution Services (SNA/DS)

- SNA/Distribution Services provides:

  - The capability to send and receive CP-MSUs, SNA/File Services (SNA/FS) agent objects, and SNA/FS files (bulk data) over an LU-LU session using LU 6.2 protocols.

The change management category uses SNA/File Services and SNA/Distribution Services for distribution of potentially large files, and of the commands to manipulate them.

(8) Program Supervisor Local Management Services (LMS)

- Upon request, the program supervisor alters software and microcode components. In addition, requests to cancel processing of requests sent earlier is optionally supported.

(9) Path Control Manager

- Provides unsolicited notification to CPMS of problems detected by the component.

(11) Session Services

- Session services in an end node provides unsolicited notification to CPMS when the CP-CP sessions to the network node server becomes active or inactive.

- Provides unsolicited notification to CPMS of problems detected by the component.

(12) Configuration Services

- Provides unsolicited notification to CPMS of problems detected by the component.

(15) Address Space Manager

- Provides unsolicited notification to CPMS of problems detected by the component.

(18) Session Connector Manager

| - Provides unsolicited notification to CPMS of problems detected by the component.

(19) Topology and Routing Services

| - Provides unsolicited notification to CPMS of problems detected by the component.

(20) Directory Services

| - Upon request, directory services locates network resources for CPMS.

- Provides unsolicited notification to CPMS of problems detected by the component.

(21) Presentation Services

- Control Point presentation services provides the LU 6.2 protocol boundary used by CPMS service transaction programs that communicate over CP-CP sessions (i.e., sessions for which the control point is the endpoint).

(22) Node Operator Facility

- The node operator facility is the component that provides for operator control of the local node, such as initialization of other control point components (including management services).

1.1.3 Overview of Management Services Communication

Communication between CPMS and PUMS is accomplished in two ways:

- Over SSCP-PU sessions
- Over LU-LU sessions used by SNA/Distribution Services (SNA/DS)

Communication between instances of CPMS in different nodes is accomplished in two ways:

- Over CP-CP and LU-LU sessions used by multiple-domain support (MDS)
- Over LU-LU sessions used by SNA/Distribution Services

Terminology

In the sections that follow, the terms XXXX_NETOP and EP_XXXX are used respectively to represent focal point and entry point *function sets*. Refer to "Implementation Choices" in topic 1.1.4 for a description of the function set concept.

Subtopics:

- 1.1.3.1 Communication Between CPMS and PUMS
- 1.1.3.2 Communication Between CPMS in Different T2.1 Nodes

1.1.3.1 Communication Between CPMS and PUMS

Subtopics:

- 1.1.3.1.1 SSCP-PU Sessions
- 1.1.3.1.2 SNA/Distribution Services

1.1.3.1.1 SSCP-PU Sessions

```
|                              °      ° |        |              "                  °  |   ° |
|  ___________________________  °      ° |        |  ___________________________  °  |   ° |
| |     path control      | LMS | °      ° |        | |     path control      | LMS | °  |   ° |
| |_____________________________| °      ° |        | |_____________________________| °  |   ° |
|                              °      ° |        |              "                  °  |   ° |
|  ___________________________  °      ° |        |  ___________________________  °  |   ° |
| |   data link control   | LMS | °      ° |        | |   data link control   | LMS |___|   ° |
| |_____________________________| °      ° |        | |_____________________________| °      ° |
|                              °      ° |        |              "                  °      ° |
|  ___________________________  °      ° |        |  ___________________________  °      ° |
| |    physical control   | LMS | °      ° |        | |    physical control   | LMS | °      ° |
| |_____________________________| °°°°°°°° |        | |_____________________________| °°°°°°°° |
|             |    node containing SSCP    |        |              "    node containing PU     |
|_____________|____________________________|        |_____________|___________________________|
              |____________________________________________________|
```

Figure 1-7. CPMS to PUMS Communication on the SSCP-PU Session. PUMS communicates with the LMS for each layer in its node.

Figure 1-7 illustrates how a network operator causes a command to be delivered to a layer manager in a remote node. In this illustration, the network operator interacts with an MS function set at the SSCP controlling the PU. The MS function set prepares the architecturally defined encoding corresponding to the operator command, then passes it to the data flow control layer (top layer of the SSCP-PU session). The message is passed through the SNA layers at both nodes, and finally delivered by the half-session to the MS function set at the remote PUMS. The MS function set interprets the command and passes the appropriate signals to the *Local management services* (LMS) for the layer (data link control in this illustration).

Any replies would follow the same path in reverse to get back to the network operator at the SSCP.

1.1.3.1.2 SNA/Distribution Services

```
                   _________________
                  | network operator |
                  |_________________|
 _________________________|_______________        _____________________________________________
| °°°°°°°°°°°°°°°°°°°°°°°°|°°°°°°°°°°°°°° |      | °°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°° |
| °                                CPMS ° |      | °                                  PUMS ° |
| °  ____________   ___________         ° |      | °  ____________   ___________         ° |
| ° |file services|__|MS function|       ° |      | ° |file services|__ÿ|MS function|       ° |
| ° |   support   |  |    set    |       ° |      | ° |   support   |  |    set    |       ° |
| ° |____________|  |___________|       ° |      | ° |____________|  |___________|       ° |
| °°°°°°°  °°°°°°°°°°°°°°°°°°°°°°°      ° |      | °°°°°°° " °°°°°°°°°°°°°°°°°°°°°°°°°      ° |
|  _____________________________         ° |      |  _____________________________         ° |
| | SNA/Distribution | SNA/File | °      ° |      | | SNA/Distribution | SNA/File | °      ° |
| | Services         | Services | °      ° |      | | Services         | Services | °      ° |
| |_____________________________| °      ° |      | |_____________________________| °      ° |
|                                °      ° |      |              "                  °      ° |
|  _____________________________ °      ° |      |  _____________________________ °      ° |
| | presentation services | LMS | °      ° |      | | presentation services | LMS | °      ° |
| |_____________________________| °      ° |      | |_____________________________| °      ° |
|                                °      ° |      |              "                  °      ° |
|  _____________________________ °      ° |      |  _____________________________ °      ° |
| |   data flow control   | LMS | °      ° |      | |   data flow control   | LMS | °      ° |
| |_____________________________| °      ° |      | |_____________________________| °      ° |
|                                °      ° |      |              "                  °      ° |
|  _____________________________ °      ° |      |  _____________________________ °      ° |
| | transmission control  | LMS | °      ° |      | | transmission control  | LMS | °      ° |
| |_____________________________| °      ° |      | |_____________________________| °      ° |
|                                °      ° |      |              "                  °      ° |
|  _____________________________ °      ° |      |  _____________________________ °      ° |
| |     path control      | LMS | °      ° |      | |     path control      | LMS | °      ° |
| |_____________________________| °      ° |      | |_____________________________| °      ° |
|                                °      ° |      |              "                  °      ° |
|  _____________________________ °      ° |      |  _____________________________ °      ° |
| |   data link control   | LMS | °      ° |      | |   data link control   | LMS | °      ° |
| |_____________________________| °      ° |      | |_____________________________| °      ° |
|                                °      ° |      |              "                  °      ° |
|  _____________________________ °      ° |      |  _____________________________ °      ° |
| |    physical control   | LMS | °      ° |      | |    physical control   | LMS | °      ° |
| |_____________________________| °°°°°°°° |      | |_____________________________| °°°°°°°° |
|              |              focal point |      |              "              entry point |
|______________|__________________________|      |______________|__________________________|
               |____________________________________________________|
```

Figure 1-8. Communication between CPMS and PUMS Using SNA/DS over LU-LU Session

Figure 1-8 illustrates how MS function sets use SNA/File Services and SNA/Distribution Services for distribution of potentially large files, requests to manipulate them, and reports to track the distribution and installation. SNA/DS uses an LU-LU session, not the SSCP-PU session, for this communication.

This method of communication is used by the change management category.

1.1.3.2 Communication Between CPMS in Different T2.1 Nodes

Subtopics:

- 1.1.3.2.1 Multiple-Domain Support
- 1.1.3.2.2 SNA/Distribution Services

1.1.3.2.1 Multiple-Domain Support

```
   _________________
  | network operator |
  |__________________|
 _________|_____________________________________        _____________________________________________
| ooooooo  oooooooooooooooooooooooooooooo |        | oooooooooooooooooooooooooooooooooooooooooo |
| o                                  CPMS o |        | o                                 _______    CPMS o |
| o |xxxx_NETOP|                         o |        | o                                |EP_xxxx|____    o |
| o |__________|                         o |        | o                                |_______|   |    o |
| o                                      o |        | o                                  "       |   o |
| o  ___________________________          o |        | o  ___________________________              |    o |
| o |  multiple-domain support  |         o |        | o |  multiple-domain support  |             |    o |
| o |___________________________|         o |        | o |___________________________|             |    o |
| oooooooooooo  oooooooooooooooooooo      o |        | oooooooooooo " oooooooooooooooooooo         |   o |
|  ___________________________  o         o |        |  ___________________________  o            |    o |
| | presentation services | LMS | o       o |        | | presentation services | LMS | o          |    o |
| |_______________________________| o       o |        | |_______________________________| o          |    o |
|                                 o       o |        |              "                  o          |   o |
|  ___________________________  o         o |        |  ___________________________  o            |    o |
| |   data flow control   | LMS | o       o |        | |   data flow control   | LMS | o          |    o |
| |_______________________________| o       o |        | |_______________________________| o          |    o |
|                                 o       o |        |              "                  o          |   o |
|  ___________________________  o         o |        |  ___________________________  o            |    o |
| | transmission control  | LMS | o       o |        | | transmission control  | LMS | o          |    o |
| |_______________________________| o       o |        | |_______________________________| o          |    o |
|                                 o       o |        |              "                  o          |   o |
|  ___________________________  o         o |        |  ___________________________  o            |    o |
| |     path control      | LMS | o       o |        | |     path control      | LMS | o          |    o |
| |_______________________________| o       o |        | |_______________________________| o          |    o |
|                                 o       o |        |              "                  o          |   o |
|  ___________________________  o         o |        |  ___________________________  o            |    o |
| |   data link control   | LMS | o       o |        | |   data link control   | LMS |___|        o |
| |_______________________________| o       o |        | |_______________________________| o          o |
|                                 o       o |        |              "                  o          o |
|  ___________________________  o         o |        |  ___________________________  o            o |
| |    physical control   | LMS | o       o |        | |    physical control   | LMS | o          o |
| |_______________________________| oooooooo |        | |_______________________________| oooooooo |
|             |                focal point |        |              "                entry point |
|_____________|____________________________|        |_____________|____________________________|
              |____________________________________________________|
```

Figure 1-9. Communication between CPMS in T2.1 Nodes. Multiple-domain support is used over CP-CP and LU-LU sessions.

Figure 1-9 illustrates how a network operator causes a command to be delivered to a layer manager in a remote node. In this illustration, the network operator interacts with an xxxx_NETOP function set at a focal point CPMS. The xxxx_NETOP function set prepares the architecturally-defined encoding corresponding to the operator command and passes it to *multiple-domain support* (MDS). MDS interacts with presentation services of the CP to send the message over a CP-CP or LU-LU session using LU 6.2 protocols. The message is passed through the SNA layers at both nodes, and finally delivered by MDS to the EP_xxxx function set at the remote CPMS. The EP_xxxx function set interprets the command and passes the appropriate signals to the *local management services* (LMS) for the layer (data link control in this illustration).

Any replies would follow the same path in reverse to get back to the network operator at the focal point CPMS.

1.1.3.2.2 SNA/Distribution Services

```
                  _________________
                 | network operator |
                 |__________________|
 _________________________|_____________        _____________________________________________
| oooooooooooooooooooooooooo|oooooooooooooo |        | oooooooooooooooooooooooooooooooooooooooooo |
| o                                  CPMS o |        | o                                  CPMS o |
| o  ____________    __________          o |        | o  ____________    __________          o |
| o |file services|__|xxxx_NETOP|         o |        | o |file services|__ÿ| EP_xxxx  |         o |
| o |   support   |  |__________|         o |        | o |   support   |  |__________|         o |
| o |____________|                        o |        | o |____________|                        o |
```

```
| °°°°°°°  °°°°°°°°°°°°°°°°°°°°°°°      ° |        | °°°°°°° " °°°°°°°°°°°°°°°°°°°°°°°      ° |
|  ___________________________________  °      ° |        |  ___________________________________  °      ° |
| | SNA/Distribution | SNA/File | °      ° |        | | SNA/Distribution | SNA/File | °      ° |
| | Services         | Services | °      ° |        | | Services         | Services | °      ° |
| |__________________________________| °      ° |        | |__________________________________| °      ° |
|                                     °      ° |        |                    "                  °      ° |
|  ___________________________________  °      ° |        |  ___________________________________  °      ° |
| | presentation services | LMS | °      ° |        | | presentation services | LMS | °      ° |
| |__________________________________| °      ° |        | |__________________________________| °      ° |
|                                     °      ° |        |                    "                  °      ° |
|  ___________________________________  °      ° |        |  ___________________________________  °      ° |
| |   data flow control   | LMS | °      ° |        | |   data flow control   | LMS | °      ° |
| |__________________________________| °      ° |        | |__________________________________| °      ° |
|                                     °      ° |        |                    "                  °      ° |
|  ___________________________________  °      ° |        |  ___________________________________  °      ° |
| | transmission control  | LMS | °      ° |        | | transmission control  | LMS | °      ° |
| |__________________________________| °      ° |        | |__________________________________| °      ° |
|                                     °      ° |        |                    "                  °      ° |
|  ___________________________________  °      ° |        |  ___________________________________  °      ° |
| |     path control      | LMS | °      ° |        | |     path control      | LMS | °      ° |
| |__________________________________| °      ° |        | |__________________________________| °      ° |
|                                     °      ° |        |                    "                  °      ° |
|  ___________________________________  °      ° |        |  ___________________________________  °      ° |
| |   data link control   | LMS | °      ° |        | |   data link control   | LMS | °      ° |
| |__________________________________| °      ° |        | |__________________________________| °      ° |
|                                     °      ° |        |                    "                  °      ° |
|  ___________________________________  °      ° |        |  ___________________________________  °      ° |
| |    physical control   | LMS | °      ° |        | |    physical control   | LMS | °      ° |
| |__________________________________| °°°°°°°° |        | |__________________________________| °°°°°°°° |
|             |                focal point |        |              "               entry point |
|_____________|____________________________|        |______________|___________________________|
              |____________________________________________________|
```

Figure 1-10. Communication between CPMS in Two T2.1 Nodes using SNA/DS over an LU-LU Session

Figure 1-10 illustrates how a network operator causes a an xxxx_NETOP function set to send a command (and optionally a data object, such as a file) to an EP_xxxx function set in a remote CPMS. In this case, SNA/Distribution Services (SNA/DS) is invoked for the transport over an LU-LU session (using LU 6.2 protocols).

1.1.4 Implementation Choices

Subtopics:

- 1.1.4.1 Introduction
- 1.1.4.2 Base and Optional Subsets of the Function Sets
- 1.1.4.3 Role Requirements
- 1.1.4.4 Electives

1.1.4.1 Introduction

Architecture can be implemented in equipment of different capabilities with various physical resources, and with various application program specializations. It is not unusual for a given implementation to choose to implement less than the complete architecture. If the various management services functions were selected individually (that is, on a function by function basis), it is unlikely that any two implementations would select exactly the same set, or that the combination of implementations within a network would result in a complete set. As a result, connectivity could be impaired or impossible, and adequate function within a network could not be guaranteed. Therefore, architecture defines the rules under which implementations select the functions to be supported. The rules for selecting the management services functions to implement are divided into the following categories:

- Base and optional subsets of the function sets
- Role requirements
- Electives

Every implementation makes decisions according to the rules in all of the categories. These rules are defined in the remainder of this section.

1.1.4.2 Base and Optional Subsets of the Function Sets

A management services function set is a collection of services that together perform an overall management services function for a physical unit. In management services, the notion exists of a mandatory or base subset of a management services function set that all implementations of that particular function set must support. The remainder of the management services function set is composed of options that implementations of that function set may choose to support depending on their role requirements. Some subsets not in the base of one role, may be in the base of another role. So, it is possible to have an "optional subset" that is actually required for a particular role. The purpose of defining base and optional functions is to allow implementations to implement the set of functions that fits their requirements and to simultaneously enforce a base level of system integrity. The number of options is large enough that, if they were selected on an individual basis, every implementation could have a unique set. This could seriously reduce connectivity. To avoid this possibility, the options are divided into subsets. These subsets are named and numbered.

The composition of a function set is represented by a diagram, as shown in the lower portion of Figure 1-11 in topic 1.1.4.3. The complete set of functions described by a function set is the outer rectangle. The lower portion of the rectangle contains the subset of functions that is always implemented for that function set (the base subset). The upper portion of the rectangle contains optional subsets that may be chosen to be implemented, depending on role requirements. A subset that is located underneath another subset in the diagram, is a prerequisite for the upper subset and cannot be omitted unless all the subsets above it are also omitted.

Conventions for Function Set Names

In order to clearly distinguish the names of MS function sets when they appear in the text of this publication, they are generally given descriptive multiple-word names, capitalized, and connected with underscore characters (much as procedure names would be written in certain programming languages). For example, the function set that describes how PUMS sends data over the SSCP-PU session is denoted by the name SEND_DATA_SSCP_PU.

Subtopics:

- 1.1.4.2.1 Rules for Function Sets and Their Subsets

1.1.4.2.1 Rules for Function Sets and Their Subsets

- A management services implementation must implement all function sets belonging to the base for its role.
- A management services implementation may implement function sets that are listed as optional for its role, depending on role requirements.
- If a function set is implemented, the base subset of that function set is implemented completely, and optional subsets may be implemented, depending on role requirements.
- Optional subsets of function sets are implemented completely if they are implemented at all.
- An implementation never provides a function defined in a function set in some way different (as perceived outside the node) from that defined by the management services architecture. (An implementation can diverge from the formal model of a node described in this book, but not in ways such that technical variations can be detected outside the node.)

Chapter 9, "General Management Services Function Sets" and Chapter 10, "Specialized Management Services Function Sets for Entry Points" describe the rules for implementation of each of the function sets in further detail.

1.1.4.3 Role Requirements

Defining a base set of functions that all implementations provide and optional sets of functions that implementations can choose according to their additional requirements is not sufficient to ensure system integrity. If all implementations implemented only a single base set of functions, universal connectivity could be achieved, but some implementations would contain unneeded function while the network as a whole could well be without needed function. Because of this, an

implementation's role in the network must be defined before any decision can be made regarding the set of base and optional functions that are to be implemented. A role is composed of base sets of function (base function sets) and optional sets of function (optional function sets). The base subset of each of the base function sets is required to be implemented to define the role, optional function sets may be chosen for implementation depending on product considerations. See the upper portion of Figure 1-11. The roles currently defined for management services are:

° Control point management services (CPMS) in an APPN end node

| ° Control point management services (CPMS) in an APPN network node

° Physical unit management services (PUMS) in a type 2.0 node

A diagram having the general form shown in the upper portion of Figure 1-11 is used to indicate the functional composition of a particular role. The complete set of functions described by the architecture for that role is bounded by the outer rectangle. The portion of the rectangle below the heavy line (°°°°°) contains the function sets whose base subsets are always implemented for that role. The portion of the rectangle above the heavy line (°°°°°) contains function sets that an implementation of that role may choose to include, depending on role requirements. A function set that is located underneath another function set in the diagram is a prerequisite for that function set and is not omitted unless all the function sets above it are also omitted. In the example, role x is composed of function sets a and b and optionally (within the constraints of any stated role requirements), a combination of optional function sets c, d, e, and f. If function set e is selected, function sets d and f are also selected. If function set c is selected, function set d is also selected. Function set d may be selected without function set c, e, or f. Function set f may be selected without function set c, d, or e.

Chapter 8, "Role Requirements of Management Services Components" describes the rules for implementation of each of the roles in further detail.

```
 °°°°°°
```

reasons.

Some choices are included as electives because, although they are observable, they cause no, or minimal, inconvenience to the observers while providing significant value to the particular implementations. Other electives do require significant extra flexibility on the part of the other components. In those cases, the development effort saved, or the improved performance provided, are proportionately more important. All electives are documented in the architecture because they have the potential of affecting connectivity.

An example of a management services elective is the method that PUMS uses in reporting that certain requests cannot be processed. The details of why the request could not be processed is described by sense data. PUMS may *elect* to send certain sense data in either a negative response, or a reply. Electives are described within the function sets where they are allowed. Refer to Chapter 10, "Specialized Management Services Function Sets for Entry Points" for additional details.

1.2 Chapter 2. Management Services Concepts

Subtopics:

- 1.2.1 Introduction to Management Services Formats
- 1.2.2 Generic Management Services Flows
- 1.2.3 Transport of Management Services Data
- 1.2.4 Focal Point Concepts
- 1.2.5 Identification of Resources and Application-Level Routing

---

1.2.1 Introduction to Management Services Formats

The overall management services function is provided by the combined functions of the control point management services, physical unit management services, and local management services components. The primary functions of CPMS in an SSCP are to receive requests for management data from network operators and forward these requests to instances of PUMS, and to receive management data from PUMS and forward this data to network operators. The primary functions of PUMS are to receive requests for management data from CPMS and forward these requests to LMS, and to receive management data from LMS and forward this data to CPMS. (Note that CPMS in a type 2.1 node that uses the SSCP-PU session for its management services communication with the SSCP performs the same functions as PUMS in a type 2.0 node.) The primary functions of LMS are to gather and maintain various types of management data, and to pass this data to PUMS or CPMS, either in reply to a request from PUMS or CPMS, or unsolicited as the result of a specific event.

Request-response unit (RU) is a generic term for a request unit or a response unit. A request unit is a message unit that contains control information. A response unit is a message unit that acknowledges a request unit; it may contain prefix information received in a request unit. If positive, the response unit may contain additional information (such as route status in response to NMVT), or, if negative, contain the sense data identifying the exception condition.

"Communication Between CPMS and PUMS" in topic 1.1.3.1 and "Communication Between CPMS in Different T2.1 Nodes" in topic 1.1.3.2 discuss the various communications between CPMS, PUMS, and LMS. The following is a summary of those communications.

- A network operator communicates with an instance of CPMS.
- Communication between instances of CPMS in different nodes takes place via multiple-domain support (MDS) transport on CP-CP or LU-LU sessions using LU 6.2 protocols and management services formats.
- CPMS communicates with PUMS on a control point to physical unit (SSCP-PU) session using management services RUs.
- Both PUMS and CPMS communicate directly with an instance of LMS.
- Bulk data is transported between CPMS and PUMS, or between instances of CPMS, using SNA/DS protocols on LU-LU sessions.

Subtopics:

- 1.2.1.1 Management Services Formats

1.2.1.1 Management Services Formats

A management services unit (MSU) is a management services encoding that is formatted according to a major vector, subvector, subfield scheme. See "Overview of the Management Services Unit (MSU)" in topic 1.2.1.1.1 for a discussion of this scheme. Table 2-1 summarizes the formats of management services requests and data between PUMS and CPMS, and between different instances of CPMS.

The MSU that flows on an SSCP-PU session is termed a *network management vector transport (NMVT)*.

The MSUs that flow on CP-CP or LU-LU sessions are general data stream (GDS) variables, which adhere to the encoding rules for GDS variables. They may contain other GDS variables, control vectors, or MS major vectors. The GDS variables used by management services are listed in Table 2-1. See *SNA Formats* for a complete description of the encodings mentioned here.

The CP-MSU is a general data stream (GDS) variable that is transported on CP-CP and LU-LU sessions within a Multiple-Domain Support Message Unit (MDS-MU). See "Formats for Multiple-Domain Support" in topic 1.2.3.2.2 for more information about the MDS-MU.

The CP-MSU is also transported within an SNA/DS MU on an SNA/DS conversation over an LU-LU session. The SNA Condition Report (SNACR), FS Action Summary, and the Agent-unit-of work are SNA/DS encodings, and are documented with the rest of the SNA/DS encodings. See "Formats for SNA/DS" in topic 1.2.3.3.3 for more information about the SNA/DS MU. See *SNA Formats* for a complete description of the encodings.

Table 2-1. Summary of Formats Used by Management Services

SSCP-PU Session (RUs)	LU-LU Sessions Used by SNA/DS (GDS Variables)	CP-CP or LU-LU Sessions Used by MDS (GDS Variables)
NMVT	(X'1212') CP-MSU	(X'1212') CP-MSU
		(X'1310') MDS-MU
		(X'1311') MDS Routing Information
		(X'1320') Text Command
		(X'1321') Partial Format Processing Method
		(X'1322') Display Datastream
		(X'1323') Context-Identified Values
		(X'1324') Context Identifier Group
		(X'1325') Context Identifier
		(X'1326') Value Group
		(X'1327') Character Value Descriptor
		(X'1328') Value Processing Method
		(X'1329') Value
		(X'132A') Hexadecimal Value Descriptor
		(X'132B') Nested Value Processing Method
		(X'1331') Value Instance Identifier
	(X'1532') SNA Condition Report	(X'1532') SNA Condition Report
	(X'1548') FS Action Summary	
	(X'1549') Agent Unit of Work Correlator	(X'1549') Agent Unit of Work Correlator

```
|_________________|___________________________|___________________________|
|_________________|___________________________|___________________________|
|                 |                           | (X'154D') Routing and     |
|                 |                           | Targeting Instructions    |
|_________________|___________________________|___________________________|
|_________________|___________________________|___________________________|
```

Figure 2-2 and Figure 2-1 illustrate, respectively, the formats of a CP-MSU and NMVT.

```
 0___________3__5______7_______8_____________byte offset_____________________________________
            |_____________________________management data_____________________________________ÿ|
 ___________ __ ______ _______ ____________________________________________________________
| NS header |  |      |       |                                                            |
| X'41038D' |  | PRID | flags |             management services major vector(s)             |
|           |  |      |       |                                                            |
|___________|__|______|_______|____________________________________________________________|
              |   |
              |   |___ÿ  procedure related identifier (used to correlate requests and replies)
              |___ÿ  retired
```

Figure 2-1. Format of the NMVT Management Services RU

```
                    |_______________________management data_____________________________ÿ|
 ________ ________ ____________________________________________________ ___________________
|        | GDS ID |                                                    |  GDS variables    |
| length | X'1212' |       management services major vector(s)         |  related to the   |
|        |        |                                                    |  MS major vector(s)|
|________|________|____________________________________________________|___________________|
    |
    |
    |
    |___ÿ  2-byte binary length of CP-MSU, including length field
```

Figure 2-2. Format of the CP-MSU Management Services GDS Variable. The 2-byte length field of a GDS variable actually consists of a 1-bit flag followed by a 15-bit integer, so that the maximum length is 32767 bytes. Note, however, that the length is restricted by multiple-domain support to 31743 bytes.

Subtopics:

- 1.2.1.1.1 Overview of the Management Services Unit (MSU)

1.2.1.1.1 Overview of the Management Services Unit (MSU)

```
|                                                                                  |
|_________________________________  major vector  ________________________________ÿ|
|                                                                                  |

 ________ _____ ____________________________________________________________________
|        |     |  ____________  ____________  ____________________________________ |
|        |     |  |          |  |          |  |  __________        __________  | |
| length | key |  |  subvector |  |  subvector | ... | | subfield² |    ...  | subfield  || |
|        |     |  |          |  |          |  |  |__________|subvector|__________|| |
|        |     |  .____________.  |__________|  |_.__________._________________| |
|________|_____|_.________________._________________.____________._________________|
              .                .                .                .
             .                .                .                .
            .                .                .                .
           ________ ______ _______            ________ _____ _______
          |        |      |       |          |        |     |       |
          | length | key¹ | data² |          | length | key | data  |
          |        |      |       |          |        |     |       |
          |________|______|_______|          |________|_____|_______|
```

¹Subvector keys are defined as follows:

Subvectors with keys X'00' through X'7F' are referred to as common subvectors. Their meaning is independent of the major vector within which they are used.

Subvectors with keys X'80' through X'FE' have a meaning unique to the MS major vector within which they are used.

²Data fields of subvectors may contain subfields.

Figure 2-3. Overview of a Management Services Major Vector

Figure 2-3 provides an overview of the major vector, subvector, subfield encoding scheme of an MSU. The *key* of the major vector identifies the management function provided. This generally correlates to an element of a major category of network management. For example, the Alert major vector, identified by a key of X'0000', provides the problem determination, and optionally the problem diagnosis, element of problem management. The Response-Time Monitor major vectors, identified by keys of X'0080' and X'8080', provide the response-time measurement element of performance management. The major vector indicates the function to be provided and includes the subvectors that carry management services data.

GDS variables may also contain subvectors and subfields as shown in Figure 2-3. However, a GDS variable may also contain other GDS variables. Refer to *SNA Formats* for details of the encoding rules for all SNA formats.

The general encoding scheme for MSUs is extended for common operations services and for one problem management scenario. Ordinarily an MSU transports a single major vector. For common operations services (described in "Common Operations Services Function Set" in topic 1.7.2), however, one or more *parameter* major vectors are also included, after certain MS major vectors. A parameter major vector is one that carries information associated with the MS major vector. Thus the entire structure shown in Figure 2-3 is repeated multiple times within a single MSU. For problem management, multiple Alert and/or Resolution major vectors may be carried in a CP-MSU (but not a NMVT). See the "Format Usage" section (beginning on page 1.3.4.5) of "Problem Determination, Diagnosis, and Resolution via Alerts and Resolution Notifications" for more information. For operations management, major vectors, parameter major vectors, and GDS variables may all be combined within a single CP-MSU. See "Operations Management Requests, Replies, and Reports" in topic 1.7.3.1.1 for more information.

When reply (i.e., solicited) data flows on an SSCP-PU session, and a single logical reply exceeds the maximum allowable NMVT size, the associated reply structure spans multiple NMVTs as shown in Figure 2-4. This figure illustrates the case of a single parameter major vector spanning multiple NMVTs. This function applies, as well, to a delimited group of parameter major vectors or to a large MS major vector (with or without associated parameter major vectors) spanning multiple NMVTs. In all of these cases, the length appearing at the start of a major vector spanning NMVTs is for the total spanned length of the major vector. A length and key does not appear in the parts of the major vector that are continued in subsequent NMVTs. Note that the MS major vector appears only once and always begins in the first NMVT of the series. NMVT header flags indicate whether a given NMVT is first, last, or intermediate in the series of NMVTs, and the presence of the NMVT Count (X'44') subvector in the MS major vector distinguishes what follows, in the current NMVT and the remaining NMVTs of the series, as all belonging to a single logical reply. The NMVT Count (X'44') subvector contains the data necessary to calculate the size of the total reply. If the MS major vector itself spans multiple NMVTs, the NMVT Count (X'44') subvector appears in the part of the MS major vector contained in the first NMVT.

The function of a single reply spanning multiple RUs applies only to the NMVT format of the Management Services Unit; i.e., it does not apply to a CP-MSU.

```
 0___________3__5______7_______8______________byte offset_____________________________________
             |_____________________________management data_____________________________ÿ|
 ____________ __ ______ _______ _________________ ___________________________________________
| NS header |  |      |       |                 |                                           |
| X'41038D' |  | PRID | flags | MS major vector |    parameter major vector (first part)    |
|           |  |      |       |                 |                                           |
|___________|__|______|_______|_________________|___________________________________________|

 0___________3__5______7_______8______________byte offset_____________________________________
             |_____________________________management data_____________________________ÿ|
 ____________ __ ______ _______ ____________________________________________________________
| NS header |  |      |       |                                                            |
| X'41038D' |  | PRID | flags |             parameter major vector (second part)           |
|           |  |      |       |                                                            |
|___________|__|______|_______|____________________________________________________________|

 0___________3__5______7_______8______________byte offset____________
             |_____________________________management data________ÿ|
 ____________ __ ______ _______ ____________________________________
| NS header |  |      |       |                                    |
| X'41038D' |  | PRID | flags | parameter major vector (last part) |
|           |  |      |       |                                    |
|___________|__|______|_______|____________________________________|
```

Figure 2-4. Single Reply Spanning Multiple NMVTs. In addition to the example shown here, multiple NMVTs may be spanned by a delimited group of parameter major vectors or by a large MS major vector (with or without associated parameter major vectors).

SNA Formats documents all MSUs. Each major vector description contains a matrix defining the subvectors that may be contained in it.

Subvectors that are unique to a major vector are documented after the major vector description, while those subvectors that can be used by multiple major vectors (called *common subvectors*) are documented at the end of the major vector section. Subfields are ordinarily documented within the description of the subvector in which they are used. SNA/MS GDS variables that contain other GDS variables are documented in tabular style, with descriptions of the contained structures following each table. Those SNA/MS GDS variables that do not contain other GDS variables are documented in the byte-offset style.

1.2.2 Generic Management Services Flows

There are three general patterns for the flow of management services data in a network, termed the *unsolicited* flow, the *request without reply* flow, and the *request/reply* flow. Each management services component, i.e., CPMS, PUMS, and LMS, has a well-defined set of responsibilities for each type of flow. The following sections discuss briefly each of these generic flows.

The following chapters provide a brief discussion of the individual management services flows:

Chapter 3, "Problem Management"
Chapter 4, "Performance and Accounting Management"
Chapter 5, "Configuration Management"
Chapter 6, "Change Management"
Chapter 7, "Operations Management"

The following chapter describes the required and optional groupings of function sets that products implement for the various MS Roles. The dependencies that each of the function sets have upon one another is also described:

Chapter 8, "Role Requirements of Management Services Components"

The responsibilities of the management services component and LMS for each type of flow are described in:

Chapter 9, "General Management Services Function Sets"
Chapter 10, "Specialized Management Services Function Sets for Entry Points"

The figures referenced in the next sections have numbers along the left side identifying each stage of the flow. These same numbers are used in the text description of each stage.

Subtopics:

- 1.2.2.1 Unsolicited Flows
- 1.2.2.2 Request Without Reply Flows
- 1.2.2.3 Request/Reply Flows for Non-Bulk Data
- 1.2.2.4 Request/Reply Flows for Bulk Data

1.2.2.1 Unsolicited Flows

If the Origin is a Type 2.0 Node or a Boundary-Function-Attached Type 2.1 Node

```
     Network                                                  LMS
     Operator              CPMS                PUMS         Component
         .                   .                   .                .
         .                   .                   .                .
         .                   .                   .    MS data     .
   (1)   .                   .                   _________________°
```

```
          .                   .     data NMVT     .                   .
   (2)    .                   _________________°                    .
          .                   .     +RSP(NMVT)    .                   .
   (3)    .                   °___________________ÿ                   .
          .     MS data       .                   .                   .
   (4)    ___________________°                    .                   .
          .                   .                   .                   .
          .                   .                   .                   .
          .                   .                   .                   .
```

Figure 2-5. The Unsolicited Flow for a Type 2.0 Node or Boundary-Function-Attached Type 2.1 Node

Figure 2-5 shows an unsolicited flow of management services data. The flow consists of the following stages:

1. As a result of the occurrence of some specified condition, an LMS component passes management services data to PUMS. An example of a condition that initiates an unsolicited flow is detection of an error of a certain type. An exhaustive list of the conditions under which an LMS component passes data unsolicited to PUMS appears in the discussion of each individual management services function.

2. PUMS formats the data it receives into a management services RU, in some cases adding additional data of its own such as product identification; then it sends the RU to CPMS on an SSCP-PU session. The general term "data record," and the more specific "data NMVT," refer to management services records flowing from PUMS to CPMS. The terms apply to both solicited and unsolicited records. A management services record flowing in the other direction, from CPMS to PUMS, is termed a "request NMVT."

3. CPMS sends a response upon receiving the RU.

4. CPMS passes the management services data to the network operator. Depending on the type of data involved, additional functions may be provided, such as routing the data to different operators, logging the RU, and analyzing the data in certain ways.

| *If the Origin is an APPN Network Node or End Node*

```
      Network            Focal Point        Origin              LMS
      Operator              CPMS             CPMS            Component
         .                   .                .                   .
         .                   .                .                   .
         .                   .                .                   .
         .                   .                .      MS data      .
   (1)   .                   .                _____________________°
         .                   .     MDS-MU     .                   .
   (2)   .                   _________________°                   .
         .      MS data      .                .                   .
   (3)   .___________________°                 .                   .
         .                   .                .                   .
         .                   .                .                   .
```

| If the origin CPMS is the focal point for the particular management
| services category involved, flow (2) does not occur.

| Figure 2-6. The Unsolicited Flow for an APPN Network Node or End Node

Figure 2-6 shows an unsolicited flow of management services data when the entry point is an APPN node. The flow consists of the following stages:

1. As a result of the occurrence of some specified condition, an LMS component passes management services data to CPMS. Examples of conditions that initiate an unsolicited flow include detection of an error of a certain type and overflow of a counter. An exhaustive list of the conditions under which an LMS component passes data unsolicited to CPMS appears in the discussion of each individual management services function.

2. CPMS formats the data it receives into a management services MDS-MU, adding additional data of its own such as product identification and then sends the MDS-MU to CPMS in its focal point over the CP-CP or LU-LU sesions used by multiple-domain support. If the CP-MSU originates in an APPN end node, then it is typically routed through the end node's network node server. The end node, however, specifies the focal point, not the network node server, as the destination of the CP-MSU.

3. After the management services data has reached the destination focal point, CPMS passes it to the network operator. Depending on the type of data involved, additional functions may be provided, such as routing the data to different operators, logging the record, and analyzing the data in certain ways.

1.2.2.2 Request Without Reply Flows

If the Destination is a Type 2.0 Node or a Boundary-Function-Attached Type 2.1 Node

```
     Network                                                    LMS
     Operator              CPMS                PUMS            Component
         .                   .                   .                  .
         .                   .                   .                  .
         .    MS request     .                   .                  .
  (1)    º___________________ÿ                   .                  .
         .                   .   request NMVT    .                  .
  (2)    .                   º___________________ÿ                  .
         .                   .     +RSP(NMVT)    .                  .
  (3)    .                   ___________________º                  .
         .                   .                   .    MS request    .
  (4)    .                   .                   º___________________ÿ
         .                   .                   .                  .
         .                   .                   .                  .
         .                   .                   .                  .
```

Figure 2-7. The Request without Reply Flow a Type 2.0 Node or a Boundary-Function-Attached Type 2.1 Node

Figure 2-7 shows a management services request, typically a command, for which no reply is expected. The flow consists of the following stages:

1. A network operator enters a management services request for one or more specified resources. Certain requests allow the operator to specify only a single resource; others allow for multiple resources to be specified implicitly, by, for example, directing a command to *all* LUs associated with a specified PU. In any case, all resources specified in a request must be associated with a single PU. See the discussion of the individual flow for statements of the types of requests supported. The network operator's request is passed to CPMS.

2. CPMS reformats the request and sends it to the appropriate instance of PUMS on an SSCP-PU session.

3. PUMS receives the request NMVT and parses it. If it finds an error, it sends a -RSP to CPMS. Otherwise, it sends a +RSP.

4. Having received a management services request, PUMS passes the request to the appropriate LMS component. If the request NMVT specified more than one resource, PUMS passes multiple requests, one for each resource to which the request was directed.

| *If the Destination is an APPN Network Node or End Node*

```
   Network                  Requesting       Destination                LMS
   Operator                    CPMS             CPMS                 Component
      .                         .                .                       .
      .                         .                .                       .
      .       MS request        .                .                       .
(1)   º_________________________ÿ                .                       .
      .                         . MDS-MU request .                       .
(2)   .                         º________________ÿ                       .
      .                         .                .       MS request      .
(3)   .                         .                º_________________________ÿ
      .                         .                .                       .
      .                         .                .                       .
```

| Figure 2-8. The Request without Reply Flow for an APPN Network Node or
| End Node

Figure 2-8 shows a management services request for which no reply is
| expected directed to an APPN network node or end node. The flow consists
of the following stages:

1. A network operator enters a management services request for one or more specified resources.

2. Based on the information passed to it in the request, the requesting CPMS formats an MDS-MU containing the request.

3. Having received a request for management services data, the destination CPMS passes the request to the appropriate LMS component. If the request specified more than one resource, CPMS passes multiple requests, one for each resource to which the request was directed.

1.2.2.3 Request/Reply Flows for Non-Bulk Data

If the Destination is a Type 2.0 Node or a Boundary-Function-Attached Type 2.1 Node

Single Reply

```
     Network                                                          LMS
     Operator                    CPMS                PUMS            Component
         .                         .                   .                 .
         . request for MS data     .                   .                 .
  (1)    °_________________________ÿ                   .                 .
         .                         .    request NMVT   .                 .
  (2)    .                         °___________________ÿ                 .
         .                         .    +RSP(NMVT)       .                 .
  (3)    .                         ____________________°                 .
         .                         .                   .   request for MS data  .
  (4)    .                         .                   °_______________________ÿ
         .                         .                   .       MS data          .
  (5)    .                         .                   _______________________°
         .                         .    data NMVT¹     .                 .
  (6)    .                         ____________________°                 .
         .                         .    +RSP(NMVT)¹    .                 .
  (7)    .                         °___________________ÿ                 .
         .       MS data           .                   .                 .
  (8)    _________________________°                   .                 .
         .                         .                   .                 .
         .                         .                   .                 .
```

¹ This request/reply exchange may occur more than one time.
See Figure 7-2 in topic 1.7.2.4 for an example flow.

Figure 2-9. The Request/Reply Flow for Non-Bulk Data for a Type 2.0 Node or a Boundary-Function-Attached Type 2.1 Node

Multiple Replies

```
     Network                                                          LMS
     Operator                    CPMS                PUMS            Component
         .                         .                   .                 .
         . request for MS data     .                   .                 .
  (1)    °_________________________ÿ                   .                 .
         . (multiple resources)    .    request NMVT   .                 .
  (2)    .                         °___________________ÿ                 .
         .                         .    +RSP(NMVT)     .                 .
  (3)    .                         ____________________°                 .
         .                         .                   .                 .
         .                         .                   .   request for MS data  .
  (4)    .                         .                   °_______________________ÿ
         .                         .                   .       (resource 1)     .
         .                         .                   .         MS data        .
  (5)    .                         .                   _______________________°
         .                         .    data NMVT      .       (resource 1)     .
  (6)    .                         ____________________°                 .
         .                         .    +RSP(NMVT)     .                 .
  (7)    .                         °___________________ÿ                 .
         .       MS data           .                   .                 .
  (8)    _________________________°                   .                 .
         .       (resource 1)      .                   .                 .
         .                         .                   .                 .
         .                         .                   .   request for MS data¹ .
  (4)    .                         .                   °_______________________ÿ
         .                         .                   .       (resource 2)     .
         .                         .                   .         MS data        .
```

```
  (5)    .                      .                    _____________________°
         .                      .    data NMVT       .       (resource 2)    .
  (6)    .                      _________________°                           .
  (7)    .                      .    +RSP(NMVT)      .            °          .
         .                      °_________________ÿ            °          .
         .       MS data        .                    .            °          .
  (8)    _______________________°         °          .                       .
         .       (resource 2)   .                    .                       .
         .                      .                    .                       .
         .                      .                    .   request for MS data .
  (4)    .                      .         °          °_______________________ÿ
         .                      .         °          .       (resource n)    .
         .           °          .                    .         MS data       .
  (5)    .           °          .                    _____________________°
         .           °          .    data NMVT       .       (resource n)    .
  (6)    .                      _________________°                           .
         .                      .    +RSP(NMVT)      .                       .
  (7)    .                      °_________________ÿ                           .
         .       MS data        .                    .                       .
  (8)    _______________________°                    .                       .
         .       (resource n)   .                    .                       .
         .                      .                    .                       .
```

[1] The architecture allows requests to the LMS to be made before previous requests have been replied to and sent to CPMS.

Figure 2-10. The Request/Reply Flow for Non-Bulk Data (Multiple Resources) for a Type 2.0 Node or a Boundary-Function-Attached Type 2.1 Node. This flow is the same as the flow for a single resource (Figure 2-9) except that stages 4 through 8 are repeated for each resource specified.

Figure 2-9 and Figure 2-10 show two varieties of a request/reply flow for non-bulk management services data. The flow consists of the following stages:

1. A network operator requests a particular type of management services data from one or more specified resources. Requests for certain types of data allow the operator to specify only a single resource; others allow for multiple resources to be specified implicitly, by, for example, requesting data from *all* LUs associated with a specified PU. In any case, all resources specified in a request must be associated with a single PU. See the discussion of the individual flow for statements of the types of requests supported. The network operator's request is passed to CPMS.

2. CPMS reformats the request and sends it to the appropriate instance of PUMS on an SSCP-PU session.

3. PUMS receives the request NMVT and parses it. If it finds an error, it sends a -RSP to CPMS. Otherwise, it sends a +RSP.

4. Having received a request for management services data, PUMS passes the request to the appropriate LMS component. If the request NMVT specified more than one resource, PUMS passes multiple requests, one for each resource for which data was requested.

5. The LMS component acts on each request that has been passed to it. It gathers the requested data and passes it to PUMS.

6. PUMS builds a management services data NMVT with the data that was passed to it, including additional data of its own, such as a time stamp; then it sends the NMVT to CPMS on an SSCP-PU session.

7. CPMS sends a response upon receiving the NMVT.

8. CPMS passes the requested management services data to the network operator that made the request.

One aspect of the soliciting/solicited flow not apparent in Figure 2-9 and Figure 2-10 is the multiplicity of levels at which request/reply correlation must take place. PUMS must first correlate the data it receives from LMS with the request that elicited it. Next, CPMS must correlate the data NMVTs it receives with the request NMVTs that it has sent.

Correlation is made more complicated in all of these instances by the following:

- MS major vectors may be sent either solicited or unsolicited. Thus, in a particular case, one of these major vectors may correlate with no request at all.

- When multiple resources are specified on a request, more than one

reply will correlate with that single request. Thus, at both points of correlation, it must be determined not only which request a particular reply correlates to, but also when the *last* reply for a given request has been received.

The NMVT contains a Procedure Related Identifier (PRID) field to be used by the CPMS to correlate the data NMVTs it receives with the request NMVTs it sends. Chapter 10, "Specialized Management Services Function Sets for Entry Points" provides further details of how request/reply correlation is done at each level.

| *If the Destination is an APPN Network Node or End Node*

```
   Network                  Requesting       Destination                 LMS
   Operator                    CPMS             CPMS                  Component
      .                        .                .                        .
      .                        .                .                        .
      .  request for MS data   .                .                        .
(1)   °________________________ÿ                .                        .
      .                        . MDS-MU request .                        .
(2)   .                        °________________ÿ                        .
      .                        .                .  request for MS data   .
(3)   .                        .                °________________________ÿ
      .                        .                .        MS data         .
(4)   .                        .                ________________________°
      .                        .  MDS-MU data   .                        .
(5)   .                        ________________°                        .
      .      MS data           .                .                        .
(6)   ________________________°                .                        .
      .                        .                .                        .
      .                        .                .                        .
```

| Figure 2-11. The Request/Reply Flow for Non-Bulk Data from an APPN
| Network Node or End Node

Figure 2-11 shows a request/reply flow of non-bulk management services
| data when the destination of the request is an APPN network node or end node. The flow consists of the following stages:

1. A network operator requests a particular type of management services data from one or more specified resources.

2. Based on the information passed to it in the request, the requesting CPMS formats a request CP-MSU. The CP-MSU flows between the requesting control point and the control point that controls the specified resources over CP-CP or LU-LU sessions used by Multiple-domain support.

3. Having received a request for management services data, the destination CPMS passes the request to the appropriate LMS component. If the request CP-MSU specified more than one resource, CPMS passes multiple requests, one for each resource for which data was requested.

4. The LMS component acts on each request that has been passed to it. It gathers the requested data and passes it to CPMS.

5. CPMS builds a CP-MSU with the data that was passed to it, including additional data of its own, such as product identification; then it sends the CP-MSU to CPMS in the requesting control point over CP-CP or LU-LU sessions used by Multiple-domain support.

6. After the requested management services data has reached the requesting control point, CPMS passes it to the network operator that made the request.

1.2.2.4 Request/Reply Flows for Bulk Data

Retrieving Bulk Data

```
                    ________ Focal Point __________       _________ Entry Point ___________

   Network         SNA/MS        SNA/FS      SNA/DS      SNA/DS        SNA/FS       SNA/MS
   Operator                      server                                server
      .               .            .           .           .             .            .
      .               .            .           .           .             .            .
      .  request for  .            .           .           .             .            .
      .  MS data      .            .           .           .             .            .
(1)   °_______________ÿ            .           .           .             .            .
```

```
      .               .  SNA/FS     .           .           .           .            .
      .               .  request    .           .           .           .            .
(2)   .               °_________________________ÿo_________ÿo___________________________ÿ
      .               .             .           .           .           .  SNA/FS    .
      .               .             .           .           .           .  request   .
(3)   .               .             .           .           .           _____________°
      .               .             .           .bulk data  .           .            .
(4)   .               .             ____________o_________o____________°            .
      .               .  signal     .           .           .           .            .
(5)   .               _______________°           .           .           .            .
      .  reply        .             .           .           .           .            .
(6)   ________________°             .           .           .           .            .
      .               .             .           .           .           .            .
      .               .             .           .           .           .            .
```

Figure 2-12. The Request/Reply Flow for Retrieving Bulk Data from an Entry Point

Figure 2-12 shows a request/reply flow to *retrieve bulk* management services data.

SNA/File Services (SNA/FS) architecture is used by management services for bulk data transport. For a description of this architecture, refer to *SNA/File Services Reference*.

For a more detailed understanding of the architectural components traversed in each node, and their sequence, refer to "Transport of Bulk Management Services Data" in topic 1.2.3.3.

The flow consists of the following stages:

1. A network operator requests a particular type of bulk management services data. The network operator's request is passed to SNA/MS in the focal point.

2. Based on the information passed to it in the request, the focal point SNA/MS formats an SNA/FS request. The request flows on an SNA/DS conversation between the focal point and the entry point using an LU-LU session.

3. Having received a request for management services data, SNA/MS passes the request to the SNA/FS server.

4. The SNA/FS server fetches the requested bulk data and sends it to the focal point SNA/FS server on an SNA/DS conversation using an LU-LU session.

5. The SNA/FS server at the focal point stores the bulk data and SNA/DS notifies SNA/MS that it has arrived, passing the report in the agent object.

6. SNA/MS notifies the network operator that the bulk data has arrived.

Sending Bulk Data

```
                    ________ Focal Point __________        _________ Entry Point ___________

   Network          SNA/MS        SNA/FS      SNA/DS      SNA/DS        SNA/FS       SNA/MS
   Operator                       server                                server
      .               .            .           .           .            .            .
      .               .            .           .           .            .            .
      .request to send             .           .           .            .            .
      .  SNA/MS data  .            .           .           .            .            .
(1)   °_______________ÿ            .           .           .            .            .
      .               .            .           .           .            .            .
      .               .  request   .           .           .            .            .
(2)   .               °____________ÿ           .           .            .            .
      .               .            .           .           .            .            .
      .               .            .       request and bulk data        .            .
(3)   .               .            °__________ÿo__________ÿo____________ÿ            .
      .               .            .           .           .            .            .
      .               .            .           .           .            .  request   .
(4)   .               .            .           .           .            °____________ÿ
      .               .            .           .  report   .            .            .
(5)   .               ___________________________o__________o___________________________°
      .  report       .            .           .           .            .            .
(6)   ________________°            .           .           .            .            .
      .               .            .           .           .            .            .
      .               .            .           .           .            .            .
```

Figure 2-13. The Request/Reply Flow for Sending Bulk Data to an Entry Point

Figure 2-13 shows a request/reply to *send bulk* management services data. The flow consists of the following stages:

1. A network operator requests that a particular type of bulk management services data be sent to an entry point. A management services function related to the data may be requested in addition. The network operator's request is passed to SNA/MS in the focal point.

2. Based on the information passed to it in the request, the focal point SNA/MS formats either a request CP-MSU or an SNA/FS request, depending on whether or not a management services function over and above simple file transfer is requested. SNA/MS invokes its SNA/FS server to fetch the bulk data from storage.

3. The request and bulk data flow on an SNA/DS conversation between the focal point and the entry point using an LU-LU session.

4. The SNA/FS server in the entry point stores the bulk data and SNA/DS notifies SNA/MS that it has arrived, passing the CP-MSU or SNA/FS request in the agent object.

5. SNA/MS generates either a report CP-MSU or an SNA/FS report as appropriate, and sends it to the focal point SNA/MS on an SNA/DS conversation using an LU-LU session.

6. After the report has reached the focal point, SNA/MS passes it to the network operator that made the request.

1.2.3 Transport of Management Services Data

The following sections describe the options available to management services application programs for transporting management services data. The SSCP-PU session is used for transferring management services data between a control point and a physical unit. The characteristics of this session are described briefly in "Transport of Management Services Data on the SSCP-PU Session" in topic 1.2.3.1.

The Change Management category uses SNA/File Services and SNA/Distribution Services for distribution of potentially large files, requests to manipulate them, and reports to track the distribution and installation. These employ LU-LU sessions. The management services transport for bulk data is described in "Transport of Bulk Management Services Data" in topic 1.2.3.3.

The third type of management services transport, defined for the transfer of MS data between control points, is termed *multiple-domain support* (MDS). The management services function set that describes the implementation of multiple-domain support is named MULTIPLE_DOMAIN_SUPPORT. This function set name or the acronym MDS are used throughout this publication to refer to *multiple-domain support*. This transport provides for transaction routing between management services application programs via CP-CP or LU-LU sessions. MDS is described in "Transport of Non-Bulk Management Services Data in APPN Networks" in topic 1.2.3.2.

Table 2-2 summarizes the management services transport options used by each of the specialized MS function sets. For the EP_XXXX function sets, it shows the option used for sending unsolicited data. An EP_XXXX function set always sends a reply via the same transport over which it received the request to which it is replying.

Table 2-2. Choice of Transport by MS Function Sets for Unsolicited Data

MS Function Set	**SSCP-PU**	**MDS (CP-CP and LU-LU)**	**File Services Support (LU-LU)**
EP_ALERT	YES(1)	YES(1)	--
EP_RTM	YES	--	--
EP_QPI(2)	--	--	--
EP_COMMON_OPERATIONS_SERVICES	YES(3)	YES(3)	--
EP_OPERATIONS_MGMT	--	YES	--

EP_CHANGE_MANAGEMENT	--	--	YES
MS_CAPS	--	YES	--

Notes:

1. This function set uses either MDS or the SSCP-PU session for its unsolicited records. If it currently has knowledge of a focal point it uses the MDS transport to send its Alerts. Otherwise the function set uses the SSCP-PU session.

2. This function set does not send any unsolicited records.

3. This function set uses either MDS or the SSCP-PU session for its one unsolicited flow, Send Message to Operator. The choice is based upon the destination NAU, i.e., use the SSCP-PU session if the destination is an SSCP, use MDS otherwise.

Subtopics:

- 1.2.3.1 Transport of Management Services Data on the SSCP-PU Session
- 1.2.3.2 Transport of Non-Bulk Management Services Data in APPN Networks
- 1.2.3.3 Transport of Bulk Management Services Data

1.2.3.1 Transport of Management Services Data on the SSCP-PU Session

Subtopics:

- 1.2.3.1.1 Overview of the SSCP-PU Session
- 1.2.3.1.2 Format Used on the SSCP-PU Session

1.2.3.1.1 Overview of the SSCP-PU Session

The primary path for transport of SNA/MS data between CPMS and PUMS is the SSCP-PU session. SNA/MS plays no role in the establishment of this session. Since the session is established when a PU is activated, it is already present when PUMS comes up. From the point of view of PUMS, the SSCP-PU session is simply a pipe through which management services requests and data can be exchanged with the PU's controlling SSCP.

The profiles for the SSCP-PU session are documented in *SNA Formats*. The SSCP-PU session for a type 2.0 node or a boundary-function-attached type 2.1 node operates according to TS profile 1 and FM profile 0. An important property of the session for SNA/MS is the maximum RU size: 256 bytes outbound (i.e., from the SSCP to the PU) and 512 bytes inbound (from the PU to the SSCP). See *SNA Formats* for further properties of the SSCP-PU session.

1.2.3.1.2 Format Used on the SSCP-PU Session

As described in "Management Services Formats" in topic 1.2.1.1, the management services RU that flows on an SSCP-PU session is the *network management vector transport* (NMVT). The general encoding scheme for NMVTs is extended in the case of common operations services in that one or more *parameter major vectors* may follow the MS major vector in the NMVT.

Use of parameter major vectors provides the potential for a large amount of data to be transported from the PU to the SSCP as part of a single reply. As a result, it is possible that a single reply may exceed the maximum allowable NMVT size of 512 bytes inbound, as discussed in "Overview of the SSCP-PU Session" in topic 1.2.3.1.1. When this happens, the reply may span multiple NMVTs, where each NMVT contains a portion of the total reply. This scheme is described in "Management Services Formats" in topic 1.2.1.1 and in Figure 2-4 in topic 1.2.1.1.1.

The capability of transporting data that spans multiple NMVTs is strictly limited to reply flows. All command and request flows, whether or not they contain parameter major vectors, are always fully contained in a single NMVT and are subject to the size limitation of 256 bytes. Unsolicited messages to an operator are similarly constrained, but have an NMVT size limitation of 512 bytes. These limits are session related and

are discussed in "Overview of the SSCP-PU Session" in topic 1.2.3.1.1.

1.2.3.2 Transport of Non-Bulk Management Services Data in APPN Networks

Subtopics:

- 1.2.3.2.1 Overview of Multiple-Domain Support
- 1.2.3.2.2 Formats for Multiple-Domain Support
- 1.2.3.2.3 Routing Messages Between Nodes (Overview)
- 1.2.3.2.4 Routing Messages to Application Programs (Overview)

1.2.3.2.1 Overview of Multiple-Domain Support

```
|                                          _______________
|                                        _______________  |
|                                      _______________  | |
|                                    |  user-defined  | | |
|     ________________________________| MS application | |_|
|     |            NOF               | |   programs    |_|
|     |______________________________| |_______________|
|                   "                        "
|     ¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤|¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤    ¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤____
|     ¤              |_______________        |                          ¤    ¤                                      ¤
|     ¤                                                  ___________________________________________                ¤
|     ¤       ___________          ___________      |                                          |                ¤
|     ¤     ___________  |         |            |     |                  _________________       |                ¤
|     ¤   ___________  | |_______ÿ|            |____|_________________ÿ| MDS_HP_SEND       |     |                ¤
|     ¤  |    node    | | |        |            |    |                  _________________       |     |                ¤
|     ¤  | components | |_|        |            |____|________________| MDS_HP_RECEIVE (3)|  |     |                ¤
|     ¤  |            |_|          |            |    |                  _________________  | |     |                ¤
|     ¤  |___________|            |            |____|____________| MSU_HANDLER    (2)|  |  |     |                ¤
|     ¤     ________              |            |    |                  _________________  |  |__|     |                ¤
|     ¤     |     DS  |            |            |____|________| MDS_SON           |  |  |        |                ¤
|     ¤  ________      |            |     MS     |    |                  _________________  |  |__|        |                ¤
|     ¤  |     SS  |  |            |            |____|____ÿ| MDS_SEND          |  |  |        |                ¤ TS
|     ¤  |         |  |            |            |    |                  _________________  |  |__|        |                ¤
|     ¤  |         |  |__________ÿ|            |____|___| MDS_RECEIVE    (1)|  |  |            |                ¤
|     ¤  |         |  |            |            |    |   |                  |  |__|            |                ¤
|     ¤  |         |_|___________ÿ|            |    |   |                  |  |    Service    |                ¤
|     ¤  |         |  |            |            |    |   |                  |__|     Transaction |                ¤
|     ¤  |         |  |            |            |    |   |                  |       Programs    |                ¤
|     ¤  |         |__|            |            |    |   |_________________|       (STPs)      |                ¤
|     ¤  |         |               |            |    |                                          |                ¤
|     ¤  |_________|               |____________|    |__________________________________________|                ¤
|     ¤                                                  "          ¤    ¤           "                          ¤
|     ¤                                              ______|________  ¤    ¤       ______|________                ¤____
|     ¤                                             |              ° ¤    ¤     |              °                ¤
|     ¤                                          ___|____________ ° | ¤    ¤  ___|____________ ° |                ¤
|     ¤                                         | Presentation   °___| ¤    ¤ | Presentation   °___|                ¤
|     ¤                                         | Services (PS)  |     ¤    ¤ | Services (PS)  |                    ¤ PS
|     ¤                                         |________________|     ¤    ¤ |________________|                    ¤
|     ¤                                                  "          ¤    ¤           "                          ¤
|     ¤                                              ______|________  ¤    ¤       ______|________                ¤____
|     ¤                                             |              ° ¤    ¤     |              °                ¤
|     ¤                                          ___|____________ ° | ¤    ¤  ___|____________ ° |                ¤ DFC
|     ¤                                         | CP-CP Half-    °___| ¤    ¤ | LU-LU Half-    °___|                ¤
|     ¤                                         | Session (HS)   |     ¤    ¤ | Session (HS)   |                    ¤____
|     ¤                                         |________________|     ¤    ¤ |________________|                    ¤
|     ¤                        CONTROL POINT                 "          ¤    ¤           "          LOGICAL UNIT      ¤ TC
|     ¤                                                      |          ¤    ¤           |                            ¤
|     ¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤
```

name, so that the distinction between CP and LU is not apparent. The MDS model assumes that the CP and LU have been defined and packaged in this way.

```
|                                                   _______________
|                                                 _______________  |
|                                               _______________  | |
|                                              |  user-defined  | | |
|                                              | MS application | |_|
|                                              |    programs    |_|
|                                              |_______________|
|                                                     "
|                                 ____________________________|______
|                                |          _______________  |      |
|                                |        _______________  |  |      |
|                                |      _______________  |  |  |      |
|              ___________       |      | architecture-  | |  |  |      |
|            ___________  |  |___|____ÿ|    defined     | |  |  |      |
|           |___________  | |  |      | MS application | |_|  |  |      |
|           |    node     | |  |      |    programs    |_|    |  |      |
|           | components  | |_|      |_______________|       |  |      |
|           |             |_|        |       "               |  |  |  ____________________________________________
|           |_____________|          |                       |  |  |                                            |
|                                    | ¤¤¤¤¤¤¤¤¤¤¤¤¤|¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤|¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤
```

between MS application programs in different nodes within a network. For outgoing messages, the router directs an instance of the MDS_SEND service transaction program to send the message over an LU 6.2 session (see "Routing Messages Between Nodes (Overview)" in topic 1.2.3.2.3 for more information). The router also delivers messages to application programs within a single node. The messages may be incoming traffic from other nodes or intra-node traffic between local application programs.

For more information see "Routing Messages to Application Programs (Overview)" in topic 1.2.3.2.4.

MDS Transaction Programs:

The MDS transaction programs provide the interface to presentation services for moving messages over sessions between nodes.

The following transaction programs are defined:

MDS_SEND Transaction Program:

An instance of this transaction program is started when the MDS router has one or more messages to send to another node. This transaction program also provides compatibility with back-level nodes. In the context of MDS, a *back-level node* is an APPN node, not at the current level of SNA, that implements a particular method for transporting MS data over LU-LU and CP-CP sessions that differs from MDS.

MDS_RECEIVE Transaction Program:

An instance of this transaction program is attached by the control point to receive an incoming message from another node. When the incoming message is received, it is passed to the MDS router.

MDS_SON Transaction Program:

The Session-Outage Notification (SON) transaction program is started by the LU when a session outage is detected for the LU-LU sessions used by MDS. The transaction program informs the MDS router so that appropriate error notifications can be sent to affected MS application programs.

The MDS router receives similar outage notifications via signals from control point session services when CP-CP sessions are lost.

| **MSU_HANDLER Transaction Program**:

| An instance of this transaction program is started by the LU to receive an
| incoming message from a back-level node. When the incoming message is
| received, it is passed to the MDS router. In the context of MDS, a
| *back-level node* is an APPN node, not at the current level of SNA, that
| implements a particular method for transporting MS data over LU-LU and
| CP-CP sessions that differs from MDS.

| **MDS_HP_SEND Transaction Program (optional)**:

| An instance of this transaction program is started when the MDS router has
| one or more messages to send over user-mode sessions, as described in
| MULTIPLE_DOMAIN_SUPPORT Optional Subset 1 (High-Performance Option).

| **MDS_HP_RECEIVE Transaction Program (optional)**:

| An instance of this transaction program is started by the LU to receive an
| incoming message over a user-mode session, as described in
| MULTIPLE_DOMAIN_SUPPORT Optional Subset 1 (High-Performance Option).

1.2.3.2.2 Formats for Multiple-Domain Support

```
|                  Multiple-Domain Support Message Unit GDS variable (MDS-MU)              |
| _____________________________________ _________ _________//__________                    |
| |      MDS Routing Information GDS variable      || Agent Unit ||    Application GDS    | |
| |_____________________ _____________________ _____|| of Work    ||       variable        | |
| | Origin Location SV |  Dest Location SV  | Flags || Correlator ||  (For example,        | |
| |_____ _____ _____|_____ _____ _____|  SV   || GDS        ||    CP-MSU or SNACR)    | |
| | NETID | NAU | Appl | NETID | NAU | Appl |       || variable   ||                    | |
| |_______|_____|_____|_______|_____|_____|_____||___________||_________//__________| |
| |_______________ multiple-domain support header _______________ÿ|                         |
|_____________________________________________________________________________________|
|                                                                  |                     | |
|                                                                  |__ 31743 bytes max.__ÿ| |
|                                                                                          |
|_____________________________________ always ° 32767 bytes _____________________________ÿ|
```

Figure 2-16. Structure of the Multiple-Domain Support Message Unit (MDS-MU)

Figure 2-16 is a conceptual view of the format used by multiple-domain support, the MDS-MU (X'1310') GDS variable. The MDS-MU contains:

° Multiple-domain support header

The header is not a GDS variable itself, but consists of the following GDS variables:

- MDS Routing Information (X'1311') GDS variable

- Agent Unit-of-Work Correlator (X'1549') GDS variable

° MS application program data

The MS application program data is a GDS variable supplied by the MS application program. It may be a CP-MSU (X'1212') GDS variable, an SNA Condition Report (X'1532') GDS variable, or some other GDS variable.

MDS-MU Message Size:

The maximum size permitted for the application program GDS variable in the MDS-MU is 31743 (X'7BFF') bytes. This restriction is imposed to simplify implementation design, so that the entire MDS-MU (MDS header and application program GDS variable) will always be ° 32767 (X'7FFF') in length.

The high-order bit (bit 0) of a GDS variable length field is not part of the length, but is a length continuation (or not-last segment) indicator. This capability to segment large data records into multiple logical records is described in *SNA LU 6.2 Reference: Peer Protocols*. MDS, however, does not support this function for the MDS-MU. For MDS-MUs, the length continuation indicator is always set to 0.

MDS Header Structure:

The MDS Header is the information used for routing between MS application programs. The MDS header is composed of two GDS variables:

° MDS Routing Information (X'1311') GDS variable

° Agent Unit-of-Work Correlator (X'1549') GDS variable

Each of these structures will be described in the following sections; see *SNA Formats* for the actual encodings.

MDS Routing Information (X'1311') GDS Variable:

This GDS variable contains the data required for MDS routing, consisting of the following structures:

° Origin Location Name (X'81') MS subvector

° Destination Location Name (X'82') MS subvector

° Flags (X'90') MS subvector

The contents of these subvectors is described in the following sections:

Origin Location Name (X'81') Subvector

NETID (X'01') Subfield

The is the network ID that qualifies the NAU name.

NAU Name (X'02') Subfield

This is the unqualified name of an SNA network accessible unit.

MS application program name (X'03') Subfield

This is an MS application program name, either a 4-byte architecturally-defined name (see Appendix D, "MS Transaction Program Names" in topic D.0) or a 1- to 8-byte installation-defined name.

Destination Location Name (X'82') Subvector

NETID (X'01') Subfield

The is the network ID that qualifies the NAU name.

NAU Name (X'02') Subfield

This is the unqualified name of an SNA network accessible unit.

MS application program name (X'03') Subfield

This is an MS application program name, either a 4-byte architecturally-defined name (see Appendix D, "MS Transaction Program Names" in topic D.0) or a 1- to 8-byte installation-defined name.

Flags (X'90') Subvector

This subvector contains the following indicators:

Message type

- MDS request
- MDS reply
- MDS error message

These message types characterize the messages *from the perspective of the MDS router*. This may or may not correspond to their characterization at the application program level.

Table 2-3 summarizes the relationship between MDS message types and application program-level flows. Since MDS's responsibilities are exactly the same for, e.g., an application program-level request with reply and an unsolicited application program-level data flow with acknowledgment, the difference between these two application program-level flows need not be made known to MDS. In this way the design for MDS can be kept simpler, while still addressing all of the application program-level transport requirements.

Table 2-3. Settings of MDS Message Type, First and Last MDS Message Flags

Application Program-Level Flow	MDS Message Type	First MDS Message Flag	Last MDS Message Flag
° request without reply ° unsolicited data without acknowledgment, e.g., Alert	MDS request	1	1
° request with reply ° unsolicited data with acknowledgment	MDS request	1	0
° reply (not last)	MDS reply	0	0
° last (or only) reply ° acknowledgment	MDS reply	0	1
° MDS error message	MDS error message	1	1

First MDS Message Indicator

This flag is set to 1 when the message is the first (or only) message for the unit of work. Otherwise it is set to 0.

Last MDS Message Indicator

This flag is set to 1 when the message is the last (or only) message for the unit of work. It is set to 0 when additional messages are expected. Table 2-3 shows how it is set in combination with the MDS message type.

Routing Verification Indicator

This flag is set to 0 when the MDS-MU was routed based on nonverified directory information. It is set to 1 when the MDS-MU was routed based on verified directory information.

Agent Unit-of-Work Correlator (X'1549') GDS variable:

This correlator provides a value unique across time for the MS application program that assigns it, allowing MDS-MUs containing requests, replies, and error messages to be correctly correlated both by MDS and the application programs. The contents of the correlator are described in Table 2-4.

Table 2-4. Contents of Agent Unit-of-Work Correlator (X'1549') GDS variable

Structure Name	GDS/SV/SF	Description of Contents
Agent Unit of Work Correlator	X'1549'	
Requester Location Name	X'01'	Name of node where the unit of work originated
Requester NETID	X'01'	NETID of originating node
NAU Name	X'02'	NAU name of originating node (1)
Requester Agent	X'04'	Name of originating MS application
Sequence number DTM	X'02'	Uniquely identifies UOW (see Table 2-5)

Table 2-5. Contents of Sequence Number Date Time (SEQNO DTM) Structure (part of Agent Unit-of-Work Correlator (X'1549') GDS variable)

Field Name	(byte offset)	Description of Contents
Length	(0)	Length of Seqno DTM structure
Key	(1)	Key (X'02')
SEQNO	(2-5)	Unique binary sequence value
Date	(6-9)	Date of UOW origination
Year	(6-7)	4-digit year, in binary
Month	(8)	2-digit month, in binary
Day	(9)	2-digit day of month, in binary
Time	(10-13)	Time of UOW origination
Hour	(10)	2-digit hour, in binary
Minute	(11)	2-digit minute, in binary
Second	(12)	2-digit second, in binary
Hundredth	(13)	2-digit hundredth of second, in binary

Time Flag	(14)	Indicates whether Date/Time is local or GMT
		X'E9' -- Date/time is GMT (no offset)
		X'4E' -- Date/time is local (ahead of GMT)
		X'60' -- Date/time is local (trails GMT)
Offset from GMT	(15-16)	Present when Date/Time is local
Hour offset	(15)	2-digit hour offset, in binary
Minute offset	(16)	2-digit minute offset, in binary

Example

Figure 2-17 shows an example of an Agent Unit-of-Work Correlator (X'1549') GDS variable that might be built by an MS application program. The offset from GMT, minus 4 hours, is that for EDT.

```
 __ _______ _/                                  Agent Unit-of-Work Correlator
|LL|X'1549'|
|__|_______|_/


    /_ _ _____ _/                               Requester Location
       |L|X'01'|                                   Name
    /_|_|_____|_/


                /_ _ _____ ________ _/          Reported on NETID
                   |L|X'01'|(Net ID)|
                /_|_|_____|________|_/


                /_ _ _____ _________ _/         Reported on Node ID
                   |L|X'02'|(CP name)|
                /_|_|_____|_________|_/


    /_ _ _____ __________________ _/            Requester Agent
       |L|X'04'|(application name)|
    /_|_|_____|__________________|_/


                    (seq. no.)  (1991)
    /_ _____ _____ ___________ _______ _/..     Sequence Number/
       |X'11'|X'02'|X'nnnnnnnn'|X'07C7'|            Date-Time (Seqno DTM)
    /_|_____|_____|___________|_______|_/..


                 (June 17, 23:59:59.99 (EST))
            ../_ _____ _____ _____ _____ _____ _____ _/..
                |X'06'|X'11'|X'17'|X'3B'|X'3B'|X'63'|
            ../_|_____|_____|_____|_____|_____|_____|_/..


              (-4:00 GMT offset)
            ../_ _____ _____ _____
                |X'60'|X'04'|X'00'|
            ../_|_____|_____|_____|
```

Figure 2-17. Example of the Agent Unit-of-Work Correlator (X'1549') GDS variable. Local time is June 17, 1991, 23:59:59.99 and offset from GMT is -4:00 (i.e., EDT). GMT is thus understood to be June 18, 1991, 03:59:59.99.

The MDS Error Message:

The MDS error message is the vehicle for reporting errors detected by an MDS_ROUTER at any point along the path of a transaction. It is also used by MS application programs under certain circumstances.Figure 2-18 shows a high-level decomposition of an MDS error message.

Note: The MDS error message is not a unique GDS variable. It is an MDS-MU with a message type of MDS error message. It always contains the SNA Condition Report (X'1532') GDS variable as its application program GDS variable.

```
| ____________________________________________________________________________ |
| |                                                                          ° |
| | | MDS Routing Info | UOW Correlator | SNA Condition Report |     MDS-MU    ° |
| | |   (X'1311') GDS  |  (X'1549') GDS  |     (X'1532') GDS    | (X'1310') GDS ° |
```

```
| | |     variable     |    variable    |        variable        |    variable   ° |
| | |__________________|________________|________________________|°°°°°°°°°°°°°°°° |
| |                  MDS-MU (X'1310') GDS                                          |
| |________________________________________________________________________________|
```

Figure 2-18. The MDS Error Message. The returned MDS-MU imbedded within the MDS Error Message is present only for a routing error associated with nonverified directory information.

An MDS error message is created by MULTIPLE_DOMAIN_SUPPORT for two categories of errors:

- Routing errors, in which an individual MDS-MU could not be delivered successfully to its destination application program. There is, however, no error report created for non-delivery of an MDS error message.

- Transaction failures, where the error pertains not to an individual MDS-MU, but to the sequence of such message units that comprise a transaction. For example, it is a transaction failure if MDS_ROUTER is able to determine, from either the receipt of session outage notification or the expiration of a timer, that no reply is forthcoming for a request that expected one.

The MDS error message has the following characteristics:

- It is sent either from MDS_ROUTER in the node that detected the error, or from one of the communicating MS application programs.

- The Agent Unit-of-Work Correlator (X'1549') GDS variable in the MDS header identifies the MDS-MU which could not be delivered or the transaction that failed.

- The application program data is an SNA Condition Report (X'1532') GDS variable, which carries an SNA Report Code (also known as SNA Sense Data) identifying the precise error that was detected. An MDS error message created by MDS also identifies the NAU and application program names for the other application program that was involved in the transaction (not the NAU and MS application program which is the destination of the MDS error message). The NAU and MS application program are identified in the SNA Condition Report (X'1532') GDS variable because the origin of the MDS error message is the MDS_ROUTER, not the partner application program.

- For routing errors associated with nonverified directory information, the undelivered MDS-MU is returned within the MDS error message. This is required so that the routing node can retry sending the message after directory information has been updated.

Table 2-6 shows the contents of an MDS error message created by an MDS_ROUTER:

Table 2-6. Contents of an MDS Error Message Created by MDS_ROUTER

Structure Name	GDS/SV/SF	Description of Contents
MDS-MU	X'1310'	
MDS Routing Information	X'1311'	
Origin Location Name	X'81'	
NETID	X'01'	NETID of reporting node
NAU Name	X'02'	NAU name of reporting node
Application ID	X'03'	X'23F0F1F0' (MDS_ROUTER)
Destination Location Name	X'82'	Destination for the MDS error message
NETID	X'01'	NETID of destination node
NAU Name	X'02'	NAU name of destination node
Application ID	X'03'	Destination MS application program (see Note)
Flags	X'90'	

MDS Message Type	__	MDS error message (X'02')
First MDS Message Indicator	__	1 _ first message for UOW
Last MDS Message Indicator	__	1 _ last message for UOW
Routing verification indicator	__	Value set when MDS error message is routed
Agent Unit of Work Correlator	X'1549'	Correlator of failed transaction
SNA Condition Report	X'1532'	
SNA Report Code	X'7D'	Sense data indicating nature of error
Reported on Destination Prefix	X'08'	(delimiter)
Reported on Location Name	X'09'	Node being reported on
Reported on NETID	X'01'	NETID of node reported on
Reported on Node ID	X'02'	NAU name of node reported on
Reported on Destination Suffix	X'0B'	(delimiter)
Reported on Agent	X'04'	MS application name reported on
MDS-MU	X'1310'	Returned MDS-MU (see Note)

Note:

For SNA Report code X'08A80009', the destination MS application program name is X'23F0F1F0' (MDS_ROUTER), and the returned MDS-MU is present. Otherwise, the returned MDS-MU is *never* present.

This extended format of an MDS error message is exchanged only between MDS_ROUTERs, and is never received by MS application programs.

The full SNA Condition Report (X'1532') GDS variable is documented in *SNA Formats*; for the MDS error message, however, only a relatively small subset of the encodings defined for this GDS variable are used.

Application Program-Level Error Reporting:

The MDS error message is not the normal vehicle for reporting application program-detected errors. These errors, e.g.:

- Command reject
- Parsing exception
- Function not supported

are reporting via application program-defined techniques (such as reply major vectors).

However, there are circumstances under which an MS application program must be able to terminate an outstanding MDS transaction unconditionally. For example, an MS application program may start a timer when an MDS request is sent to another MS application program. If no reply has been received when the timeout period has elapsed, the sending MS application program may conclude that something is wrong with the destination application program, causing it not to respond. Since the sending MS application program is not going to wait for the reply any longer, it must terminate the outstanding MDS transaction with an MDS error message.

The SNA Condition Report (X'1532') GDS variable in this MDS error message contains just the SNA Report Code (containing the sense data). The Reported On Location Name and Reported on Agent structures are not used, since the partner application program is fully identified in the destination of the MDS error message.

The format for an MDS error message sent by an MS application program is shown in Table 2-7.

Table 2-7. Contents of an MDS Error Message generated by an MS application

Structure Name	GDS/SV/SF	Description of Contents
MDS-MU	X'1310'	
MDS Routing Information	X'1311'	
Origin Location Name	X'81'	
NETID	X'01'	NETID of reporting node
NAU Name	X'02'	NAU name of reporting node
Application ID	X'03'	Name of reporting MS application
Dest. Location Name	X'82'	Destination for the MDS error message
NETID	X'01'	NETID of destination node
NAU Name	X'02'	NAU name of destination node
Application ID	X'03'	Name of destination MS application
Flags	X'90'	
MDS message type	__	MDS error message (X'02')
First MDS Message Indicator	__	1 _ first message for UOW
Last MDS message indicator	__	1 _ last message for UOW
Routing verification indicator	__	Value set when MDS error message is routed
Agent Unit of Work Correlator	X'1549'	Correlator of terminated transaction
SNA Condition Report	X'1532'	
SNA Report Code	X'7D'	Sense data indicating nature of error

(1) The *NAU* (network accessible unit) name is the CP or LU of the node.

1.2.3.2.3 Routing Messages Between Nodes (Overview)

```
 ________
|        |_____________________ LU-LU _____________________________
|   NN   |                     (SNASVCMG)                          |
|        |_______ LU-LU _______                                    |
|________|       (SNASVCMG)     |                                  |
   (FP)                         |                                  |

 ________                  ________                          ________
|        |                |        |_____ CP-CP _____ÿ|        |
|   NN   |____ LU-LU _____ÿ|   NN   |     (CPSVCMG)     |   EN   |
```

```
      |        |    (SNASVCMG)    |        |__              |        |
      |________|                  |________|  |              |________|
         (FP)                      (server)   |               ________
                                              |              |        |
                                              |__ CP-CP _____ÿ|   EN   |
                                                (CPSVCMG)     |        |
                                                              |________|
```

Figure 2-19. Sessions Used by Multiple-Domain Support in an APPN Network. MDS default routing uses LU-LU sessions (mode SNASVCMG) between network nodes (including focal points) and CP-CP sessions (mode CPSVCMG) between network node servers and their client end nodes. An SNASVCMG LU-LU session directly from a focal point to an end node may also be used. All the LU-LU and CP-CP sessions use LU 6.2 protocol.

Figure 2-19 shows the sessions used by MULTIPLE_DOMAIN_SUPPORT for default routing in an APPN network. MDS provides the benefits of session concentration for end nodes having multiple focal points (for different MS categories), by allowing them to pass all their MS traffic over a single session to their network node server. This session concentration also benefits nodes that support focal point application programs, which do not have the burden of direct session communication with every end node in a network. Focal point start-up overhead is thus significantly lightened.

Attempting to concentrate sessions further among network nodes would introduce more complexity to the routing mechanism. Additional session concentration would also increase the overall network impact of MS traffic, since transaction routing of messages is significantly more expensive in CPU cycles than direct session traffic.

```
     (FP)
|  ________                                                  ________
| |        |_____________________ LU-LU ____________________________ÿ|        |
| |        |                     (SNASVCMG)                          |        |
| | CP/LU  |_____________________ LU-LU ____________________         | CP/LU  |
| |        |                     (SNASVCMG)                 |        |        |
| |        |________ LU-LU ______                           |        |        |
| |________|        (SNASVCMG)    |                        |        |________|
       "                          |                        |
       |                          |                        |
       |                          |                        |       Subarea
    LU-LU                         |                        |       Network
  (SNASVCMG)                      |                        |
°°°°°|°°°°°°°°°°°°°°°°°°°°°°°°°°°|°°°°°°°°°°°°°°°°°°°°°°°|°°°Boundary function°°°
       |                          |                        |
       |                          |                        |       APPN
                                                           Network
  ________                  ________                 ________
 |        |                |        |               |        |
 |   NN   |____ LU-LU _____ÿ|   NN   |___ CP-CP ___ÿ|   EN   |
 |        |    (SNASVCMG)   |        |   (CPSVCMG)   |        |
 |________|                 |________|               |________|
    (FP)                     (server)
```

Figure 2-20. Sessions Used by Multiple-Domain Support in a Mixed Network. A "mixed" network in this context includes both APPN and subarea components. Connectivity across the subarea boundary function requires the use of LU-LU sessions.

Figure 2-20 illustrates the sessions used by MDS in a "mixed" network (containing both APPN and subarea components). Each of the nodes in the illustration contains an implementation of CPMS, but for purposes of connectivity, these may be represented in the network as an LU as well as a control point. The SNASVCMG mode name for the sessions shown is defined between LUs. As an option, the LU names may coincide with the CP names, but in this case the names are known (or registered) in both ways (as LUs and CPs) to allow them to participate in both CP-CP and LU-LU sessions. CPs (either network nodes or end nodes) in the APPN network may also appear as LUs through the subarea boundary function.

| *Perspective of Network Node*:

| When an application program in a network node has a message to send to an
| application program in another node, MDS in the network node asks control
| point directory services to locate the target. Directory services returns
| a description of the target in relationship to its network node server.
| Thus, if the target happened to be an end node, directory services would
| return: (2)

```
|          NNCP <cp name> ENCP <cp name>
```

| In this case, MDS would route the messages as shown in Figure 2-21. Note
| that in this and subsequent figures, both the MDS and application program
| message types are identified for each segment of the flow. The first set
| of square brackets indicates the information carried in the MDS header:

the MDS message type ([MDS req] or [MDS rep]), the setting of the first MDS message flag (first or ¬first), and the setting of the last MDS message flag (last or ¬last). The second set of square brackets indicates the application program message type ([appl req.], [appl unsol], [appl ack], and [appl rep.]). Brackets after the colon, e.g., [appl req.], indicates an application program-level request.

```
|      NN                                  NN                                  EN
| (origin appl)                        (server)                          (target appl)
|      .                                   .                                   .
|      . [MDS req, first,¬last] [appl req.] .                                 .
|      o_____________________________________  [MDS req, first,¬last] [appl req.] .
|      .                                   |_____________________________________ÿo
|      .                                   . [MDS rep,¬first, last] [appl rep.] |
|      . [MDS rep,¬first, last] [appl rep.]  _____________________________________|
|      o_____________________________________|                                   .
|      .                                   .                                   .
|      .                                   .                                   .
|      |_______________ LU-LU _______________ÿ|_________________ CP-CP _____________ÿ|
|                     (SNASVCMG)                                (CPSVCMG)
```

Figure 2-21. Normal Request/Reply Routing from a Network Node to an End Node. Messages between the requesting application program in the network node and the target application program in the end node are routed through the network node server.

If the target were a network node, directory services would return:

```
|          NNCP <cp name>
```

In this case, MDS would route the messages as shown in Figure 2-22.

```
|      NN                                  NN
| (origin appl)                       (target appl)
|      .                                   .
|      . [MDS req, first,¬last] [appl req.] .
|      o_____________________________________ÿo
|      . [MDS rep,¬first, last] [appl rep.] |
|      o_____________________________________|
|      .                                   .
|      .                                   .
|      |_______________ LU-LU _______________ÿ|
|                     (SNASVCMG)
```

Figure 2-22. Normal Routing between Network Nodes for Request/Reply. The network nodes communicate directly.

By always routing directly between network nodes and through a network node server for end nodes, MDS is able to use directory services information exclusively to make all routing decisions. No additional routing tables are needed.

Even with this FP-NN-EN routing topology in place, it is possible to communicate directly between a focal point and an end node. For example, if a particular end node happens to be a large data host that produces a significant volume of MS traffic, that end node can be defined explicitly in a focal point's SOC. MS application programs at the focal point will tell MDS to communicate directly with the end node, thus alleviating the potentially high transaction routing burden in the network node server. Or direct (i.e., one session hop only) routing might be required by a particular MS application program because of expected high message rates. This would be an application program design point, not a network installation policy decision. (No MS application programs currently in the MS architecture have this requirement, but it is anticipated as a future need.) In either case, MDS will treat the end node the same as a network node, and will use a SNASVCMG session directly to it (see Figure 2-23).

```
|      NN                                  EN
| (origin appl)                       (target appl)
|      .                                   .
|      . [MDS req, first,¬last] [appl req.] .
|      o_____________________________________ÿo
|      . [MDS rep,¬first, last] [appl rep.] |
|      o_____________________________________|
|      .                                   .
|      .                                   .
|      |_______________ LU-LU _______________ÿ|
|                     (SNASVCMG)
```

Figure 2-23. Direct LU-LU Routing to an End Node for Request/Reply. MDS in the network node, when direct routing is specified, communicates with the end node via an LU-LU session, bypassing the CP-CP session between the end node and its network node server. This is true whether the end node is a client of the origin network node server or of a different network node server.

In addition to request/reply flows, focal point application programs may

| also send requests for which no replies are expected. These flows are
| illustrated in Figure 2-24 (for a request directed to an end node),
| Figure 2-25 (for a request directed to a network node), and Figure 2-26
| (for a request directed to an end node defined explicitly in the focal
| point's sphere of control).

```
|      NN                                  NN                                  EN
| (origin appl)                        (server)                        (target appl)
|      .                                   .                                   .
|      . [MDS req, first, last] [appl req.] .                                 .
|      o_____________________________________  [MDS req, first, last] [appl req.] .
|      .                                  |_____________________________________ÿo
|      .                                   .                                   .
|      .                                   .                                   .
|      |_______________ LU-LU _______________ÿ|_________________ CP-CP _____________ÿ|
|                     (SNASVCMG)                              (CPSVCMG)
```

| Figure 2-24. Normal Network Node to End Node Routing for Request without
| Reply. The network node on the left communicates indirectly
| with the target end node through its network node server.

```
|      NN                                      NN
| (requesting appl)                      (target appl)
|      .                                        .
|      .   [MDS req, first, last] [appl req.]   .
|      o_________________________________________o
|      .                                        .
|      .                                        .
|      |_________________ LU-LU _________________ÿ|
|                       (SNASVCMG)
```

| Figure 2-25. Normal Network Node to Network Node Routing for Request
| without Reply

```
|      NN                                      EN
| (target appl)                        (requesting appl)
|      .                                        .
|      .   [MDS req, first, last] [appl req.]   .
|      o_________________________________________o
|      .                                        .
|      .                                        .
|      |_________________ LU-LU _________________ÿ|
|                       (SNASVCMG)
```

| Figure 2-26. Direct Routing from End Node to a Network Node for a Request
| without Reply. When direct routing is specified in an end
| node, an LU-LU session to the destination will be used, not
| routing through the network node server via the CP-CP
| session.

| *Perspective of Network Node Server*:

| MDS in a network node server must maintain awareness of all client end
| nodes for the purposes of message routing. Since this awareness is
| equivalent to the end node awareness that session services and directory
| services in the network node server must maintain, MDS can use the same
| data provided to those components. This data consists of:

| CPSVCMG session awareness for APPN end nodes

| CP Capabilities messages received from client end nodes

| MDS in a network node accepts MDS-MUs from its client end nodes over the
| CPSVCMG sessions. Messages from any other node must be received over
| SNASVCMG sessions, not CPSVCMG sessions. If an MDS-MU is received from
| another network node on a CPSVCMG session, it will be rejected.

| Network node processing will be described for the following flows:

| ° Outbound (toward client end nodes)

| - Request/reply

| - Unsolicited data

| ° Inbound (from client end nodes)

| - Request/reply

| - Unsolicited data

| **Outbound MDS Requests:**

| An outbound MDS request is one for which the destination is a client end
| node in the domain of the network node. These are requests at the
| application program level as well. This flow is illustrated in
| Figure 2-21 (for an outbound request/reply) and in Figure 2-24 (for an
| outbound request without reply).

| **Inbound MDS Requests:**

| These are messages that originate in a client end node. At the
| application program level, they are messages containing unsolicited data.

| MDS in the network node server routes these messages in the same way it
| would route messages from an MS application program in its own node. This
| is illustrated for an MDS request/reply in Figure 2-27. As the figure
| indicates, this is an application program-level flow of unsolicited data
| for which an acknowledgment is being requested.

```
|      NN                                  NN                                  EN
| (target appl)                        (server)                            (origin appl)
|      .                                   .                                   .
|      .                                   . [MDS req, first,¬last] [appl unsol] .
|      . [MDS req, first,¬last] [appl unsol]  _____________________________________o
|      o_________________________________|                                     .
|      | [MDS rep,¬first, last] [appl ack]   .                                 .
|      |_____________________________________   [MDS rep,¬first, last] [appl ack]  .
|      .                                   |_____________________________________ÿo
|      .                                   .                                   .
|      .                                   .                                   .
|      .                                   .                                   .
|      |_______________ LU-LU _______________ÿ|_________________ CP-CP _____________ÿ|
|                     (SNASVCMG)                            (CPSVCMG)
```

| Figure 2-27. Default End Node to Network Routing for MDS Request and
| Reply. The MDS reply transports the application-level
| positive acknowledgment.

| If the message had been an MDS request without reply, the flow would look
| like that shown in Figure 2-28. At the application program level, this is
| unsolicited data for which an acknowledgment is not being requested.

```
|      NN                                  NN                                  EN
| (target appl)                        (server)                          (origin appl)
|      .                                   .                                   .
|      .                                   . [MDS req, first, last] [appl unsol] .
|      . [MDS req, first, last] [appl unsol]  _____________________________________o
|      o_________________________________|                                     .
|      .                                   .                                   .
|      .                                   .                                   .
|      |_______________ LU-LU _______________ÿ|_________________ CP-CP _____________ÿ|
|                     (SNASVCMG)                            (CPSVCMG)
```

| Figure 2-28. Default End Node to Network Node Routing for an MDS Request
| without Reply

| If the target for the record happened to be an end node, directory
| services would return:

```
|          NNCP <cp name> ENCP <cp name>
```

| In this case, MDS would route the message as shown in Figure 2-29.

```
|   EN                             NN                              NN                              EN
| (target appl)                 (server)                        (server)                      (origin appl)
| .                                .                               .                                .
| .                                .                               .[MDS req, first, last] [appl req].
| .                                .[MDS req, first, last] [appl req] ___________________________________o
| .[MDS req, first, last] [appl req] ___________________________________|                                .
| o_________________________________|                                .                                .
| .                                .                               .                                .
| |_____________ CP-CP _____________ÿ|_____________ LU-LU _____________ÿ|_____________ CP-CP ____________ÿ|
|              (CPSVCMG)                         (SNASVCMG)                          (CPSVCMG)
```

| Figure 2-29. Default Routing of End Node to End Node Request without
| Reply. Each end node communicates indirectly through its
| network node server via its CP-CP sessions. Network node to
| network node communication is always direct, via LU-LU
| sessions.

| A request/reply flow in that same case is show in Figure 2-30.

```
|   EN                              NN                              NN                              EN
| (target appl)                  (server)                        (server)                      (origin appl)
|  .                                 .                               .                               .
|  .                                 .                               .[MDS req, first,¬last] [appl req].
|  .                                 .[MDS req, first,¬last] [appl req] _________________________________o
|  .[MDS req, first,¬last] [appl req] _________________________________|                               .
|  o_________________________________|                              .                               .
|  |[MDS rep,¬first, last] [appl rep].                               .                               .
|  |_________________________________ [MDS rep,¬first, last] [appl rep].                               .
|  .                                 |_________________________________ [MDS rep,¬first, last] [appl rep].
|  .                                 .                               |_________________________________ÿo
|  .                                 .                               .                               .
|  .                                 .                               .                               .
|  |____________ CP-CP ____________ÿ|____________ LU-LU ____________ÿ|_____________ CP-CP ___________ÿ|
|               (CPSVCMG)                         (SNASVCMG)                        (CPSVCMG)
```

```
| Figure  2-30. Default Routing of Request/Reply from End Node to End Node.
|               Each end node communicates indirectly through its network
|               node server via its CP-CP sessins.  Network node to network
|               node communication is always direct, via LU-LU sessions.
```

| *End Node Perspective*:

```
| An MS application program in an end node communicates with MS application
| programs in other nodes in one of two ways:

| 1.  Indirectly via the MDS routing at the network node server

|     The first option is the default routing provided by MDS, which is
|     described in previous sections.

| 2.  Directly via a SNASVCMG session.

|     The second option is used when the end node communicates directly with
|     a focal point because of policy set for the network installation
|     (i.e., end node in SOC) or because of application program-unique
|     requirements.
```

| **Default Routing**:

```
| MDS in the end node performs default routing by sending and receiving all
| MS messages over the CPSVCMG session to the network node server.
```

| (2) Note that this is the conceptual information returned by | directory services, not the actual data. Refer to | "MULTIPLE_DOMAIN_SUPPORT Function Set" in topic 2.9.2 for a | description of the precise information provided by directory | services.

1.2.3.2.4 Routing Messages to Application Programs (Overview)

```
Multiple-domain support uses application program names for routing within
a node.  The application program routing is accomplished in the following
manner:

1.  MS application program names follow the LU 6.2 TP naming technique,
    which provides for 4-byte architecturally-defined values or 1-to
    8-character installation-defined names (LU 6.2 base support).

2.  The MS architecture defines application program names to correspond
    with MS function sets (see Appendix D, "MS Transaction Program Names"
    in topic D.0).

In order for communication to take place between two MS application
programs, the names by which the application programs are known to MDS
must be understood.  That is to say, the application program that
initiates a unit of work (e.g., sends a request message) must know the
correct name of the target application program so that the message can be
correctly routed.  This understanding of application program names is
achieved in either of the following ways:

°   Implicitly

    The application program names are predefined (i.e., "hardcoded") when
```

the functions are implemented. For architecturally-defined functions which rely upon predefined application program names, the names are registered values (refer to Appendix D, "MS Transaction Program Names" in topic D.0 for a list of these names).

° Explicitly

The application program names are not understood when the functions are implemented. Instead, an application program becomes aware of the actual name of a target application program via some initialization exchange. For functions that use the architecturally-defined focal point mechanism, the actual name of a target application program is learned via an exchange of MS Capabilities major vectors between the focal point and entry point. (Refer to "Properties of Focal Points" in topic 1.2.4.2 for more information about MS Capabilities.) This exchange provides the name of the focal point application program to the entry point application program; it does not however, provide the name of the entry point application program to the focal point application program. If the focal point application program needs to send requests to the entry point, then the entry point name will have to be known *implicitly*, unless the focal point application program is triggered by the receipt of an unsolicited message from the entry point application program.

```
 ____________________________________      ____________________________________
| °°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°° |    | °°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°° |
| °  _______  _______  _______      ° |    | °  _______  _______          ° |
| ° |ALERT_ ||MS_CAPS||EP_    |... ° |    | ° |EP_    ||MS_CAPS|...       ° |
| ° |NETOP  ||       ||ALERT  |    ° |    | ° |ALERT  ||       |          ° |
| ° |_______||_______||_______|    ° |    | ° |_______||_______|          ° |
| °    "                             ° |    | °    |(1)                    ° |
| °    |(5)                          ° |    | °                            ° |
| °  _______________________         ° |    | °  _______________________   ° |
| ° |       MDS_ROUTER        |      ° |    | ° |       MDS_ROUTER        | ° |
| ° |_________________________|      ° |    | ° |_________________________| ° |
| °          "                       ° |    | °          |(2)               ° |
| °          |(4)                    ° |    | °                            ° |
| °   _______________               ° |    | °    ____________            ° |
| °  | MDS_RECEIVE TP |              ° |    | °   | MDS_SEND TP |          ° |
| °  |________________|       CPMS ° |    | °   |_____________|      CPMS ° |
| °°°°°°°°°° " °°°°°°°°°°°°°°°°°°°°°° |    | °°°°°°°°°°|°°°°°°°°°°°°°°°°°°° |
|            |     Focal Point node  |    |            |  Entry Point node  |
|____________|_______________________|    |____________|____________________|
             |                                         |
             |                                         |(3)
             |_________________________________________|
```

Figure 2-31. Routing by MS Application names

Figure 2-31 illustrates application program routing for the case of an Alert flowing from the EP_ALERT function set in an entry point node to the ALERT_NETOP function set in a focal point node. MS_CAPS in each node is the function set that performs MS Capabilities exchanges to initialize the focal point to entry point relationship (refer to "Properties of Focal Points" in topic 1.2.4.2 for more information). The numbered items below refer to the numbers in that figure.

1. EP_ALERT creates an Alert and then builds an MDS-MU to transport it. The destination NAU in the MDS header is the focal point node, and the destination application program is ALERT_NETOP.

   This MDS-MU is passed to the MDS router. (3)

2. The MDS router sees that the destination NAU is not its own node, so it passes the MDS-MU to the MDS_SEND TP.

3. MDS_SEND sends the MDS-MU over an LU 6.2 conversation to the destination NAU. It is received by MDS_RECEIVE in the destination node. (This example assumes the simplest case for routing between the two nodes.)

4. MDS_RECEIVE passes the MDS-MU to the MDS Router.

5. The MDS router sees that the destination is this node, so it examines the destination application program name (ALERT_NETOP), comparing it to MS application programs that are known in this node.

   The MDS-MU is then passed to the MS application program.

---

(3) Note that the MDS router is a single persistent process within the management services component of the node, while the MDS service transaction programs are non-persistent.

1.2.3.3 Transport of Bulk Management Services Data

This section describes how management services uses SNA/DS for transport of requests, reports, and bulk data.

Subtopics:

- 1.2.3.3.1 Introduction to SNA/DS and SNA/FS
- 1.2.3.3.2 The SNA/FS Server
- 1.2.3.3.3 Formats for SNA/DS
- 1.2.3.3.4 How Management Services Uses SNA/FS
- 1.2.3.3.5 Choice of SNA/DS Roles, Electives, and Options
- 1.2.3.3.6 Security

1.2.3.3.1 Introduction to SNA/DS and SNA/FS

Refer to Figure 2-32.

```
 _________________
|                 |
|                 |
|     Agent       |
|       "         |
|       |         |
|_______|_________|_________________________ _______________
|       |                                  |               |
|                                          |               |     /==========\
|                                          |               |     |          |
|                                        _|_ÿ             |     | Storage  |
|                 SNA/DS                   |    Server     |=====|          |
|                                          |               |     |          |
|                                          |               |     \__________/
|                                          |               |
|                                          |               |
|__________________________________________|_______________|
```

Figure 2-32. SNA/DS View of Agent and Server

In SNA/DS, the *agent* is an application program transaction program that is using SNA/DS as a transport mechanism.

The *server*, also provided by the user of SNA/DS, is invoked by SNA/DS to handle staging and destaging of (typically large) files to the system storage facilities.

1.2.3.3.2 The SNA/FS Server

For network management, the server is involved in distributing large files, but not administering them. In fact, the server cannot distinguish between a network management file and, say, a job to be submitted for execution, or a file simply to be stored with no further processing. Such actions are the responsibility of the agent.

The building and parsing of the object handled by the server (the server object) for network management need be no different from that for other SNA/DS agents. For this reason, architecture has been developed for the server, called *SNA/File Services* (SNA/FS).

For a detailed description of SNA/File Services, refer to *SNA/File Services Reference*.

1.2.3.3.3 Formats for SNA/DS

```
 _____________________________________________________________________________
|                          SNA/DS message unit                                |
| _________ _________ ____________________ _______________________ _________  |
|| SNA/DS || SNA/DS  ||                    ||                     || SNA/DS | |
|| prefix ||   GDS   ||      SNA/DS        ||        SNA/DS        || suffix | |
||  GDS   ||variables||   agent object     ||     server object     ||  GDS   | |
||variable||  (e.g.  ||   GDS variable     ||      GDS variable     ||variable| |
||        || correl- ||                    ||                       ||        | |
||        ||  ator)  ||                    ||                       ||        | |
```

```
 ||________||_________||____________________||_______________________||________| |
 |                                                                                |
 |________________________________________________________________________________|
 |                     |                  |                                       |
 |                     |__ 32767 bytes __ÿ|                                       |
 |                     |      maximum     |                                       |
 |                                                                                |
 |_______________ length limited only by capacity of implementations ___________ÿ|
```

Figure 2-33. Structure of the SNA/DS Message Unit Used by SNA/MS. The SNA/DS message unit contains control information used by SNA/DS, and GDS variables containing application data such as a correlator, an agent object containing agent-to-agent instructions, and a server object containing file control information and a file.

Figure 2-33 is a conceptual view of the encoding used by SNA/DS, the message unit. The message unit contains the following SNA/DS GDS variables:

- SNA/DS prefix and suffix GDS variables to delimit the message unit.

- An SNA/DS agent object, present if agent-to-agent instructions are required by the MS application program. For example, an instruction to retrieve a file is carried in the agent object, but when the file is returned no agent object is necessary.

  There are two types of agent-to-agent instructions:

  1. SNA/MS commands or reports, contained within a CP-MSU; and

  2. SNA/FS commands or reports, contained with an SNA/FS GDS variable.

- An SNA/DS server object, present if the MS application program requires use of SNA/FS for a particular request. An MS application program may use SNA/DS to transport commands and replies that do not require bulk data.

  The server object always contains SNA/FS GDS variables containing file control information, but may or may not contain the file in addition. For example, a request to retrieve a file contains a server object with file control information only.

SNA/DS Message Size:

The maximum size permitted for the agent object is 32767 (X'7FFF') bytes. This is because it is to carry agent-to-agent instructions rather than bulk data. The size of the server object that does carry the bulk data is constrained only by implementation limits; therefore, so is the size of the message unit containing the server object.

1.2.3.3.4 How Management Services Uses SNA/FS

A user SNA/MS request is converted by a focal point or entry point into another kind of request, called a *command*, that it sends to one or more entry points or focal points. For example, the Retrieve request is converted to an SNA/FS *Transfer-to-Requester* command. A command is executed by the target node, and results in a *report* to the requester. Each report, associated with a request, is given to the requesting user when it is received by the requesting node. Reports are defined by each of the SNA components. An SNA/DS report is received if an error occurred in the distribution network. If distribution is successful, then either an SNA/FS report or an SNA/MS report is received. SNA/FS reports occur if the request was for file transfer only, or for SNA/MS requests that resulted in unsuccessful file transfer. If file transfer succeeds for an SNA/MS request, then SNA/MS reports indicate the success or failure of the SNA/MS request.

Commands and reports are carried in SNA/DS *message units* (MU)s.

Refer to "Management Services Formats" in topic 1.2.1.1 for a description of the MU format and a pictorial representation of the MU.

Each MU may contain the following items:

- A command or report, defined by either SNA/MS or SNA/FS, and contained in the agent object;

° SNA/FS control information contained in the server object; and

° A file, also contained in the server object.

An MU may identify, but not carry, a file, in which case a SNA/FS control information is present without a file. An example of this is a *Retrieve* request. On the other hand, a command or report need not be present. For example, a successful Retrieve request results in the return of a file and its control information, but no report.

Different from both user requests and focal point commands is the SNA/FS *server instruction* in the control information. The server instruction indicates to the SNA/FS server at the destination node how to manipulate the file into, or out of, the storage facilities. For example, a Transfer-to-Requester command flows with a *Fetch* server instruction, and the reply contains a *Create/Load-or-Replace* server instruction.

Example of Data Flow:

We shall follow a step-by-step sequence of events, starting with a typical example of a request, issued by a user at a focal point.

The SNA/FS capability is symmetric with respect to focal point and entry point, so this request could also originate at an entry point, in the case of a request that involved file transfer only (and not some additional action such as change management).

First, the request is issued by the user, and contains the agent unit-of-work correlator, SNA/FS control information (including the file name), and a list of destinations.

Network management request

```
          |
 ________|________
|        |        |           FOCAL POINT
|                |
|      SNA/MS     |
|                 |
|                 |
|_________________|_________________________ _______________
|                                          |               |
|                                          |               |     /==========\
|                                          |               |     |          |
|                                          |    SNA/FS     |     | Storage  |
|                 SNA/DS                   |    Server     |=====|          |
|                                          |               |     |          |
|                                          |               |     \__________/
|                                          |               |
|                                          |               |
|__________________________________________|_______________|
```

Figure 2-34. Example Flow: User Issues Request

Next, SNA/MS builds the agent object, containing the command to the entry point. The SNA/DS SEND_DISTRIBUTION verb is issued, specifying the agent object, the correlator, the destination list, the destination agent name (SNA/MS), the destination server name (SNA/FS), and the SNA/FS server parameters.

```
 _________________                  FOCAL POINT
|                 |
|                 |
|     SNA/MS      |   SNA/DS SEND_DISTRIBUTION verb
|       |         |
|       |         |
|_______|_________|_________________________ _______________
|       |                                  |               |
|                                          |               |     /==========\
|                                          |               |     |          |
|                                          |    SNA/FS     |     | Storage  |
|                 SNA/DS                   |    Server     |=====|          |
|                                          |               |     |          |
|                                          |               |     \__________/
|                                          |               |
|                                          |               |
|__________________________________________|_______________|
```

Figure 2-35. Example Flow: SNA/MS Builds Agent Object and Issues SEND_DISTRIBUTION

For each destination as appropriate, SNA/DS allocates a conversation, builds an SNA/DS MU with the help of the SNA/FS server, and sends the MU on that conversation. The MU is pictured here.

```
 _________________
|                 |
|                 |                    FOCAL POINT
|      SNA/MS     |
|                 |
|                 |
|_________________|_________________________ _____________
|                                          |             |
|                                          |             |     /==========\
|                                          |             |     |          |
|                                          |    SNA/FS   |     | Storage  |
|                 SNA/DS                   |    Server   |=====|          |
|                                          |             |     |          |
|                                          |             |     \__________/
|     |                                    |             |
|     |                                    |             |
|_____|____________________________________|_____________|
      |
 ____|
```

```
 _______ ______ _____ ______ _______ _______ ____ _______ ______________________ ____
|Prefix,|Origin|Date/|Corre_|Agent  |Server |Dest| Agent |    Server            |Suf_|
| flags | Name |Time |lator |TP name|TP name|Name| object|    object            |fix |
|_______|______|_____|______|_______|_______|____|_______|______________________|____|
                                SNA/DS message unit
```

Figure 2-36. Example Flow: SNA/DS Builds a Message Unit and Sends It to the Entry Point

At the destination node, SNA/DS receives and decodes the MU with the help of the SNA/FS server. The server interprets the SNA/FS control information and stores the file.

```
 _________________
|                 |
|                 |                    ENTRY POINT
|     SNA/MS      |
|                 |
|                 |
|_________________|_________________________ _____________
|                                          |             |
|                                          |             |     /==========\
|                                          |             |     |          |
|                                          |    SNA/FS   |     | Storage  |
|                  SNA/DS                  |    Server   |=====|          |
|                                          |           _|_____|_ÿ        |
|  ________________________________________|__ÿ          |     \__________/
|  |                                       |             |
|  |                                       |             |
|__|_______________________________________|_____________|
   |
   |
```

```
 _______ ______ _____ ______ _______ _______ ____ _______ ______________________ ____
|Prefix,|Origin|Date/|Corre_|Agent  |Server |Dest| Agent |    Server            |Suf_|
| flags | Name |Time |lator |TP name|TP name|Name| object|    object            |fix |
|_______|______|_____|______|_______|_______|____|_______|______________________|____|
                                SNA/DS message unit
```

Figure 2-37. Example Flow: SNA/DS at the Entry Point Receives the MU and Invokes the SNA/FS Server

The SNA/MS agent at the entry point issues RECEIVE_DISTRIBUTION and is returned parameters, including: the correlator value, the agent object, and the SNA/FS parameters.

```
 _________________
|                 |                    ENTRY POINT
|                 |
|     SNA/MS      |
|       "         |  SNA/DS RECEIVE_DISTRIBUTION verb
```

```
|       |         |
|_______|_________|_________________________ _____________
|       |                                  |             |
|       |                                  |             |     /==========\
|                                          |             |     |          |
|                                          |    SNA/FS   |     | Storage  |
|                  SNA/DS                  |    Server   |=====|          |
|                                          |             |     |          |
|                                          |             |     \__________/
|                                          |             |
|                                          |             |
|__________________________________________|_____________|
```

Figure 2-38. Example Flow: SNA/MS at the Entry Point is Returned RECEIVE_DISTRIBUTION Parameters

SNA/MS then parses the agent object, interprets the results of the SNA/FS server instruction, and executes the intended command.

The entry point returns (echoes) the correlator value supplied by the focal point.

The return flow is similar, except that at the focal point, a reply is given to the issuer of the request.

More detailed examples are provided in "EP_CHANGE_MGMT Function Set" in topic 2.10.4.

SNA/FS Commands and Server Instructions Used by SNA/MS:

Management services uses the following SNA/FS commands:

- TRANSFER_TO_REQUESTER - The agent in the target node is requested to return one or more files.
- REPORT_FS_ACTION - The other agent is requested to report whether or not its server executed the server instruction successfully.
- REPORTING_FS_ACTION - The other agent is being informed whether or not a server instruction was executed successfully.

The agent object is omitted in the reply to a TRANSFER_TO_REQUESTER.

Management services uses the following SNA/FS server instructions:

- CREATE/LOAD_OR_REPLACE - This requires the SNA/FS server to determine whether or not the file received already exists on its system. If it does, the server replaces it. If it doesn't the server creates a new file.
- CREATE/LOAD - This requires the SNA/FS server to accept the contents of the server object and store it in a new file.
- DELETE - This requires the SNA/FS server to locate the identified object and delete it.
- ENCODE_ONLY - This requires that the SNA/FS server encode (build) the server object and then pass the result to SNA/DS. The server object does not contain a file, just file control information, so no fetching is required.
- DECODE_ONLY - This requires that the SNA/FS server decode (parse) the identifier and then pass the result to the agent. The server object does not contain a file, just file control information, so no storing is required.
- FETCH - This requires the source server to fetch the files from storage.

Refer to Table 2-8 for a summary of SNA/FS commands and server instructions by role.

Table 2-8. Base SNA/FS Commands and Instructions by Role

Command	FP ÿ EP	EP ÿ FP	FP _ÿ FP	EP _ÿ EP

Base SNA/FS Commands				
omitted agent object	(Note)	YES	YES	YES
REPORT_FS_ACTION	YES	YES	YES	YES
REPORTING_FS_ACTION	YES	YES	YES	YES
TRANSFER_TO_REQUEST	R YES	(Note)	YES	YES
Base Server Instructions				
CREATE / LOAD_OR_REPLACE	YES	YES	YES	YES
DECODE_ONLY	YES	YES	YES	YES
DELETE	YES	YES	YES	YES
ENCODE_ONLY	--	--	--	--
FETCH	YES	YES	YES	YES
CREATE / LOAD	YES	YES	YES	YES

Note: In normal operation, a focal point will never send an MU omitting an agent object to an entry point unless the entry point issues TRANSFER_TO_REQUESTER. However, an entry point is expected to accept one should it occur.

The following SNA/FS intentions are taken by management services:

° EXECUTING - Used when executable files are sent to entry points.

° STORING - Used when any files are sent to a focal point. Changes, for example, are not executed at focal points, but are stored for distribution.

The following SNA/FS exception actions are requested by management services:

° ABEND - If an SNA/FS server exception condition arises, the server is not required to return the storage files to their original condition.

° BACKOUT - If an SNA/FS server exception condition arises, the server *will* attempt to return the storage files to their original condition.

Bulk Data Commands and Replies:

To simplify processing and error reporting, each agent object, if present, will carry only one command or reply. These are described either by SNA/FS or management services.

SNA/FS commands are determined by the request made by the user and are symmetric with respect to focal point and entry point implementations. These commands and their associated replies are carried in a FS_Agent_Request or FS_Agent_Report GDS variable within the agent object.

Management services commands are sent by a focal point to another focal point or to an entry point and are determined by the request made by the user. These commands and their associated replies are carried in a CP-MSU within the agent object.

A focal point is expected to provide the user with the capability of grouping requests into sequences with execution of subsequent requests conditioned on successful execution of prior requests. The user must be able to instruct the requesting node to begin execution of such groups of requests at a specific date and time.

Use of the SNA/FS Global File Name:

SNA/MS requires the use of *global* file names. That is, a file has a name that can be handled by each SNA/FS server in the network, and that can be recognized by each SNA/MS agent.

One of the important motivations for using SNA/FS is the requirement to minimize the implementation cost incurred by an SNA/MS focal point product to identify the files used by the wide variety of products in an SNA network. SNA/FS provides a global name for a file, consisting of a set of tokens. SNA/FS standardizes the number of tokens allowed (up to 10), the size of each token (up to 16 characters), total name length (65-n where n is the number of tokens), and the character set allowed for the token

values (a limited set of character graphics displayable on most types of displays: Coded Graphics Character Set ID 01134-00500).

SNA/MS supports two different implementations of the SNA/FS global name; *change-management-structured* and *enterprise-structured*.

Change Management Structured Names:

The format of change management structured names is defined by SNA/MS, and the SNA/FS architecture maintains a registration of values for the highest-order token. For example, MCODE is the registered value for change files containing microcode. SNA/MS maintains a registration of the values of some of the other tokens, delegating authority in some cases to other administrative organizations. For example, the IBM machine type is the second token for microcode. As the result, a focal point is required to implement only one input panel, say, for a user to identify microcode files for a potentially wide variety of types of target entry points.

Definitions of token values by SNA/MS were made to satisfy two general requirements. The first requirement is that each file must be uniquely identified. The second requirement is to provide the user with some idea of the type and identity of the change when displaying the name, or to allow an application program to process the token values. For example, the file name MCODE.9135.NA.PATCH.1234 indicates that the file contains a patch rather than an engineering change. This is needed both to uniquely identify the file, and also to provide useful information on display.

Token values are assigned to each type of file used by change management. Refer to the tables beginning on page C.0 for details about the tokens used by SNA/MS and associated rules defined for their use.

Enterprise Structured Names:

Unlike change management structured names, the individual token values (with the exception of the first token) for enterprise structured names are not predefined by SNA/MS. Rather, the enterprise that owns (i.e., is the naming authority for) the change object designs its global name. In order to ensure inter-enterprise uniqueness, SNA/MS requires that the high-order token of the name contain the naming authority's enterprise code contained in its structured network ID(s) as assigned and registered by IBM. Within enterprise code, the naming authority is responsible for uniquely naming the change object by assigning appropriate changed component token(s) and, optionally, *architecturally-defined subtree* and *version* tokens.

An *architecturally-defined subtree* is a set of architecturally-defined tokens that is determined by the presence of a special leading token called a *subtree identifier (STI)* token. The following STI tokens are architecturally defined and registered:

GRP Group object

REF Refresh object

UPD Update object

FIX Fix object

The presence of an STI token in a global name indicates the nature of the object and signifies special meaning for one or more of the tokens that follow it. The architecturally-defined subtree tokens associated with each STI, and the rules for their use, are described in the tables starting on page C.0.

In addition to the architecturally-defined subtree, one or more further qualifying or *version* tokens may also appear in the global name. Such a token may be used, for example, to distinguish between the French and German versions of a change object.

Using the naming constructs described above, a typical enterprise structured name might be MYFOOD.BUSINESS.MESSAGES:REF.12.SPANISH. This global name would describe the level-12 refresh for the Spanish version of the Messages subcomponent of the Business application program owned by the Myfood enterprise. The ":" in the displayed name indicates the presence of an STI token that introduces the "REF.12" architecturally-defined subtree. Use of the ":" as a subtree introducer is employed within this document, but as in all token separators, is *not* an architecturally-defined value.

Whether using change-management-structured or enterprise-structured SNA/FS global names it is advantageous for the user to create (or have provided by product developers) change files that can be installed on a large

number of entry points. This allows the fan-out feature of SNA/DS to be fully exploited, and reduces the user's effort. For example, files containing microcode can be designed to be applicable to many control units, whereas those containing customizing data are specific to individual control units.

Use of SNA/FS Application Program Intervention Exit:

Products may implement their SNA/FS server to invoke an application program exit based on file type. This allows closed protocol boundary implementations to perform agent processing before the first storage operation performed by the server. Some of the exception conditions recognized by change management are recognized in this context. The report codes are eventually returned from one agent to the other.

Use of SNA/FS Partial Name Processing: SNA/MS makes use of SNA/FS *partial name processing* for both retrieval and distribution. Partial name processing is used when the user wishes to specify only some of the file identification tokens (typically, the higher order ones). An example of this is when the latest version of a customizing data file is to be retrieved, but the user cannot remember the version number contained in the lowest identification token (and does not care what its value is).

On distribution, partial name processing may be needed to specify which file to destroy to make room for a new one, when entry point storage constraints arise. An important requirement addressed by the architecture is that destruction take place only when installation has been requested properly and pre-tests have been performed successfully. The complete identity of the file replaced is included in the report of successful installation.

1.2.3.3.5 Choice of SNA/DS Roles, Electives, and Options

Refer to *SNA/Distribution Services Reference*, for details about the SNA/DS options.

A node participating in SNA/File Services must support SNA/DS format set 2 with only the following features required:

1. Role: End-only

   Focal points and entry points that do not implement the intermediate routing function can be configured to communicate through SNA/DS intermediate nodes and obtain the benefits of SNA/DS.

2. Use of correlator (to carry Agent Unit-of-Work Correlator)

3. Integrity: High

4. Protocol boundary: closed

   A product can choose to implement the SNA/DS layer solely for the purposes of network management. In doing so, an open protocol boundary would be useful if other application programs will want to use it, but is not required.

5. Electives:

   - Receive-time enhancements

     Network management requires the capability to check more fields in the SNA/DS header than the minimum required (e.g. the destination DSU list).

   - Use of RGN

     RGN is needed to carry NETID.

In addition, the MU Operations function is recommended for attended nodes, because it provides the ability to list and purge distributions.

Because user names are not required for network management, certain optimizations are possible. In particular, a user directory and destination fan-out are *not* required.

1.2.3.3.6 Security

The following levels of security are available in LU 6.2 and SNA/DS to products implementing network management.

Refer to *SNA/Distribution Services Reference* and *SNA Transaction Programmer's Reference Manual for LU Type 6.2*, for details.

1. LU 6.2 session-level security (LU-LU verification)

2. LU 6.2 conversation-level security verification

3. SNA/DS security, to ensure that the path traversed by a SNA/DS conversation is secure if so desired.

4. The SNA/DS origin name is provided to the SNA/FS server on the SNA/DS server protocol boundary, so that an intervention exit in the server can examine it.

5. Additional security to ensure that change management agents are trusted at a SNA/DS DSU may be provided by the node's operating system.

Network management data that is transported in an SNA/DS MU with the security service level must originate at a trusted DSU and always be transported in an MU specifying a security service level.

1.2.4 Focal Point Concepts

The concept of a *focal point* was developed to allow the network administrator the opportunity to centrally manage a distributed or peer network. A focal point is an entry point that provides centralized management and control for other entry points for one or more network management categories.

When a focal point to entry point relationship needs to be established to enable the sending of unsolicited data from the entry point, *MS Capabilities* major vectors are exchanged (using the MDS transport service) between the MS_CAPS function set in the focal point and the MS_CAPS function set in the entry point. This exchange establishes the relationship between a focal point and an entry point for a particular category of management services. (4) The details of this exchange are described later in this section in "Establishment of a Sphere of Control" in topic 1.2.4.2.1. The set of nodes having this relationship with a focal point is known as the *sphere of control*, or SOC, of the focal point, and each of the individual nodes directly in the sphere of control is known as a *SOC node*.

APPN network nodes are SOC nodes both for themselves and for their client end nodes. The network node provides a simplified focal point notification message to its client end nodes. This default arrangement simplifies network administration, since focal points need be aware only of the network nodes in their SOC. It also allows for simplified end node implementations, since the end node need support only the focal point notification message from its network node server. However, end node implementations may optionally provide the same level of support as network nodes, and be SOC nodes themselves. This function may be useful in certain network configurations, where the services of a network node server are not available.

Nodes can be assigned to a focal point's sphere of control, or can be acquired independently of network operator definition (for example, from the topology of the network). In the first case, the focal point is referred to as an *assigned* focal point; in the second case, the focal point is referred to as a *default* focal point.

Assigned Focal Point:

Nodes can be assigned to a focal point's sphere of control in two ways.

- An *explicitly* defined sphere of control is one that is defined at a focal point. The focal point is responsible for initiating and establishing this focal point to entry point relationship. Recovery of the relationship after failures may be assigned to either the focal point or the entry point, depending upon the focal point's capabilities (this assignment of responsibility is made at the time the focal point initially contacts the entry point).

- An *implicitly* defined sphere of control is one that is defined at the various entry points. It is not explicitly defined at the focal point. The focal point finds out about the SOC nodes and adds them to its sphere of control as they ask for services by sending an MS Capabilities message to the focal point. The entry points are responsible for initiating, establishing, and maintaining this focal point to entry point relationship.

Default Focal Point:

A *default* focal point does not have a sphere of control explicitly assigned. Instead, it learns of the identity of APPN network nodes by examining the network topology, and sends an MS Capabilities message to each network node, offering to provide focal point services for a particular MS category. This request from a default focal point is accepted by a network node only if no other focal point has been assigned to it for the category (either explicitly or implicitly).

Since APPN network topology information is limited to network nodes, a focal point will only send this kind of MS Capabilities message to an APPN network node, and never to APPN end nodes.

Multiple Focal Points:

A network may have multiple focal points. These focal points may have responsibility for the same or different category of management services data. However, the spheres of control for multiple focal points for the same management services category may **not** overlap (in other words, a node may not be in the sphere of control of multiple focal points for the same category at the same time).

There are two kinds of category names used by focal points:

1. The architecturally-defined names for the various management services function sets (see Table D-1 in topic D.0 for a list of these names).

2. Installation-defined names for categories not specified by the management services architecture.

   Implementations of the architecture may provide the facilities for user-written application programs to act as focal point and entry point functions for user-defined categories.

---

(4) For the change management category, the relationship between focal point and entry point is *not* established by the exchange of MS Capabilities. For that category, a network planner prepares a *change management plan*, which identifies a set of entry points and a set of actions to be performed by the focal point for those entry points. The set of entry points in a particular plan is *not* a sphere of control.

Subtopics:

- 1.2.4.1 Types of Focal Points
- 1.2.4.2 Properties of Focal Points

1.2.4.1 Types of Focal Points

There are two *types* of focal points:

- Primary

  A primary focal point is the preferred destination for unsolicited data for a particular management services category.

- Backup

  A backup focal point is one that provides management services for a node in the event that the services of the primary focal point are unavailable.

Both primary and backup focal points may be assigned (*implicitly* or *explicitly*) or may be defaulted (for APPN network nodes only). The mechanism by which a focal point was assigned is significant with regard to focal point *ranking*. For example, a primary focal point which was assigned implicitly (referred to as an *implicit primary*) has a lower rank

than a primary assigned explicitly (known as an *explicit primary*). Focal point ranking is described in "Takeover by Other Focal Points" in topic 1.2.4.2.2.

The exchanges associated with the establishment of a focal point to entry point relationship for primary and backup focal points are described, at a conceptual level, in the following sections.

Subtopics:

- 1.2.4.1.1 Primary Focal Point
- 1.2.4.1.2 Backup Focal Point

1.2.4.1.1 Primary Focal Point

A *primary* focal point for a particular MS category is understood to be the preferred destination for unsolicited data belonging to that category. It is also a place from which requests for additional data, or enabling of additional functions, may originate.

The illustrations in the following sections illustrate, at a conceptual level, how the focal point to entry point relationship is established for a *primary* focal point. The following exchanges are illustrated:

- Focal point notification to APPN end nodes
- Explicit sphere of control
- Implicit sphere of control
- Default focal point

These illustrations use English language analogies to convey the basic architectural concepts. More details about the actual message units and transport are given in "Properties of Focal Points" in topic 1.2.4.2. In the illustrations which follow, the messages such as "I will be your primary FP," which are exchanged between the nodes are all encoded in the MS Capabilities (X'80F0') major vector, and are transported between the nodes over the LU-LU and CP-CP sessions used by the multiple-domain support.

The illustrations in the following sections show only APPN network nodes acting as SOC nodes; however, APPN end nodes may optionally be SOC nodes themselves. A *SOC end node* provides the same level of function as a network node, with the exception of the network node's support for non-SOC end nodes.

Focal Point Notification to APPN End Nodes:

```
 _________                _________              _________
|  Node A  |              |  Node B  |           |  Node C  |
|   (FP)   |              |   (NN)   |           |   (EN)   |
|__________|              |__________|           |__________|
     .                        .                       .
     .    FP-EP establishment .                       .
      _ _ _ _ _ _ _ _ _ _ _ _ ÿ                       .
     .                        .       FP is A         .
     .                         _______________________ÿ
     .                        .                       .
     .                        .                       .
     .     ** loss of FP **   .                       .
     .                        .    FP is <null>       .
     .                         _______________________ÿ
     .                        .                       .
```

Figure 2-39. Focal Point Notification from APPN Network Node to Client End Nodes

Regardless of the focal point type (primary or backup) and the MS Capabilities exchanges that take place to establish a focal point to entry point relationship between a focal point and an APPN network node, the end node clients of that network node receive the same focal point notification, illustrated in Figure 2-39. An end node implementation which provides only the *base* function for the MS_CAPS function set does not participate directly in any other MS Capabilities exchanges. Whenever a focal point (for any category of management services) is acquired by the network node server, the end node receives a notification identifying the category and the focal point. When a focal point is no longer available, the network node server sends the same notification, but with null values for the focal point identity.

Explicit Sphere of Control:

```
|
|     _________          _________          _________
|    |  Node A  |       |  Node B  |       |  Node C  |
|    |   (FP)   |       |   (NN)   |       |   (EN)   |
|    |__________|       |__________|       |__________|
|         .                  .                  .
|         . I will be your primary FP.          .
|         ___________________________ÿ          .
|         .          Accept          .          .
|         ___________________________          .
|         .                          .   FP is A  .
|         .                          ___________________________ÿ
|         .                          .                  .
```

Figure 2-40. MS Capabilities Exchanges for Explicit SOC

An *explicit* sphere of control is one where the set of nodes to be included are defined by the network operator at the focal point. Support for explicit sphere of control is optional for APPN end nodes.

When an explicit focal point becomes active, it begins to establish the sphere of control for its category. This is described in more detail in "Establishment of a Sphere of Control" in topic 1.2.4.2.1. The explicit focal point attempts to send an MS Capabilities request to each node in its sphere of control table (SOC nodes) indicating that it will be the focal point for those nodes for its particular category of management services data. Since some nodes listed in the sphere of control may not be active at the the time the focal point is initialized, the focal point will use some mechanism to periodically resend MS Capabilities to those nodes, so that they may be added to the sphere of control as they become active. The MS Capabilities exchange for an explicit sphere of control is illustrated in Figure 2-40.

In the MS Capabilities request, the explicit focal point informs the SOC node of its own identity, and the identity of the backup focal point, if there is one. It also indicates to the SOC node which party (focal point or entry point) has the responsibility for recovery after failures. It is the MS Capabilities exchange that establishes the relationship between the focal point and entry point.

Implicit Sphere of Control:

```
|
|     _________          _________          _________
|    |  Node A  |       |  Node B  |       |  Node C  |
|    |   (FP)   |       |   (NN)   |       |   (EN)   |
|    |__________|       |__________|       |__________|
|         .                  .                  .
|         . Will you be my primary FP?          .
|         ___________________________          .
|         .          Accept          .          .
|         ___________________________ÿ          .
|         .                          .   FP is A  .
|         .                          ___________________________ÿ
|         .                          .                  .
```

Figure 2-41. MS Capabilities Exchanges for Implicit SOC

SOC nodes may, as an option, provide for local definition of focal point policy information. Such a node initiates an MS Capabilities exchange with the focal point that was defined (using local system definition) to it, indicating that it wishes to receive focal point services from that node. A focal point is required to accept this request, unless it not currently serving as a focal point for the indicated management services category. The MS Capabilities exchange for an implicit sphere of control is illustrated in Figure 2-41. Support for implicit sphere of control is optional for APPN end nodes.

Nodes that connect to a network via switched lines, and make infrequent contact with their focal point, may benefit from using this option.

| If the entry point having a focal point system defined to it is a network
| node, it still has the responsibility to notify its client end nodes of
| their focal point. It also still has the responsibility to acquire the
| services of a backup focal point, if one was defined to it, in the event
| of a failure in communications with the primary focal point.

| *Default Focal Point*:

```
|
|     _________          _________          _________
|    |  Node A  |       |  Node B  |       |  Node C  |
|    |   (FP)   |       |   (NN)   |       |   (EN)   |
```

```
|     |__________|              |__________|            |__________|
|          .                      .                        .
|          . I will be your default FP.                    .
|          __________________________ÿ                    .
|          .          Accept          .                    .
|          ____________________________                   .
|          .                          .        FP is A     .
|          .                          __________________________ÿ
|          .                          .                    .
```

Figure 2-42. MS Capabilities Exchanges for Default SOC

A *default* focal point is one that sends out MS Capabilities that requests "to be your focal point *if you do not currently have one*" (for a particular management services category). The recipient of this request will reject it if the entry point already has a focal point for the category. This differs from the request sent by an explicit focal point, which is accepted even if the entry point has an active focal point for the category. The MS Capabilities exchange for a default focal point is illustrated in Figure 2-42. The purpose of a default focal point is to provide focal point services for any nodes that may have been inadvertently left out of the spheres of control of the assigned focal points, usually in a multiple focal-point network. Requests are sent to all the network nodes not in the default focal-point's sphere-of-control table. (It uses the APPN topology database information to find out about these nodes.) Notice that implementation of a default focal point could result in significant network traffic when the focal point is initialized.

When a network node receives a request from a default focal point, it takes one of three actions:

- If it currently has an assigned focal point for the affected management services category, it sends a negative reply indicating that it does not need the default focal point's services. (It may also reject the request for any of the reasons that an ordinary MS Capabilities from an assigned focal point is rejected.)

- If it does not currently have an assigned focal point for the MS category, but does currently have a default focal point, it accepts the request, and sends a revocation Focal Point Authorization Reply (X'62') subvector to the current default focal point, stating that it no longer needs the services of the original default. This *default takeover* is necessary to allow a default focal point to retake a node that has switched to the default's backup when an error occurred.

- If it has neither an assigned nor a default focal point for the affected management services category, it simply accepts the request. Once it has done this, its relationship to the default focal point is no different from a relationship with an assigned focal point except for possible ranking decisions when the next MS Capabilities exchange takes place.

In this way, a default focal point is able to provide service to network nodes that are not currently assigned any focal point, without affecting nodes that have been assigned a focal point through MS Capabilities exchange. It is optional for a focal point to serve as a default focal point. It is not optional for a node to be capable of receiving requests from default focal points, since the knowledge of whether a node will exist in a network serviced by a default focal point cannot be known ahead of time.

There are several significant points to be made about default focal points and their requests:

- Default focal points can serve network nodes only by way of network node awareness. End nodes are guaranteed coverage because each must belong to some network node's domain, but default requests are never sent directly from a focal point to an end node.

- A single focal point can be both an assigned focal point (for a specified set of SOC nodes) and a default focal point (for all other network nodes). Such a focal point sends assigned requests to nodes in its own sphere-of-control table, and default requests to all other network nodes.

- If the sphere-of-control tables in a network have all been constructed correctly, then requests from a default focal point should be eventually overridden by the assignment of an explicit or implicit focal point. This is because all network nodes should be covered by some focal point, which means that each network node should have been included in some focal point's sphere-of-control table.

In networks with a single focal point, where the only nodes in the sphere of control would be network nodes, a focal-point implementation may use only default focal point to establish focal-point/entry-point relationships. This usage eliminates the need for defining a sphere-of-control table in this particular circumstance. It is important to note that the default focal point relies entirely upon the dynamic APPN topology database to define those network nodes that are candidates for

```
| its sphere of control.  Thus, this type of focal point is useful only
| within a single, contiguous APPN network.  It is not as useful in
| dis-contiguous APPN networks, such as those that are interconnected via a
| subarea network, since the topology database in one portion of the APPN
| network does not reflect the existence of network nodes in another portion
| of the network.
```

1.2.4.1.2 Backup Focal Point

```
|
|     __________              __________              __________
|    |  Node A  |            |  Node B  |            |  Node C  |
|    |   (FP)   |            |   (FP)   |            |   (NN)   |
|    |__________|            |__________|            |__________|
|         .                       .                       .
|         . I will be your primary FP; backup FP is B     .
|         _______________________________________________ÿ
|         .                    Accept                     .
|         _______________________________________________
|         .                       .                       . FP is A
|         .                       .                       __________ÿ
|         .                       .                       .
|       ****************** some time later **********************
|         .                       .                       .
|         .                        Data                   .
|         . *** error *** ______________________________
|         .                     error notification        .
|         .              |_____________________________ÿ
|         .                       Will you be my backup FP?
|         .                       _________________________
|         .                       .        Accept         .
|         .                       _________________________ÿ
|         .                       .                       . FP is B
|         .                       .                       __________ÿ
|         .                       .         Data          .
|         .                       _________________________
|         .                       .                       .
|       ****************** some time later **********************
|         .                       .                       .
|         .            Will you be my primary FP?         .
|         _______________________________________________
|         .                 Accept  .                     .
|         _______________________________________________ÿ
|         .                       .You are no longer my FP.
|         .                       .    My new FP is A     .
|         .                       _________________________
|         .                       .                       . FP is A
|         .                       .                       __________ÿ
|         .                       .                       .
```

Figure 2-43. Switch to Backup FP, Then Recovery of Primary

A *backup* focal point is defined as a focal point that provides management services for a node in the event of a communications failure with the primary focal point. Both being a backup focal point and having a backup focal point are optional functions. The MS Capabilities exchanges associated with learning the identity of a backup focal point, switching to the backup, and switching back to the primary when it becomes available, are illustrated in Figure 2-43. Support for switching to a backup focal point is optional for APPN end nodes.

The use of a backup focal point is described in detail in "Switching to a Backup Focal Point" in topic 1.2.4.2.3. After a SOC node has switched to a backup focal point, the relationship with the primary focal point will be reestablished when possible. If the focal point is an *explicit* primary, and it sets the MS Capabilities Recovery Responsibility indicator (byte 4 bit 0) in the Primary Authorization Request (X'10') subfield of the X'61' subvector to 0, it is responsible to reestablish the relationship when possible by resending the X'61' subvector. If the focal point set the recovery responsibility indicator to 1, then the entry point is responsible for reestablishing the relationship just as if the focal point had been *implicitly* defined. If the primary is an *implicit* focal point, the entry point must reestablish the relationship by resending the Entry Point Authorization Request (X'63') subvector.

All policy input concerning the existence of a backup focal point is done at the same node at which the primary focal point was defined. In other words, definition of a backup focal point occurs at the primary focal point for explicit spheres of control, and definition of a backup focal point occurs at the same node where the definition of the primary focal point occurred for implicit spheres of control. In the same manner, the definition of a backup for a default focal point occurs at the primary default focal point. Backup focal point definition may be on a node basis (in which case each node is assigned a specific backup for each category), or a single backup can be specified for the entire category's sphere of control, as an optimization.

A node within a sphere of control of an explicitly assigned or default focal point learns about its backup focal point (if it has one) through the MS Capabilities exchange with its focal point. The backup focal point's identity is included in the same subvector with the primary focal

point's identity. A node's backup focal point can be changed or removed by changing the entry in the sphere of control table at its primary focal point. The primary then initiates a new exchange of MS Capabilities with the affected node, to pass to it either the name of the new backup focal point or an indication that it no longer has a focal point.

Once the relationship defined in the MS Capabilities exchange is accepted, a SOC node removes all knowledge of its primary *and* backup focal points from its current table, whether it was determined from prior MS Capabilities exchanges or from local definition. This means that if an accepted MS Capabilities contains no identification of a backup focal point, the SOC node no long has a backup, even if a backup focal point was known previously. An exception to this rule is if the *backup focal point flag* in the Focal Point Identification Flags (X'02') subfield in the Focal Point Identification (X'21') subvector is set to 1, which is provided for use if the focal point wants the node to keep the same backup that it already has, without the focal point having to know or send the name. In this case, the backup focal point attains the same rank as the primary focal point, regardless of its previous definition.

1.2.4.2 Properties of Focal Points

Focal points have certain characteristics that allow them to provide services to and manage various nodes in a network. The most important of these is the property of having a *sphere of control*, or a set of nodes they are responsible to manage. This leads to various other activities that involve changing a node's inclusion in a sphere of control, and recovering services in the event of a communication error to the focal point. A focal point for a category that does not support any unsolicited data would have a sphere of control, but would not need to establish focal point to entry point relationships with MS Capabilities exchanges.

In "Establishment of a Sphere of Control" in topic 1.2.4.2.1, detailed time lines of the exchanges that occur between function sets are presented. For a more detailed explanation of the events occurring at each step in the establishment of a sphere of control, see "MS CAPS Function Set" in topic 2.9.3.

Subtopics:

- 1.2.4.2.1 Sphere of Control
- 1.2.4.2.2 Takeover by Other Focal Points
- 1.2.4.2.3 Recovery Properties of Focal Points

1.2.4.2.1 Sphere of Control

A focal point for an MS category has a sphere-of-control table, listing
| the nodes for which it is to provide focal-point services. The method by
| which a node is included in or excluded from this table is different for
| explicitly assigned, implicitly assigned, or default focal points
| (explicitly entered by the network operator, added after contact by the
| SOC node, or added when the network node is added to the network
| topology). Nodes listed in this table are said to be "in the
sphere-of-control table." Once MS Capabilities are exchanged, establishing the relationship between the focal point and the entry points, these nodes are said to be "in the sphere of control."

The typical case is for the nodes in a focal point's sphere-of-control table to be network nodes. In this case, a network node domain, i.e., the network node and its client end nodes, are treated as a unit from the focal point's perspective.

An end node may (as a policy decision of the individual defining the focal point to entry point pairs) appear directly in a focal point's sphere-of-control table. In this case the focal point deals directly with the end node, and the end node is effectively removed from its network node server's domain for this category of management services. Note, however, that it is optional for APPN end node implementations to support this function.

Spheres of control for a single management services category are non-overlapping; in other words, a node has only one focal point at a time for each management services category.

Subarea focal points exist as either implicit focal points or use some nondeterministic method to manage their spheres of control. A *non-deterministic* method in this context is one that relies upon periodic retries to determine if connectivity exists between the focal point and a node in the sphere of control. These retries may be implemented at either the focal point side or the SOC node side. Retry is done by the SOC node either when the focal point is implicit (defined at the SOC node), or for an explicit focal point, the focal point may indicate in the Focal Point

Authorization Request (X'61') subvector of the MS Capabilities major vector that the SOC node is responsible for retry. A more efficient approach is available to a focal point in an APPN network, which may use updates to the topology database as a *deterministic* method to determine if the desired connectivity exists. However, a focal point in an APPN network must also implement nondeterministic retry mechanisms, since, in certain configurations (those involving subarea connectivity between portions of an APPN network), the topology database is incomplete.

Knowledge of the current focal point for each management services category is forwarded from a network node server to its client end nodes through a type of MS Capabilities exchange. This "abbreviated" MS Capabilities exchange includes the category and the current focal point identification only. The same exchange is used to notify local application programs that have registered for this category. The network node server also forwards an indication when a focal point is no longer available. The end node waits for notification of a focal point to resume sending of unsolicited data.

Establishment of a Sphere of Control:

The following sections outline in detail the steps necessary in the establishment of a focal point to entry point relationship between various combinations of nodes in a network (focal points, network nodes, and end nodes). Each section should be consulted in combination with the appropriate sections of "MS_CAPS Function Set" in topic 2.9.3, that describe the actual function provided by the MS_CAPS function set.

In the following sections, references exist to communication across protocol boundaries and between nodes using MS Capabilities major vectors. The protocol boundaries are documented in "Protocol Boundaries Between MS Function Set Groups" in topic 2.9.1.2, and the MS Capabilities major vector and its associated subvectors and subfields are documented in *SNA Formats*.

Initiation of an Explicit Sphere of Control:

An explicit sphere of control is initiated by the focal point. As the focal point becomes aware of connectivity to nodes in its sphere of control table, it establishes the focal point to entry point relationship by exchanging MS Capabilities with the nodes. The focal point uses MDS to transport these requests.

The focal point sends an MS Capabilities major vector with a Focal Point Authorization Request (X'61') subvector requesting to be the focal point of the target node. The following sequence of events occurs:

```
   ____________________________________ ____________________________________
  |          Focal Point A           ||            SOC Node B              |
  | __________ _________ __________ || _________ __________ __________ |
  ||XXXX_NETOP|| MS_CAPS |   MDS    ||||   MDS   |  MS_CAPS ||  EP_XXXX ||
  ||__________||_________|__________||||_________|__________||__________||
  |____________________________________||________________________________|
       °          °          °          °          °          °
 1)    °          °          °          °         o___(PBT)__°
       °          °      MDS-MU [61SV]  °          °          °
 2)    °          °_______________________          °          °
       °          °          °         |___(PBQ)_ÿo          °
       °          °          °          °          |          °
       °          °       MDS-MU [62SV]  ___(PBR)__|          °
 3)    °          o_______________________|         |         °
       °          °          °          °          |          °
       °       MDS-MU [62SV (revoke)]    ___(PBR)__|          °
 4)  _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _____|         |         °
       °          °          °          °          |          °
 5)    °          °          °          °          |___(PBV)__ÿo
|      °          °          °          °          |          °          _
|  6)  °          °          °           ___(PBR)__|          °          | NN
|      °          °          °         |          MDS-MU [E1SV]          | only
|      °          °          °         |_ _ _ _ _ _ _ _ _ _ _ _ _ _ ÿ  _|
       °          °          °          °          °          °
```

Figure 2-44. Initialization of an Explicit Sphere of Control. This figure illustrates the flows that occur across protocol boundaries and between function sets needed to establish an explicit sphere of control. The figure illustrates a focal point and a SOC node in its sphere of control. Flows marked with PB*n* represent communication across a protocol boundary within a node, and flows marked with *xx*SV represent MS Capabilities major vectors containing the identified command subvector.

The explanation of the flows contained in Figure 2-44 is as follows, with references given to the guide numbers on the left of the figure (note that the diagram begins after the MS_CAPS components on each node and the MS application programs on each node have registered themselves with the MDS component across protocol boundary O): (5)

1. Each EP_XXXX application program on node B registers its existence with MS_CAPS. This involves telling MS_CAPS the name of the application program and the MS category it wants information about. This is accomplished across protocol boundary T (documented in "Protocol Boundary T--Register MS Category" in topic 2.9.1.15).

2. The MS_CAPS component in focal point A sends MS Capabilities to the nodes listed in its SOC, including node B.

3. Acceptance or rejection of the MS Capabilities major vector is returned by node B by creating a MS Capabilities major vector with a X'62' command subvector and the proper subfields, placing the major vector in an MDS-MU, and passing the MDS-MU across protocol boundary R to the MDS function set for transmission.

   If the MS Capabilities Recovery Responsibility indicator (byte 4 bit 0) in Primary Authorization Request (X'10') subfield of the X'61' subvector is set to 1, indicating that the SOC node is responsible for error recovery, the explicit primary and backup focal foint names are saved in non-volatile storage as the new *implicit* focal point names, permanently overriding any previous system definition. The previous implicit focal point definition will either be erased or simply marked as deleted, provided that the implementation does not reuse those definitions. For example, an implementation *will not* save the overridden implicit focal point name to use as a fallback in case the new implicit primary and backup are not available, since such usage would violate the expressed policy intent of the network manager.

4. If node B had a previously active focal point for the category included in the X'61' subvector, and it accepted the request from focal point A, a MS Capabilities major vector with a Focal Point Authorization Reply (X'62') subvector is sent to the previous focal point, revoking its authorization.

5. When MS_CAPS in node B has responded positively to the X'61' subvector and added the focal point name to its MS Capabilities table, it sends an MS Capabilities major vector to its local application programs which have registered for the indicated focal point category identifying focal point A as the current focal point for the category. This notification is accomplished across protocol boundary V (documented in "Protocol Boundary V--Current Focal Point" in topic 2.9.1.17) from the MS_CAPS component to the registered application programs.

6. If SOC node B is a network node, its MS_CAPS component also sends to each client end node an MS Capabilities major vector containing Focal Point Notification (X'E1') subvector to notify it of the new focal point name.

Initiation of an Implicit Sphere of Control:

A node may optionally have its focal points' names defined to it by local definition or operator input. Such a node is responsible for initiating the MS Capabilities exchange. A node need not try to contact its implicit focal point unless it has something to send. The MS Capabilities sent by these nodes contain different command subvectors than the requests outlined in the initiation of an explicit sphere of control. The requests can be for the services of a primary or a backup focal point.

While it is optional for a SOC node to support local definition of focal point names, the support for an implicit focal point is a prerequisite to SOC End Node (MS_CAPS optional subset 2), since an explicit primary focal point may assign recovery responsibility to the SOC node in Primary Authorization Request (X'10') subfield of the X'61' subvector.

The implicit focal point definition at a node must include the NETID and NAU name of the desired focal point as well as the category name. It includes this in its request for services to the focal point. The focal point includes in its response the actual application program name associated with the category (this is name to be used for routing MDS-MUs to the focal point application program, which may be different from the category name). Nodes initiating MS Capabilities exchange with an implicit focal point must be able to periodically retry the exchange, since the focal point may not be available at first.

Encoding of Implicit Focal Point Name

The encoding of an MS Capabilities major vector for an implicit focal point request uses the Focal Point Identification (X'21') subvector to identify the focal point. This encoding has a 4-part identifier for the focal point and all 4 parts must always be included. The required values are category name, NETID, NAU name, and MS application program name. However, the MS application program name included in a Entry Point Authorization Request (X'63') subvector is not significant, since MS_CAPS at the focal point will always return the *actual* MS

```
| application program name in the Focal Point Identification (X'21')     |
| subvector which accompanies the Entry Point Authorization Reply        |
| (X'64') subvector.  The entry point MS_CAPS can default the MS         |
| application program in the Entry Point Authorization Request (X'63')   |
| subvector to be the same value as the category name.                   |
|                                                                        |
|________________________________________________________________________|
```

A SOC node that periodically disconnects from the network may need to have the focal point names pre-defined locally. It initiates an MS Capabilities exchange each time it reconnects to the network and before it has something to send.

```
   ________________________________ ________________________________
  |          Focal Point A         ||            SOC Node B          |
  | __________ ________ __________ || _________ __________ __________ |
  ||XXXX_NETOP|| MS_CAPS |   MDS    ||||   MDS   |  MS_CAPS ||  EP_XXXX ||
  ||__________||________|__________||||_________|__________||__________||
  |________________________________||________________________________|
        °         °         °          °         °         °
 1)     °         °         °          °         o _ _ _ (PB10) _ _ _
        °         °         °          °         |         °
        °                MDS-MU [63SV]     ___(PBR)__|         °
 2)              o_____________________|        °         °
        °        |       MDS-MU [64SV]      °         °         °
 3)     °        |_____________________     °         °         °
        °         °         °          |___(PBQ)_ÿo         °
        °         °         °          °         |         °
        °     MDS-MU [62SV (revoke)]        _ (PBR)_ |         °
 4) _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _|         |         °
        °         °         °          °         |         °
 5)     °         °         °          °         |___(PBV)__ÿo
|       °         °         °          °         |         °          _
| 6)    °         °         °           ___(PBR)__|         °          | NN
|       °         °         °          |          MDS-MU [E1SV]         | only
|       °         °         °          |_ _ _ _ _ _ _ _ _ _ _ _ _ ÿ  _|
        °         °         °          °         °         °
```

Figure 2-45. A SOC Node Requests to be Included in the Implicit Sphere of Control. This figure illustrates the flows that occur across protocol boundaries and between function sets between a SOC node establishing the relationship with a focal point. This focal point will then have this node in its implicit sphere of control. Flows marked with PB*n* represent communication across a protocol boundary within a node, and flows marked with *xx*SV represent MS Capabilities major vectors containing the indicated command subvector.

The following is an explanation of the steps pictured in Figure 2-45.

1. The definition of the implicit focal point's identity is entered at SOC node B by the node operator.

2. After the identity of the implicit focal point has been defined, node B attempts to contact the focal point. It does this by creating an MS Capabilities major vector containing a X'63' command subvector, packaging it within a CP-MSU. This CP-MSU is placed in an MDS-MU, and passed to the MDS function set across protocol boundary R, which sends it to the prospective focal point node. Periodic retries by MS_CAPS may be necessary if this communication fails.

3. After the MDS-MU is received across protocol boundary Q from MDS, MS_CAPS determines whether or not to accept or reject the request. It responds by an MS Capabilities major vector with a X'64' command subvector.

4. If the node which originated the request had a previously active focal point, it generates a revocation MS Capabilities major vector containing a X'62' command subvector, and asks MDS to transmit it to the previous focal point. If there was no previously active focal point, this step is skipped.

   MS_CAPS must save the identity of active focal points in non-volatile storage. The revocation of a previously active focal point must take place *even if the entry point system has been re-initialized since the initial MS Capabilities exchange with the previous focal point*. This step is necessary since the previous focal point may still show the entry point as active in its sphere of control, even though the entry point node experienced a re-initialization.

5. When the MS_CAPS component in node B has successfully acquired a focal point, it sends an MS Capabilities major vector to the application programs on the node which registered themselves.

6. If SOC node B is a network node, its MS_CAPS component also sends to each client end node an MS Capabilities major vector containing Focal Point Notification (X'E1') subvector to notify it of the new focal point name.

Focal Point Notification to APPN End Nodes:

MS_CAPS in an APPN end node receives MDS-MUs from MS_CAPS in its network node server containing the MS Capabilities major vector with the Focal Point Notification (X'E1') subvector. This focal point notification is a simple one-way message: when MS_CAPS in the network node server acquires a focal point for a category, it sends Focal Point Notification (X'E1') subvector to MS_CAPS in each client end node. The Focal Point Notification (X'E1') subvector identifies the focal point and the category. The end nodes are not aware of the nature of the focal point (explicit, implicit, or backup), only that it is active. MS_CAPS in each end node, upon receiving the Focal Point Notification (X'E1') subvector, notifies its MS application programs, if any, which have registered for the category.

Similarly, when a network node server loses a focal point for a category, it sends Focal Point Notification (X'E1') subvector to each end node indicating that a particular category no longer has a focal point. This is indicated by null values for the focal point identification in the Focal Point Identification (X'21') subvector which accompanies the Focal Point Notification (X'E1') subvector.

The focal point notification flow is illustrated in Figure 2-46.

```
  ____________________________________ ____________________________________
 |          Network Node A            ||            End Node B              |
 | __________ _________ __________    || __________ __________ __________  |
 || EP_XXXX  || MS_CAPS |   MDS    |||| MDS     |  MS_CAPS ||  EP_XXXX  ||
 ||__________||_________|__________|||| _________|__________||__________||
 |____________________________________||________________________________|
      o          o          o           o          o          o
 1)   o          o          o           o          ___(PBT)___o
      o          o      MDS-MU [E1SV]   o          o          o
 2)   o          o_______________________           o          o
      o          o          o           |___(PBQ)_ÿo           o
      o          o          o           o          |          o
 3)   o          o          o           o          |___(PBV)___ÿ
      o          o          o           o          o          o
```

Figure 2-46. Focal Point Notification From a Network Node to an End Node. This figure illustrates the flow of a focal point notification from a network node server to a client end node.

The explanation of the flows contained in Figure 2-46 is as follows, with references given to the guide numbers on the left of the figure (note that the diagram begins after the MS_CAPS components on each node and the MS application programs on each node have registered themselves with the MDS component across protocol boundary O): (6)

1. Each EP_XXXX application program on node B registers its existence with MS_CAPS. This involves telling MS_CAPS the name of the application program and the MS category it wants information about. This is accomplished across protocol boundary T (documented in "Protocol Boundary T--Register MS Category" in topic 2.9.1.15).

2. The MS_CAPS component in network node A sends MS Capabilities to all of its client end nodes, including node B. The MS Capabilities contains Focal Point Notification (X'E1') subvector. Focal Point Identification (X'21') subvector identifies the category and the focal point name. If the focal point for a category is no longer active, the Focal Point Identification (X'21') subvector will indicate this via null values in the name fields.

3. MS_CAPS sends an MS Capabilities major vector to its local application programs which have registered for the indicated focal point category identifying the current focal point for the category. This notification is accomplished across protocol boundary V (documented in "Protocol Boundary V--Current Focal Point" in topic 2.9.1.17) from the MS_CAPS component to the registered application programs.

---

(5) Refer to "Protocol Boundaries Between MS Function Set Groups" in topic 2.9.1.2.

(6) Refer to "Protocol Boundaries Between MS Function Set Groups" in topic 2.9.1.2.

1.2.4.2.2 Takeover by Other Focal Points

In order to provide for dynamic change of sphere of control, a SOC node always accepts an MS Capabilities request from a focal point of equal or greater *rank* than its current focal point. The rank list for a SOC end node differs from the rank list for a network node, so two separate lists are given.

Focal Point Rank List for SOC End Nodes

1. Explicit primary focal point
2. Implicit primary focal point
3. Backup focal point (see note 1)
4. Focal point identity by virtue of receipt of Focal Point Notification (X'E1') subvector (see note 2)

Notes:

1. The backup focal point (number 3 in the list) has the same rank whether it originally came from implicit definition or from a Focal Point Authorization Request (X'61') subvector primary request.

2. If an end node has a back-level network node server, it *assumes* that the network node server is its focal point for the category ALERT_NETOP. (Actually, the back-level network node server will forward Alerts received from end nodes to its own focal point.) In this case, the end nodes gives the back-level network node server the same priority as a focal point identified in a Focal Point Notification (X'E1') subvector.

Focal Point Rank List for Network Nodes

1. Explicit primary
2. Implicit primary
3. Backup (implicit or explicit, see note 1)
4. Default primary
5. Backup (default, see note 2)
6. Pseudo focal point (if that option is supported)

Notes:

1. The backup focal point (number 3 in the list) has the same rank whether it originally came from implicit definition or from a Focal Point Authorization Request (X'61') subvector primary request.

2. A backup focal point (number 5 in the list) that is provided by a default focal point in a Focal Point Authorization Request (X'61') subvector secondary request has a lower rank than a backup that comes from implicit definition or from a Focal Point Authorization Request (X'61') subvector primary request.

Explicit focal point names (both primary and backup) received in Focal Point Authorization Request (X'61') subvector *permanently* override any previous implicit information when the focal point assigns responsibility for MS Capabilities recovery to the entry point via the MS Capabilities Recovery Responsibility indicator (byte 4 bit 0) in Primary Authorization Request (X'10') subfield of the X'61' subvector. When that indicator is set to 1, the SOC node stores the explicit focal point names as its implicit focal point names, as well. When connectivity to the explicit primary focal point is lost, error recovery is the same as for an implicit primary: a Entry Point Authorization Request (X'63') subvector is sent to the explicit backup focal point, if one is known, and periodic attempts are initiated to reestablish contact with the (now) *implicit* primary.

If the focal point sets the MS Capabilities Recovery Responsibility indicator (byte 4 bit 0) to 0, indicating that it is responsible for recovery, then the SOC node does not alter its implicit focal point information.

The implementation of focal point rank is described in the finite-state machines in "MS_CAPS Function Set" in topic 2.9.3.

If a node has an implicit focal point name, but has not experienced problems contacting that focal point, when it receives a default focal point request, it will assume that connectivity exists to the implicit focal point and reject the default request, giving the current implicit focal point name as its current focal point. If an error occurs when attempting to contact the implicit focal point, and it subsequently receives a default request, the SOC node will accept the default request; however, the SOC node will attempt to recover its implicit focal point when connectivity is restored. The method by which this determination is made is an implementation choice, and is not specified by the architecture. An example of a method for making this determination is a periodic (timer-based) attempt to reestablish the connection and send an MS Capabilities major vector containing Entry Point Authorization Request (X'63') subvector.

A SOC node must save its focal point names in nonvolatile storage, so that

implicit focal point identities will not lost if the node is reinitialized.

When a node currently associated with a focal point is taken over by a new focal point, it sends an MS Capabilities to the first focal point, to inform that focal point that it is no longer in its SOC. When a node *revokes* its previous focal point in this way, it sends an MS Capabilities major vector containing a X'62' subvector. It indicates its new focal point name in the major vector (NETID, NAU name, and MS application program name), as well as a reason code for the revocation.

| A revocation is not sent in either of the following cases:

| ° When a SOC end node has a focal point identity by virtue of receipt of
| a Focal Point Notification (X'E1') subvector.

| ° When a network node is a pseudo focal point (for use only in subarea
| networks)

The focal point does not send a reply to a revoking MS Capabilities. It updates the status of this node in the sphere-of-control table and notes the name of the focal point that took control.

1.2.4.2.3 Recovery Properties of Focal Points

When an error occurs in the communication of information between an entry point and a focal point, the recovery responsibility depends upon where the failure is detected.

If a failure occurs on the session between a network node server and a client end node, MDS in the end node creates MDS error messages which are passed to each MS application program immediately affected by the failure. A copy of each MDS error message is also passed by MDS to the MS_CAPS component. At this point the end node has no path to any focal point through the network node server (those focal points known to the end node from Focal Point Notification (X'E1') subvectors received from the network node server). For an end node which is not a SOC end node, this failure affects all focal points, since the only focal points it knows are those which the network node server told it about. A SOC end node might have some focal points, not learned through Focal Point Notification (X'E1') subvector from the network node server, which would be unaffected by loss of communication with the network node server.

MS_CAPS in the end node notifies all MS application programs which registered for focal point information for those categories affected by the failure. Each entry point application program reverts to its initial state, that is, the state it was in before it received any commands from a focal point.

Backup for an end node that is not a SOC node is thus provided not by giving the end node a backup focal point to use when it needs to, but by acquiring the services of a new network node server. In this way focal point services for an end node are similar to APPN services in general: the end node must have a network node server in order to get these services, and when it loses them temporarily, its mechanism for reacquiring them is to acquire a new network node server.

If an error is detected in communication between a SOC node and the focal point, the responsibility to attempt recovery is assumed by the party that initiated the focal point to entry point relationship. The assignment of this responsibility is discussed in "Backup Focal Point" in topic 1.2.4.1.2. If that attempt is not sucessful, the SOC node has the option to acquire the services of a backup focal point, if one is known.
| During an attempt to acquire the services of a backup focal point, an APPN
| network node does not notify its client end nodes of the temporary loss of
| management services support. This occurs if the backup focal point
| rejects the SOC node's request. If an explicit primary focal point
assigns responsibility to itself by the setting of the MS Capabilities Recovery Responsibility indicator (byte 4 bit 0), then the MS_CAPS component in the focal point, upon receiving a copy of an MDS error message from MDS, will attempt to resend MS Capabilities Focal Point Authorization Request (X'61') subvector to the SOC node to re-establish the relationship.

The SNASVCMG LU-LU session between a SOC node and its current focal point may be taken down without affecting the focal point to entry point relationship, so long as connectivity is maintained. It is when a node tries to *send* data to its focal point, and fails, that it breaks its relationship with that focal point, and it attempts to switch to its backup focal point (if it has one). This processing is further discussed in "Switching to a Backup Focal Point."

Implicit focal points do not try to recover nodes that contacted them. If a node in the sphere of control has been rejected or revoked, or if the explicit focal point has assigned responsibility for recovery to the SOC

node in Primary Authorization Request (X'10') subfield of the X'61' subvector, an explicit focal point does not retry either.

Switching to a Backup Focal Point:

The sequence of events involved in a switch over to a backup focal point is depicted in Figure 2-47.

```
     ___________________________________ ___________________________________ ___________________________________
    |          Focal Point B            ||          Focal Point A            ||           SOC Node C            |
    | _________ _________ _________     || _________ _________ _________     || _________ _________ _________     |
    ||XXXX_NETOP|| MS_CAPS |   MDS   ||||XXXX_NETOP|| MS_CAPS |   MDS   ||||   MDS   |  MS_CAPS || EP_XXXX ||
    ||__________||_________|_________||||__________||_________|_________||||_________|__________||_________||
    |___________________________________||___________________________________||___________________________________|
          o         o         o         o         o         o         o         o         o
          o         o         o         o         o         o         o   MDS-MU [data]    o
 1)       o         o         o         o         o         o         o  ___________________o
          o         o         o         o         o         o  __________|         o         o
          o         o         o         o         o          ERROR    o         o         o
          o         o         o         o         o         o  ___________   MDS-MU [errmsg]   o
 2)       o         o         o         o         o         o         |___________________ÿo
          o         o         o         o         o         o         |         o         o
 3)       o         o         o         o         o         o         |___(PBN)_ÿo         o
          o         o         o         o         o         o         o         |         o
          o         o         o         o    MDS-MU [63SV]  o          ___(PBR)__|         o
 4)       o         o_________________________________________________|         o         o
          o         |         o         MDS-MU [64SV]      o         o         o         o
 5)       o         |_________________________________________________         o         o
          o         o         o         o         o         o         |___(PBQ)_ÿo         o
          o         o         o         o         o         o         o         |         o
 6)       o         o         o         o         o         o         o         |___(PBV)__ÿ
|       o         o         o         o         o         o         o         |         o      _
|       o         o         o         o         o         o          ___(PBR)__|         o      | NN
|       o         o         o         o         o         o         |          MDS-MU [E1SV]     | only
|       o         o         o         o         o         o         |_ _ _ _ _ _ _ _ _ _ _ ÿ  _|
          o         o         o         o         o         o         o         o         o
 7)       o         o         o         o         o         o         *** (retry trigger) ***
          o         o         o         o         o         o         o         o         o
          o         o         o         o         o      MDS-MU [63SV]  ___(PBR)__o         o
 8)       o         o         o         o         o_____________________|         o         o
          o         o         o         o         |      MDS-MU [64SV]  o         o         o
 9)       o         o         o         o         |_____________________         o         o
          o         o         o         o         o         o         |___(PBQ)_ÿo         o
          o         o         o         o         o         o         o         |         o
          o         o         o          MDS-MU [62SV (revoke)]        ___(PBR)__|         o
10)       o         o_________________________________________________|         |         o
          o         o         o         o         o         o         o         |         o
11)       o         o         o         o         o         o         o         |___(PBV)__ÿ
|       o         o         o         o         o         o         o         |         o      _
|       o         o         o         o         o         o          ___(PBR)__|         o      | NN
|       o         o         o         o         o         o         |          MDS-MU [E1SV]     | only
|       o         o         o         o         o         o         |_ _ _ _ _ _ _ _ _ _ _ ÿ  _|
          o         o         o         o         o         o         o         o         o
```

Figure 2-47. Switching to a Backup Focal Point. This figure illustrates the flows which occur across protocol boundaries and between function sets on two focal points and a SOC node when an error is detected flowing information to a focal point and a switch to a backup focal point takes place. The flows establishing the focal point to entry point relationships occur before this figure, in precisely the same manner and sequence as shown in Figure 2-44 in topic 1.2.4.2.1. Flows marked with PB*n* represent communication across a protocol boundary within the node, and flows marked with *xx*SV represent MS Capabilities major vectors which contain the named command subvector.

The sequence of flows starts with the SOC node that already knows the current focal point (FP A) and knows the identity of its backup focal point (FP B).

1. Some time after the flows have occurred that establish the focal point to entry point relationship between focal point A and SOC node C (see Figure 2-44 in topic 1.2.4.2.1 for details of these flows), a communication is attempted from the EP_XXXX application program on node C to the XXXX_NETOP application program on focal point A.

2. A error prevents the delivery of the MDS-MU to the XXXX_NETOP application program program on focal point A. This error may involve either a problem in the underlying communications capability between the two nodes, or it may be some problem at the application programming level, such as a failure of the XXXX_NETOP application program itself. An MDS error message is returned to the originating application program.

3. MDS copies all MDS error messages to MS Capabilities via protocol
| boundary N. On a network node, this includes MDS error messages for
| local application programs as well as MDS error messages for
| application programs on client end nodes.

4. The backup focal point's identity was learned through the exchange of MS Capabilities with the primary focal point. MS_CAPS on node C sends an Entry Point Authorization Request (X'63') subvector to the backup focal point asking for its services.

5. MS_CAPS in node B determines whether or not to accept or reject this request (for example, it would reject the request if the application program for the requested category was not active) and creates an Entry Point Authorization Reply (X'64') subvector to respond to the request for backup services.

6. MS_CAPS in node C sends a notification of the new focal point's identity to its local application programs that have registered for information about the category. The application programs do not know that the new focal point is a backup. If the attempt to acquire the backup focal point had not been successful, MS_CAPS in node C would have send a notification to the local MS application programs that there was no longer a focal point for the category. This is accomplished by supplying null values for the focal point identification.

   If SOC node C is a network node, its MS_CAPS component also sends to each client end node an MS Capabilities major vector containing a Focal Point Notification (X'E1') subvector with the same information provided to the local application programs.

7. When node C has successfully switched to its backup focal point B, it initiates a retry mechanism so that it may switch back to its primary focal point A when communication is once again possible. The nature of this retry mechanism is not specified by the architecture. It may be a timer to periodically attempt to send MS Capabilities to focal point A. The SOC node may also elect to retry focal point A periodically when it has a data record to send to focal point B.

   This retry mechanism is triggered in the example.

8. MS_CAPS in node C prepares an MS Capabilities major vector containing an Entry Point Authorization Request (X'63') subvector and attempts to send it to MS_CAPS in focal point A.

9. The communications succeeds, and MS_CAPS in focal point A receives the MS Capabilities with the Entry Point Authorization Request (X'63') subvector. It accepts it, and prepares MS Capabilities with an Entry Point Authorization Reply (X'64') subvector as a reply. This reply is sent back to MS_CAPS on node C.

10. MS_CAPS in node C receives the Entry Point Authorization Reply (X'64') subvector, marks focal point A as active, prepares MS Capabilities with a revocation Focal Point Authorization Reply (X'62') subvector, and sends it to MS_CAPS in backup focal point B.

11. MS_CAPS in node C node sends notification of the new focal point's identity to its local application programs. The application programs do not know that the new focal point is the restored primary.

    If SOC node C is a network node, its MS_CAPS component also sends to each client end node an MS Capabilities major vector containing a Focal Point Notification (X'E1') subvector with the same information provided to the local application programs.

1.2.5 Identification of Resources and Application-Level Routing

Subtopics:

- 1.2.5.1 Identifying SNA-Addressable Resources
- 1.2.5.2 Identifying Resources That Are Not SNA-Addressable
- 1.2.5.3 Routing a Request to a Specified Application Program

---

1.2.5.1 Identifying SNA-Addressable Resources

SNA/Management Services identifies SNA-addressable resources with the SNA Address List (X'04') subvector. This subvector transports the network and local addresses by which resources are ordinarily identified within an SNA node. CPMS provides address-to-name and name-to-address translation for these addresses, so that a network operator viewing management services data such as Alerts, or entering management services commands, need only know resources by their names.

The SNA Address List is used only on the SSCP-PU session, and only for resources for which the control point is prepared to perform address-to-name, or name-to-address, translation. It is not used in the CP-MSUs that transport MS major vectors between control points.

1.2.5.2 Identifying Resources That Are Not SNA-Addressable

Resources that are not SNA-addressable are identified in different ways in request records (i.e., records that flow from an XXXX_NETOP to an EP_XXXX) and in replies and unsolicited records that flow in the opposite direction (i.e., from an EP_XXXX to an XXXX_NETOP). The two sections that follow detail these ways.

Subtopics:

- 1.2.5.2.1 Requests
- 1.2.5.2.2 Replies and Unsolicited Records

1.2.5.2.1 Requests

The Name List (X'06') subvector is the structure that identifies resources in requests. (This subvector also provides another capability, that of routing records to served network management application programs; see "Routing a Request to a Specified Application Program" in topic 1.2.5.3 for a description of this capability.) These application programs are not the MS function sets, which are themselves referred to as MS application programs in multiple-domain support. Rather, they are network management application programs served by an MS function set group. The Associated Resource Name List (X'01') subfield is used in common operations services requests to identify the resource or set of resources to which a request applies. "Common Operations Services Commands, Replies, and Unsolicited Traffic" in topic 2.10.5.5 contains a summary of the ways in which this subfield is used to identify resources in these requests.

1.2.5.2.2 Replies and Unsolicited Records

The Hierarchy/Resource List (X'05') subvector is the structure that identifies resources in replies and unsolicited records. The primary difference between it and the Name List (X'06') subvector is the provision for transporting code points that identify the *types* of resources involved, as well as their names.

Since the Alert (X'0000') major vector is the record that most fully uses the capabilities of the Hierarchy/Resource List subvector for identifying resources, the discussion of this subvector appears in the EP_ALERT section of this manual. See "Hierarchy Information in Alerts" in topic 2.10.1.19.10 for this discussion.

1.2.5.3 Routing a Request to a Specified Application Program

The architecture for SNA/Management Services provides for the routing of management services traffic to specific application programs running at a destination node. These application programs are not the MS function sets, which are themselves referred to as MS application programs in multiple-domain support. Rather, they are network management application programs served by an MS function set. Since a request must first be routed to the serving function set (in other words, to the serving MS application program), and then to the served application program, this type of routing is referred to as *second-level application program routing*.

The Routing and Targeting Instructions (X'154D') GDS variable is the structure used in Operations Management to accomplish second-level routing. It contains the Name List subvector, whose subfields provide several diverse methods for targeting remote resources. The Common Operations Services architecture uses the Name List subvector also, but only within a major vector (not GDS variable) encoding.

Common Operations Services second-level application program routing uses the Destination Application program Name (X'50') subfield within the Name List (X'06') subvector. This subfield transports a 1- to 8-character application program identifier. When it receives a major vector containing this subfield, an MS function set group that supports routing of this type knows that the major vector is intended for an application program that it serves. If it is aware of an application program with the

specified identifier, the MS function set group passes the major vector to it. Otherwise it rejects the major vector, sending sense data X'8018 0001' (application program unknown).

For more information concerning Operations Management second-level routing, see "The Routing and Targeting Instructions GDS Variable" in topic 1.7.3.3.1.

Currently the only function set groups providing second-level application program routing are EP_COMMON_OPERATIONS_SERVICES and EP_OPERATIONS_MGMT.

1.3 Chapter 3. Problem Management

Problem management is the process of managing a problem or potential problem from its detection through its final resolution. The term *problem* denotes an error condition resulting in a loss of availability of a system resource that is visible to an end user. Problems may originate in hardware, software (operating systems and application programs), microcode media, or as a result of external causes such as user procedures or environmental abnormalities. For an overview of the elements of problem management and their relationships to one another, refer to Figure 3-1.

Figure 3-1. Overview of Flow Between Problem Management Elements. SNA management services are provided to assist with elements bounded by the dotted lines. Though ALERT_NETOP provides no problem bypass and recovery function, an interface is provided to operations management for these services. See "EP_ALERT Optional Subset 14 (Operations Management Support)" in topic 2.10.1.18 for more detail.

Subtopics:

- 1.3.1 Problem Determination
- 1.3.2 Problem Diagnosis
- 1.3.3 Problem Resolution
- 1.3.4 Problem Determination, Diagnosis, and Resolution via Alerts and Resolution Notifications
- 1.3.5 Managing Problems in APPN Networks
- 1.3.6 Link Services

1.3.1 Problem Determination

Problem determination is the element of problem management that detects a problem or impending problem and completes the steps necessary for problem diagnosis to begin. Refer to Figure 3-2 for an overview of the steps required for problem determination.

The problem-determination process may be performed automatically by a system process, may be performed as a series of manual processes, or may be a combination of both automated and manual processes. There are two types of problem determination: node problem determination and system problem determination.

- Node problem determination addresses the class of problems associated with a node and its resources that are managed by a local node operator (either programmed or human). Typical of the type of problem managed by a local node human operator are "intervention required" situations such as "out of forms," "forms jam," and "mount requests."

- System problem determination is managed by a network operator at a

network control center, and provides the management of those problems that are not managed by the local node operator.

The term *problem determination* as used in this book refers only to system problem determination.

¹ The network operator will contact personnel responsible for problem diagnosis.

Figure 3-2. Overview of Problem-Determination Steps

Subtopics:

- 1.3.1.1 Overview
- 1.3.1.2 Problem Detection
- 1.3.1.3 Collection and Analysis of Problem Data
- 1.3.1.4 Reporting of Problem-Determination Data
- 1.3.1.5 Recording and Retrieval of Problem-Determination Data
- 1.3.1.6 Presentation of Problem-Determination Data
- 1.3.1.7 Processing of Requests for Problem-Determination Data

---

1.3.1.1 Overview

To complete system problem determination, network operators (human or programmed) must first be aware that a problem exists and then have access

to the data that will lead to the identity of the failing system resource, e.g., a machine or a link, and the data required to determine the organization responsible for its maintenance. System problem determination is composed of the following steps:

1.3.1.2 Problem Detection

The detection step of problem determination provides for the recognition of the loss or impending loss of availability of a system resource to an end user. The action of a network operator may be required for some or all of the problem determination, problem diagnosis, or problem resolution processes. The following conditions are detectable:

° Permanent loss of the availability of a resource that can be corrected only by intervention external to the reporting component

° Temporary loss of the availability of a resource that is corrected without intervention external to the reporting component

  This includes cases where the resource cannot consistently be used in the manner in which it was intended because of recurring errors.

° Response-time degradation beyond a predetermined value

° Problems recovered from, the existence of which prevented reporting at the time of occurrence

° Anomalies that have the potential to cause an interruption in availability to the end user, e.g., display screens starting to distort, a temperature out of a tolerable range, or a machine emitting a strange noise

Problem detection may be performed by any of the following:

° Local management services

° An end user or node operator (in cases where a problem is not automatically detected by local management services)

1.3.1.3 Collection and Analysis of Problem Data

The collection and analysis step of problem determination determines the following:

° Problem origin and cause

  - Physical identity of the system resource that caused the problem; e.g., machine type, model number, and serial number

  - Logical identity of the system resource that caused the problem, e.g., the name or address used by the system to identify the resource (This information is dynamic and may be changed by system generation or customization options.)

  - A very brief generic description of the problem and classification of its cause to one of the following:

    -- Hardware
    -- Software
    -- Microcode
    -- Media
    -- External cause; e.g., the end user

  This information is used in conjunction with system and/or user procedures so that the service organization responsible for problem diagnosis can be determined, directed to the location where the problem exists, and have a brief description of the problem to be worked on.

  Sufficient data to complete problem determination should be collected at the first occurrence of a problem. When possible, problem diagnosis data is also collected and analyzed at this time. If sufficient data is not collected at the time of the initial error, the problem will have to be re-created before problem determination and/or problem diagnosis can be completed. This is often unreliable and time consuming if the problem is intermittent.

The analysis step processes not only problem data (as collected by the problem-data collection step) but also other network management data that may be pertinent to the problem, such as change management or configuration management data.

° Action to be taken by the network operator to complete problem determination

° Severity or impact of the problem; e.g., permanent or temporary, so the network operator may set the priorities of the actions required

° Correlation of the following:

- Data related to this problem that is required for problem diagnosis or problem resolution
- End-user verbal reports of this same problem
- Other unsolicited notifications for this same problem sent by other nodes; Alerts are an example of of unsolicited notifications which are explored further in this chapter.

The collection and analysis of problem data may be performed by some or all of the following:

° The detector of the problem, i.e., the end user or local management services

° The local system operator (if applicable) if analysis had not been completed when the problem was reported by either an unsolicited notification or a verbal report

° The network operator if collection and analysis had not been completed when the problem was reported by either an unsolicited notification or a verbal report

The collection may be an iterative process and may use any or all of the following methods:

- Manually observing other indications at the control point

- Requesting the local node operator at the node reporting the problem to manually observe other indications of the problem at the node

- Requesting the local operator at the node reporting the problem to use services available at that node

---

1.3.1.4 Reporting of Problem-Determination Data

The reporting step of problem determination provides for reporting of problem-determination data to the control point responsible for managing the resource determined to be the origin of the problem.

° If an end user detected the problem, it may be reported directly to the network operator by a phone call or to the control point by a manually-initiated management services RU (Alert). It is optional for products to provide a mechanism to allow an operator or end user to initiate a management services RU (Alert).

° If local management services in a type 2.0 node detected the problem, it would report it to PUMS to be reported to the control point using a management services RU (Alert).

° If local management services in a type 2.1 node detected the problem, it would report it to the Alert-processing function in CPMS.

° If a management services application program detected the problem, it would report it to the Alert-processing function.

1.3.1.5 Recording and Retrieval of Problem-Determination Data

The recording and retrieval step of problem determination is implemented at the control point, processes only that data reported via Alerts, and provides the following:

- Selective logging of Alert data on the Alert data base, based on criteria specified by the network operator

- Data base management (e.g., storage, retrieval) of the collected data

- Interaction with the problem tracking and control and problem bypass and recovery elements

---

1.3.1.6 Presentation of Problem-Determination Data

The presentation step of problem determination is implemented at the control point and provides the following:

- Selective presentation of Alert data to the network operator based on criteria such as Alert type or product type

- Formatting and presentation of problem-determination data to the network operator in a manner designed to foster real-time awareness

---

1.3.1.7 Processing of Requests for Problem-Determination Data

The request step of problem determination is implemented at the control point and provides a mechanism for the network operator to request data from the Alert data base, or from the data-collection step for those cases where the data was not automatically reported.

1.3.2 Problem Diagnosis

Problem diagnosis is the element of problem management that determines the precise cause of a problem and reports the precise action required to resolve the problem. Figure 3-3 gives an overview of the steps required for problem diagnosis.

```
                    from
            problem determination
                      |

       ______________________________
      |                              |
 _____ÿ| collection of diagnostic data¹ |____________________
|      |                              |                     |
|      |_____________ _______________|                     |
|                    |                                     |
|                                                          |
|      ______________________________                      |
|      |                              |                     |
|______|   analysis of diagnostic data  |                     |
       |                              |                     |
       |_____________ _______________|        _____________|____________
                    |                         |                          |
                                             |  processing requests for   |
                                            |      diagnostic data²      |
       ______________________________        |                          |
      |                              |       |__________________________|
      |  reporting and presentation of |                  "
      |   problem resolution data      |                  |
      |                              |                from
      |______________________________|            problem determination
```

¹ If this process is performed automatically, it is started by the problem detection function of problem determination.
² If this process is not performed manually, the network operator will contact the personnel responsible for controlling it.

Figure 3-3. Overview of Problem Diagnosis Steps

The problem-diagnosis process may begin at the node detecting the problem and may be distributed through as many other nodes as is necessary for completion. The process may be performed automatically by a system process, may be performed as a series of manual processes, or may be a combination of both automated and manual processes.

If problem diagnosis is performed automatically by the system, it is usually done in parallel with problem determination, beginning with the collection step and ending with the presentation of both problem-determination and problem-resolution data. See Figure 3-5 in topic 1.3.4.3 and Figure 3-6 in topic 1.3.4.3 for details.

If problem diagnosis is performed manually, it begins at the conclusion of problem determination. Problem-determination output is used to determine the system resource responsible for the problem and the general area where problem diagnosis will begin. If diagnostic data was gathered automatically with problem-determination data, it is used to assist in problem diagnosis. If additional data is needed, it is gathered and analyzed manually or with the help of problem management services. Some problems may require several iterations of this process, each time gaining more data and eliminating more components from the set of possible causes.

Subtopics:

- 1.3.2.1 Overview
- 1.3.2.2 Collection of Diagnostic Data
- 1.3.2.3 Analysis of Diagnostic Data
- 1.3.2.4 Processing Requests for Diagnostic Data

1.3.2.1 Overview

Problem diagnosis is composed of the following steps:

° Collection of diagnostic data

° Analysis of diagnostic data

° Reporting and presentation of problem-resolution data

° Processing requests for diagnostic data

These are discussed individually in the following subsections.

1.3.2.2 Collection of Diagnostic Data

The collection step of problem diagnosis gathers information necessary to support the analysis step of problem diagnosis. Problem-diagnosis data is typically unique to an implementation and is defined in implementation documentation. The transport of that data, if it is available, is defined by management services. The collection step may be performed by any of the following:

° Local management services or CPMS (In these cases the diagnostic data or problem-resolution data is transported in an Alert.)

° Personnel responsible for problem diagnosis using the following:

- Serviceability aids and tools available at the node

- Resource control commands Query Resource Data and Test Resource if supported at the node

- Execute Command to transport requests for functions available at the node that are not otherwise within the scope of SNA/Management Services.

1.3.2.3 Analysis of Diagnostic Data

The analysis step of problem diagnosis processes diagnostic data to determine the precise cause of a problem and the precise action required for its resolution. The analysis step may be performed by any of the following:

° Local management services or CPMS (In these cases the diagnostic data or problem-resolution data is transported in an Alert.)

° Personnel responsible for problem diagnosis using any of the following:

- SNA problem management services at the control point

- Serviceability aids and tools available at the node.

- Manual analysis of diagnostic data either at the node or at the control point

- Use of the probable cause data reported by Analyze Status if it is supported at the node.

The output of this step is referred to as problem-resolution data.

1.3.2.4 Processing Requests for Diagnostic Data

The request step of problem diagnosis provides a mechanism for the operator to allow diagnostic data to be requested for those cases where collection, analysis, and reporting are not performed automatically by a system process and must be controlled by personnel responsible for problem diagnosis.

1.3.3 Problem Resolution

Problem resolution occurs when network operators (either human or programmed) no longer consider a particular problem to exist. This may be caused by correction of the error condition through analysis and recovery or simply as an administrative decision that the problem no longer warrants attention (for instance, if the temporary bypass is accepted as a permanent correction).

Problem resolution may be performed either automatically by a system process, as a manual process, or as a combination of both manual and automated processes. The steps required for problem resolution are represented in Figure 3-4.

```
                    |
        from problem diagnosis or
            problem recovery
                    |

 _______________________________
|        collection of          |
|    problem resolution data    |
|_______________ _______________|
                    |

 _______________________________
|        reporting of           |
|    problem resolution data    |
|_______________ _______________|
                    |
                                          _____________________________
 _______________________________          |     processing of network   |
|   recording and retrieval of  |         |     operator requests for   |
|     problem resolution data   |_____|   problem resolution data   |
|_______________ _______________|         |_____________________________|
                    |

 _______________________________
|       presentation of         |
|    problem resolution data    |
|      to network operator      |
|_______________________________|
```

Figure 3-4. Overview of Problem Resolution Steps

Subtopics:

- 1.3.3.1 Overview
- 1.3.3.2 Collection of Problem Recovery Data
- 1.3.3.3 Reporting of Problem Resolution Data
- 1.3.3.4 Recording and Retrieval of Problem Resolution Data
- 1.3.3.5 Presentation of Problem Resolution Data
- 1.3.3.6 Processing of Requests for Problem Resolution Data

1.3.3.1 Overview

To complete system problem resolution, network operators (human or programmed) must first be aware that a problem previously existed, what the identity of the problem was, and that the error condition does not

currently exist. Optionally, the operator may have access to information concerning the correction of the problem provided by components involved in problem bypass and recovery.

System problem resolution involves the following steps:

- Collection of Problem Resolution Information
- Reporting of Problem Resolution Information
- Recording and Retrieval of Problem Resolution Information
- Presentation of Problem Resolution Information

1.3.3.2 Collection of Problem Recovery Data

The collection step of problem resolution determines the following:

- Corrected problem identity

  In order to complete problem management of problems previously reported, it is necessary to identify the resolved problems that correspond to the current resolution activity. This is done through previous generation of unique identifiers and use of the same identifiers in further communication about the problem.

- Corrected resource identity

  The physical identity of the system resource that caused the problem (e.g., machine type, model number, and serial number) and the logical identity of the system resource that caused the problem (e.g., the name or address used by the system to identify the resource) is needed so that the service organization responsible for problem management can be notified of the actual resource that was modified to correct the problem. This resource may be different from the one proposed after previous steps.

- Recovery information

  All available information related to the resolution of the problem should be collected for subsequent reporting to, and storage by, network managers responsible for problem tracking and trend analysis. The collection step processes not only problem data (as collected by the components responsible for problem bypass and recovery) but also other data that may be pertinent to the problem, such as change management or configuration management data.

The collection of problem resolution data may be performed by the following:

- The detector of the problem (human or programmed)

- The local system operator (if applicable) if the problem is being resolved without recovery or if programmed operators are unable to determine the correction of the problem or correlate the correction to an existing problem

- The network operator if resolution occurs without recovery or if the recovery is reported verbally

---

1.3.3.3 Reporting of Problem Resolution Data

The reporting step of problem resolution provides for reporting of resolution data to the control point responsible for managing the resource determined to be the origin of the problem.

- If an end user detected the correction of the problem, it may be reported directly to the network operator by a phone call or to the control point by a manually-initiated resolution notification. It is optional for products to provide a mechanism to allow an operator or end user to initiate sending of a resolution notification.

- If local management services in a type 2.0 node detected the correction of a problem, it would report it to PUMS to be reported to the control point using a resolution notification.

- If local management services in a type 2.1 node detected the correction of a problem, it would report it to CPMS.

- If a management services application program detected the correction of a problem, it would report it to the Alert-processing function.

1.3.3.4 Recording and Retrieval of Problem Resolution Data

The recording and retrieval step of problem resolution is implemented at the control point. It processes only that data reported via resolution notifications, and provides the following:

- Selective logging of resolution data on the Alert database, based on criteria specified by the network operator

- Database management (e.g., storage, retrieval) of the collected data

---

1.3.3.5 Presentation of Problem Resolution Data

The presentation step of problem resolution is implemented at the control point and provides the following:

- Selective presentation of resolution data to the network operator based on criteria such as resolution type or product type

- Formatting and presentation of problem resolution data to the network operator in a manner designed to foster timely awareness of the correction of problems.

1.3.3.6 Processing of Requests for Problem Resolution Data

The request step of problem resolution is implemented at the control point and provides a mechanism for the network operator to request resolution data from the Alert database, or from the collection step for those cases where the data was not automatically reported.

1.3.4 Problem Determination, Diagnosis, and Resolution via Alerts and Resolution Notifications

Subtopics:

- 1.3.4.1 Functions Provided
- 1.3.4.2 Relationship of Errors, Problems, and Alerts
- 1.3.4.3 Variations in Alert Data
- 1.3.4.4 Implementation
- 1.3.4.5 Format Usage

1.3.4.1 Functions Provided

The PUMS component of each PU within a network and CPMS in a type 2.1 node provides an unsolicited notification when a problem is detected with any system resource that it manages or uses. That notification is an Alert. For details on the types of problems to be reported, refer to "Problem Detection" in topic 1.3.1.2. The unsolicited notification is reported to the Alert focal point for the failing system resource, and provides a method for transporting data related to the problem. These components may also provide unsolicited notifications when a correction to an error condition is detected. This notification is a resolution notification. In a manner similar to Alerts, it is reported to the Alert focal point for the failing system resource and provides a method for transporting data concerning the correction of the error condition. The optional correlation between the detection, diagnosis, and resolution of a problem is accomplished through the optional inclusion in each notification of a unique token that identifies the incident in question. Since a single corrective action may resolve multiple error conditions, a resolution notification can carry multiple incident identifiers. Alerts convey the occurrence of a single incident, and therefore carry only a single optional incident identifier.

Alerts and resolution notifications are processed (filtered) at the control point to determine if they should be stored or presented to a network operator.

The advantages of using unsolicited notifications are as follows:

- They facilitate the managing of a network from a central location.

- They provide immediate notification to the network operator of the existence of a problem and the action to be taken. Alerts allow the network operator to be aware of the problem before the end user complains about the loss of availability, thus aiding responsiveness to, and communication with, the end user. Resolution notifications allow the network operator to be aware of the disappearance or correction of problems in a similar manner.

- They provide for the transport of data captured at the time of failure rather than having to depend on later re-creation techniques that may not be effective.

- They provide for automating problem determination and problem diagnosis, thus decreasing the time and skill level required to resolve problems.

1.3.4.2 Relationship of Errors, Problems, and Alerts

The relationship of errors, problems, and unsolicited notifications must be understood before the conditions under which Alerts are sent can be understood. Before continuing, the reader should be familiar with the following terms:

Alert A record sent to an Alert focal point to communicate the existence of an Alert condition.

Alert condition A problem or impending problem for which some or all of the process of problem determination/diagnosis/resolution is expected to require action at a problem management focal point.

error The smallest detectable anomaly or exception that can occur in an information system. Errors may be caused by hardware, software, microcode, media, or external causes, e.g., people or environmental abnormalities.

error condition A state whose existence is indicated by (1) the occurrence of an error, and, possibly, (2) additional specific events, often errors resulting from attempts to recover from the initial error. Associated with each type of error is a recovery procedure to be followed when an error of that type occurs. The results of this recovery procedure determine whether the error condition is to be classified as *recoverable* or *irrecoverable*. An error condition is classified as *irrecoverable* if the operation being attempted when the error was detected is not completed without the use of recovery techniques external both to the component detecting the error, e.g., a hardware adapter, and to its controlling component, e.g., microcode controlling the adapter. A *recoverable* error condition is one in which the operation being attempted at the time the error was detected is completed by using a recovery procedure internal either to the component detecting the error or to its controlling component.

impending problem An error condition that has the potential for resulting in a loss of availability of a system resource to an end user.

problem An error condition resulting in a loss of availability of a system resource to an end user.

Error conditions become problems only when they affect availability, and become impending problems only when they threaten to affect availability. Also, problems or impending problems are reported by unsolicited notifications only when they require action by the network operator.

The term "impending problem" covers two different senses in which a condition may threaten to affect availability:

- If a quantity, such as temperature or utilization, will affect availability if it exceeds a particular threshold, then it threatens to affect availability as it approaches this threshold.

- If a piece of redundant hardware or software fails, then it threatens to affect availability by reducing the level of redundancy in place to handle the next failure. This is true regardless of whether the initial failure occurrs on the then-active component, forcing a switchover to a backup, or on a backup component not currently in use.

The Alert is sent to communicate the existence of a problem or impending problem for which some or all of the process of problem analysis has been completed, taking into consideration all of the available problem data, by

the node detecting the problem.

Problems can be detected only in those cases where a loss of availability can be detected. The following methods may be used to detect a loss of availability.

° Availability can be measured in those cases where it is possible to predict the performance that can be expected. If this can be done, it is possible to detect a performance level that deviates from the norm by an amount that for all practical purposes constitutes a loss of availability to the end-user and thus can be defined as a problem. This procedure is referred to as *performance analysis*.

° Rather than directly detecting a loss of availability, a procedure referred to as *error-rate analysis* can be used to infer loss of availability.

Error conditions are divided into two categories, irrecoverable and recoverable, which vary in their effects on availability.

- *Irrecoverable error condition*

  -- The occurrence of *n* consecutive errors of certain types constitutes an irrecoverable error condition. The number *n* varies depending on the type of error, implementation, or technology, and defines a threshold beyond which subsequent retries of the operation that prompted the initial error have little or no chance of resulting in a successful operation. Other thresholding techniques may also be used to determine an irrecoverable error condition. The technique used depends on the product or the technology involved.

  -- For errors of certain other types, a single occurrence constitutes an irrecoverable error condition. The operation resulting in the error is not retried. An example of this type of error is an "off-line," "unavailable," or "data unsafe" condition indicated via status or sense information.

  Unless specifically defined by the architecture, it is left to each implementation to determine what constitutes an irrecoverable error condition. Thresholds for these purposes may be set and/or modified by system definition parameters or by local or remote operators.

  Most irrecoverable error conditions cause a loss of availability of a system resource and thus fall into the category of problems. If, however, the operation requested by the end user is completed without the end user's knowledge of a loss of availability, the irrecoverable error condition is not classified as a problem. This may be the case in devices that have redundant hardware logic or other alternate means of performing an operation. These cases fall into the category of impending problems since the error condition cannot be resolved and the affected component will not be available without external intervention.

  Note that an error that is unrecoverable at one level within a node may be recoverable at a different level within the same node. In this section the term "irrecoverable error" indicates as error that is not recoverable at *any* level within the node experiencing it.

- *Recoverable error condition*

  Recoverable error conditions can be categorized as problems or impending problems when they have been judged to cause, or to have the potential for causing, a loss of availability to the end user. The following criteria may be used to make this judgment:

  -- Availability may be affected when the accumulated number of recoverable error conditions equals some preset threshold. The threshold may be described as *x* failures out of *y* attempts, or *x* failures in *t* time. It may also be described as *x* consecutive failures followed by a good operation, as long as *x* is less than the threshold used to determine an irrecoverable error condition.

  -- Availability may be affected by a single occurrence of an error such as exceeding a buffer threshold, storage usage threshold, or queue-length threshold.

  Unless specifically defined by the architecture, it is left to each implementation to determine what constitutes a loss of availability; such thresholds used to make this determination are dependent upon the error type and the device and are considered to be implementation unique. They may be set and/or modified by system definition parameters or by local or remote operators.

If both performance analysis and error-rate analysis are used to indicate the existence of a problem, two indications may exist for the same problem. This may aid in problem determination since it may indicate that it was the error condition that caused the loss of availability detected also as abnormal performance.

Errors conditions that do not affect availability and thus are not reported as problems may be logged locally. This data will be available, if required, for problem diagnosis. Logging of error data is an implementation decision based on the service requirements. Another option for error conditions that do not immediately affect availability is to report them as impending problems. Typically, this type of reporting is used for cases where temperatures or voltages are out of specification but the device continues to function normally. Reporting as an impending problem can also be used for error rates that do not affect availability but are judged by an implementation to be beyond what is normally expected.

1.3.4.3 Variations in Alert Data

Because of the variety of problems that may be encountered and the various ways that they may manifest themselves, it is not always possible for products to generate Alerts that contain complete problem determination and problem diagnosis information. The network operator may be presented data that represents any of the following situations:

- A problem has been detected but analysis is not complete. The origin and general cause of the problem have not been determined. Some data may have been gathered.

  Action Required by Network Operator

  The network operator is responsible for any additional collection of problem data that may be required, and for analysis to determine the general cause and the resource that is the origin of the problem. Collection may consist of: (1) retrieving data that was collected at the time of the failure but was not sent with the Alert; (2) running diagnostics to attempt to re-create the failure and collect data about it. User procedures are then employed to determine the organization responsible for problem diagnosis for further handling of the problem.

- A problem has been detected, analysis has been completed, and the origin and general cause of the problem have been determined.

  Action Required by Network Operator

  The network operator is responsible for employing user procedures to determine the organization responsible for problem diagnosis and further handling of the problem.

- A problem has been detected, analysis has been completed, and the origin and general cause of the problem have been determined. Problem diagnosis data has been collected but analysis of that data is not complete. The action required for problem resolution is not known.

  Action Required by Network Operator

  The network operator is responsible for employing user procedures to determine the organization responsible for problem diagnosis. The problem and the collected problem diagnosis data are turned over to that organization.

- A problem has been detected. Analysis has been completed and the origin and general cause of the problem have been determined. Problem diagnosis data has been collected and analysis of that data is complete. The action required for problem resolution is known.

  Action Required by Network Operator

  The network operator is responsible for using user procedures to determine the organization responsible for problem resolution. The problem and the collected problem resolution data are turned over to that organization.

Figure 3-5 shows the minimum function supplied by the Alert (the transport of problem data or problem-determination data). Figure 3-6 shows the maximum function supplied by the Alert (the transport of problem determination and diagnostic or problem-resolution data). The differences between the two figures have been underscored in Figure 3-6.

Note that in Figure 3-5, if complete problem-determination data is not presented to the network operator, the operator must use manual procedures to perform collection and analysis until problem determination is complete

and the problem may be given to the personnel responsible for problem diagnosis. In Figure 3-6, if problem diagnosis has been completed and complete problem-resolution data is presented to the network operator, the network operator will contact the personnel responsible for problem resolution. If problem diagnosis is not complete, the network operator will contact the personnel responsible for problem diagnosis to use manual procedures to perform collection and analysis until problem diagnosis is complete and the problem may be given to the personnel responsible for problem resolution.

```
 ............................... ______________________________ ........................
                                |      problem detection       |        "             "
                                |_____________ _______________|        |             |
                          ______________________ÿ|                     |             |
                         |                                             |            |
                         |                                             |            |
                         |      ____________________________           |            |
                         |      |  collection of problem data  |   LMS or CPMS       |
                         |      |      unique to origin        |        |            |
                         |      |_____________ _______________|        |            |
                         |                                             |            |
                         |      ____________________________           |            |
 .......................|______|       problem analysis       |...............       |
                                |_____________ _______________|        "            |
                                                                      |            |
                                ____________________________          |            |
                                |  collection of problem data  |        |          minimum
                                | unique to reporting component |  PUMS or CPMS      function
                                |_____________ _______________|        |           provided
                                                                      |            by
                                 ___________________________          |            Alert
                                | reporting of problem data or |                     |
 ..............................|  problem_determination data  |...............       |
                                |_____________ _______________|        "            |
 to problem bypass and                                                |            |
 recovery and problem                 ___________________________________          |            |
 tracking and control  ________| recording and retrieval of problem |      |            |
                 ______________ÿ| data or problem_determination data |      |            |
               |                |_________________ __________________|     CPMS           |
               |                                                         |            |
               |                                                         |            |
               |                  ___________________________            |            |
               |                 | presentation of problem data |        |            |
               |                 | or problem_determination data |       |
 .............|................|_____________ _______________|........|...................
 ____________|_____________________                                      |
| processing of requests for problem data |        |                     |
|     or problem_determination data       |        |
|_________________________________________|.......|.......................................
    "                                 _________________|                     "
     |                                                 |                     |
 requests                 ___________________________  |                     |
   for          _ÿ| collection of problem data |  |                network operator
   data        |  |_____________ _______________|  |                        |
               |                                  |                         |
               |  ___________________________    |                         |
               |__|  analysis of problem data  |_ÿ|                         |
                  |___________________________|  |
 ..............................................|.......................................
                                                                            "
                                 ___________________________                |
                              __ÿ| collection of diagnostic data |   personnel responsible
                             |   |_____________ _______________|  for problem diagnosis
                             |                                            |
                             |   ___________________________              |
                             |___| analysis of diagnostic data  |         |
                                 |_____________ _______________|         |
                                                                          |
                                 ___________________________              |
                                | reporting of diagnostic data |
 ...............................|_____________ _______________|........................
                                                                        "        "
                          ___________________________________        PUMS or CPMS    |
                         |  reporting of problem resolution data  |        |         |
                         |_____________ _____________________|                      |
 ..............................................|...................................   |
                                                                        "      function
                          ___________________________________            |      provided
                         |   recording and retrieval of problem    |     |         by
                 _____________ÿ|        problem_resolution data          |     |    resolution
               |         |_____________ _____________________|       CPMS   notification
               |                                                         |         |
               |                                                         |         |
               |          ___________________________________            |         |
               |         | presentation of problem resolution data |     |
 .............|.............|_____________ _____________________|........|.............
 ____________|_____________________                                      |
| processing of requests for problem |____requests for data                |
|     problem_resolution data        |
|____________________________________|.......................................................
```

Figure 3-5. Minimum Function Supplied by the Alert and Resolution Notification

```
 ............................... ______________________________ ........................
                 ____________________|      problem detection       |             "         "
  __________ÿ|                      |_____________ _______________|             |         |
 |                                 _____________________ÿ|                      |         |
 |                                |                                             |         |
 |   ________________             |                                             |         |
 |   |  collection of   |         |   ____________________________              |         |
```

```
|   | diagnostic data  |      |  |   collection of problem data   |         LMS or CPMS    |
|   |_________ _________|      |  |       unique to origin         |             |          |
|                              |  |_______________ _________________|             |          |
|   _________________          |                                                 |          |
|   |   analysis of    |       |   ________________________________             |          |
|___| diagnostic data  |......|__|        problem analysis        |.....................   |
    |_________ _________|       |_______________ _________________|                "          |
              |                                                                   |          |
              |                                                                   |          |
              |                   ________________________________                |          |
              |                  |   collection of problem data   |               |       maximum
              |                  | unique to reporting component  |        PUMS or CPMS  function
              |                  |_______________ _________________|               |       provided
              |                                                                   |        by
              |                 ________________________________________          |        Alert
              |______________ÿ|  reporting of problem determination and |         |          |
                              |  diagnostic or problem_resolution data  |                   |
..............................|_________________ ______________________|................   |
                                                                                  "          |
to problem bypass and         ________________________________________            |          |
recovery and problem  ______|   recording and retrieval of problem    |          |          |
tracking and control          |       determination and diagnostic      |          |          |
                 _____________ÿ|        or problem_resolution data      |          |          |
               |              |_________________ ______________________|       CPMS         |
               |                                                                  |          |
               |               ________________________________________           |          |
               |              | presentation of problem determination and |       |          |
               |              |   diagnostic or problem_resolution data   |
..............|..............|_________________ ______________________|........|............
 _____________|_____________________                |                                |
| processing of requests for problem |               |                                |
| determination and diagnostic or    |               |                                |
|     problem_resolution data        |               |
|____________________________________|...............|........................................
                "                                   |                                       "
                |                                   |                                       |
   requests for data                                |                        network operator
                                  ________________________|                                |
                                 |                        |
.........................|........................|........................................
                         |     _________________ÿ|                                       "
                         |     |                                                         |
                         |     |   ______________________________                         |
                         |     |   | collection of diagnostic data |      personnel responsible
                         |     |   |_______________ _________________|      for problem diagnosis
                         |     |                                                         |
                         |     |   ______________________________                         |
                         |     |___| analysis of diagnostic data  |                       |
                         |         |_______________ _________________|                       |
                         |                                                                 |
                         |         ______________________________                          |
                         |         | reporting of diagnostic data |
.........................|.........|_______________ _________________|...............................
                         |                                                       "          "
                         |     ________________________________________          PUMS or CPMS    |
                         |___ÿ|  reporting of problem resolution data   |         |          |
                               |_________________ ______________________|                   |
.................................................|..................................   |
                                                                                  "      function
                               ________________________________________            |      provided
                               |   recording and retrieval of problem    |          |         by
                 _____________ÿ|        problem_resolution data          |          |     resolution
               |              |_________________ ______________________|       CPMS   notification
               |                                                                  |         |
               |               ________________________________________           |         |
               |              | presentation of problem resolution data   |       |
..............|..............|_________________ ______________________|........|............
 _____________|_____________________                                              |
| processing of requests for problem |___requests for data                        |
|     problem_resolution data        |
|____________________________________|.....................................................
```

Figure 3-6. Maximum Function Supplied by the Alert and Resolution Notification

---

1.3.4.4 Implementation

Alert and resolution notification support is provided in an entry point by the "EP ALERT Function Set" in topic 2.10.1. Alert support is a base function implemented by all nodes. Resolution notification support is an optional subset of the EP_ALERT Function Set.

1.3.4.5 Format Usage

```
NETWORK MANAGEMENT VECTOR TRANSPORT (NMVT)
     Alert Major Vector (X'0000')
     Flow:   From PU to SSCP (Normal)


CONTROL POINT MGMT. SERVICES UNIT (CP-MSU)
```

```
Alert Major Vector (X'0000')
Flow:   From CP to CP
```

An Alert may be initiated when any nodal component detects a condition indicating the loss or impending loss of availability of a system resource to an end user. It is thus a flow of the type described in "Unsolicited Flows" in topic 1.2.2.1. Information to be included in an Alert (X'0000') major vector is passed from the initiating LMS to its local PUMS or local CPMS. The Alert major vector contains the type of the Alert, a general classification and cause of the condition being reported, the identity of the resource to which the Alert applies, and additional information to assist in problem determination. PUMS constructs an NMVT containing this major vector and sends it to CPMS on its PU-SSCP session.

When CPMS receives the NMVT, it optionally logs the data, and passes it to the network operator. Refer to Figure 3-7 for an example flow.

```
Network                                                  LMS
Operator          CPMS                PUMS             Component
   .                .                   .                  .
   .                .                   .                  .
   .                .                   .     Alert data   .
   .                .                   _____________________o
   .                .   NMVT (X'0000')  .                  .
   .                _________________o                    .
   .                .   +RSP(NMVT)      .                  .
   .                o_________________ÿ                   .
   .   Alert data   .                   .                  .
  _________________o                   .                  .
   .                .                   .                  .
   .                .                   .                  .
```

Figure 3-7. Example Flow Showing an Alert

If CPMS in an APPN node builds the Alert, it sends the Alert major vector in a CP-MSU to its Alert focal point. Refer to Figure 3-8 for an example flow.

```
Network         Alert Focal Point          APPN Node                    LMS
Operator              CPMS                    CPMS                    Component
   .                  .                        .                         .
   .                  .                        .                         .
   .                  .                        .                         .
   .                  .                        .        Alert data       .
   .                  .                        ____________________________o
   .                  .                        .                         .
   .                  .                        .                         .
   .                  .    CP-MSU (X'0000')    .                         .
   .                  _________________________o                        .
   .    Alert data    .                        .                         .
 ___________________o                         .                         .
   .                  .                        .                         .
   .                  .                        .                         .
```

Figure 3-8. Example Flow Showing Alert Originating in an APPN Node. The CP-MSU is transported from the APPN node to its focal point over the CP-CP or LU-LU sessions used by multiple-domain support.

Figure 3-9 illustrates a special type of Alert, the *delayed* Alert. If the condition causing the Alert interrupts communication between PUMS and CPMS, the Alert cannot be sent immediately. In this case, PUMS stores the Alert until communication with CPMS has been re-established. It then sends the Alert to CPMS.

Delayed Alerts are not supported in the APPN environment.

```
Network                                                  LMS
Operator          CPMS                PUMS             Component
   .                .                   .                  .
   .                .                   .                  .
   .                .  x             x  .            error * detected
   .                .  x             x  .     Alert data   .
   .                .  x             x  _____________________o
   .                .  x             x  .                  .
   .                .  x             x  .                  .
   .                .  x   session   x  .                  .
   .                .  x interrupted x  .                  .
   .                .  x             x  .                  .
   .                .  x             x  .                  .
   .                .  x             x  .                  .
   .                .   NMVT (X'0000')  .                  .
   .                _________________o                    .
```

```
          .                .   +RSP(NMVT)      .                     .
          .                °___________________ÿ                    .
          .   Alert data   .                   .                     .
          _________________°                  .                     .
          .                .                   .                     .
          .                .                   .                     .
```

Figure 3-9. Example Flow Showing Delayed Alert. The Alert is reporting a condition that caused it to be delayed, in this case a condition that interrupted the SSCP-PU session.

Related to delayed Alerts are *held* Alerts. Like a delayed Alert, a held Alert cannot be sent immediately because of an interruption in communication between the detecting node and the node to which the Alert was to be sent (controlling CPMS, or focal point). The held Alert, however, does not report the condition that caused the interruption; it instead reports some (typically) unrelated condition, that just happened to be detected when communication was interrupted. The key difference between held and delayed Alerts lies in the fact that if a delayed Alert successfully reaches CPMS, then it follows that the problem it reports has been either resolved or bypassed, at least temporarily. Otherwise, the delayed Alert could not have reached CPMS. With a held Alert, no such inference is possible: the problem it reports may have been resolved or bypassed, but it is equally possible that it is still impacting availability.

NETWORK MANAGEMENT VECTOR TRANSPORT (NMVT)
Resolution Major Vector (X'0002')
Flow: From PU to SSCP (Normal)

CONTROL POINT MGMT. SERVICES UNIT (CP-MSU)
Resolution Major Vector (X'0002')
Flow: From CP to CP

In a manner similar to Alerts, any nodal component may initiate a resolution notification upon detecting that an error condition has been corrected. Information to be included in the Resolution (X'0002') major vector is passed from the initiating LMS to its local PUMS or CPMS component. The Resolution major vector contains the identification of the resource that was actually modified to resolve the error condition, an optional identification token that correlates the corrective action to previously reported Alerts, and information that specifies how the correction was made (if known). The Resolution major vector is transported to the Alert focal point in the same manner as described previously for Alerts.

Refer to Figure 3-10 for an example flow.

```
       Network                                                  LMS
       Operator            CPMS                PUMS            Component
          .                 .                   .                  .
          .                 .                   .                  .
          .                 .                   .  Resolution data .
          .                 .                   ___________________°
          .                 .   NMVT (X'0002')  .                  .
          .                 ___________________°                  .
          .                 .   +RSP(NMVT)      .                  .
          .                 °___________________ÿ                  .
          . Resolution data .                   .                  .
          _________________°                   .                  .
          .                 .                   .                  .
          .                 .                   .                  .
```

Figure 3-10. Example Flow Showing a Resolution Notification

Figure Figure 3-11 shows what occurs in APPN networks. CPMS in an APPN node builds the Resolution major vector and it sends the Resolution major vector in a CP-MSU to its Alert focal point.

```
   Network         Alert Focal Point            APPN Node                    LMS
   Operator              CPMS                      CPMS                   Component
      .                  .                         .                         .
      .                  .                         .                         .
      .                  .                         .                         .
      .                  .                         .     Resolution data     .
      .                  .                         ___________________________°
      .                  .                         .                         .
      .                  .                         .                         .
      .                  .    CP-MSU (X'0002')     .                         .
      .                  ___________________________°                       .
      . Resolution data  .                         .                         .
     ___________________°                         .                         .
      .                  .                         .                         .
```

```
      .                   .                         .                       .
```

Figure 3-11. Example Flow Showing a Resolution Notification Originating in an APPN Node. The CP-MSU is transported from the APPN node to its focal point over the CP-CP or LU-LU sessions used by multiple-domain support.

Multiple Alert and/or Resolution Major Vectors per CP-MSU:

Multiple Alert and/or Resolution major vectors may be included in a CP-MSU flowing from an APPN node to an Alert focal point. Multiple Alert and/or Resolution major vectors may *not* be included in an NMVT. Figure 3-12 shows an example of a CP-MSU containing major vectors of both types.

```
   Network         Alert Focal Point                              APPN Node                       LMS
   Operator              CPMS                                        CPMS                       Component
      .                   .                                           .                             .
      .                   .                                           .                             .
      .                   .                                           .                             .
      .                   .                                           .    Alert data :             .
      .                   .                                           .      Incidents D, E, F      .
      .                   .                                           _____________________________°
      .                   .                                           .                             .
      .                   .                                           .                             .
      .                   .                                           .    Resolution data :        .
      .                   .                                           .      Incidents B, C, D      .
      .                   .                                           _____________________________°
      .                   .                                           .                             .
      .                   .                                           .                             .
      .                   .  CP-MSU (X'0002':C, X'0000':E, X'0000':F, .                             .
      .                   .          X'0000':D, X'0002':B, X'0002':D) .                             .
      .                   ________________________________________°                             .
      . Alert and         .                                           .                             .
      .   Resolution data .                                           .                             .
     __________________°                                            .                             .
      .                   .                                           .                             .
      .                   .                                           .                             .
```

Figure 3-12. Example Flow Showing Alerts and Resolution Notifications Originating in an APPN Node

In Figure 3-12, Alert major vectors for incidents D, E, and F, and Resolution major vectors for incidents B, C, and D are being carried in the same CP-MSU (the CP-MSU is transported from the APPN node to its focal point over the CP-CP or LU-LU sessions used by multiple-domain support).

The only restriction on the ordering of Alert and Resolution major vectors in a CP-MSU is that if it can be determined that an Alert and a Resolution major vector for the same incident are present, then the Alert must precede the Resolution major vector. Thus in this example, the Alert for incident D must precede the Resolution major vector for incident D. Many possible orderings of the major vectors will satisfy this requirement; one such ordering is shown.

The letters B-F are used as incident identifications in this example for simplicity only. See *SNA Formats* for the complete description of the Incident Identification (X'4A') MS common subvector.

1.3.5 Managing Problems in APPN Networks

Problems are managed in APPN networks in much the same way that they are managed in subarea networks. In particular, Alerts provide a network operator with unsolicited notification of problems detected by APPN nodes just as they do for problems detected by PUs in a subarea network. A specific set of Alerts has been defined for reporting APPN protocol errors. These Alerts are defined in "Alerts for APPN Nodes" in topic A.5.

Problem management in APPN networks makes use of the general scheme of focal points and spheres of control defined by the management services architecture. Focal points are established for centralized management and control of APPN networks. For problem management, an Alert focal point is established. For session concentration reasons, an Alert focal point in an APPN network typically has only network nodes in its sphere of control; an Alert originating in an end node is forwarded to an Alert focal point by the end node's network node server.

More discussion of focal points can be found in "Properties of Focal Points" in topic 1.2.4.2. The transport mechanism for cross-domain communication is discussed in "Transport of Non-Bulk Management Services Data in APPN Networks" in topic 1.2.3.2.

1.3.6 Link Services

Link services assists in problem determination and problem diagnosis for problems associated with communication links.

Subtopics:

- 1.3.6.1 Managing Links

1.3.6.1 Managing Links

Subtopics:

- 1.3.6.1.1 Links Traversing X.21 Networks
- 1.3.6.1.2 SDLC Links
- 1.3.6.1.3 Links Traversing Local Area Networks
- 1.3.6.1.4 Links Traversing X.25 Packet-Switched Data Networks

1.3.6.1.1 Links Traversing X.21 Networks

There are two environments to consider, links traversing X.21 networks in a hierarchical environment and links traversing X.21 networks in a peer environment.

```
    SNA primary          _________________          SNA secondary
     ________           |                 |          __________
    | type 4  |         |  inter_exchange |         |  remote  |
    |  node   |_______ÿ|     network     |_______ÿ|  node    |
    |         |         |                 |         |          |
    |_________|         |_________________|         |__________|

              |_______ÿ|                 |_______ÿ|            local loops


              |_____________________________________ÿ|            X.21 network
```

Note: The remote node can be a boundary-function-attached type 2.0 or 2.1 node

Figure 3-13. Boundary Function Attachment Over an X.21 Network in a Hierarchical Environment

Functions Provided (Hierarchical Environment):

Figure 3-13 shows a type 4 node communicating with a boundary-function-attached type 2.0 or 2.1 node over a link established by means of an X.21 network. For an introduction to X.21 networks see *CCITT Recommendation, X.21*, 1984.

The management services support unique to X.21 link connections is limited to one function: issuing Alerts for such conditions as unsuccessful connection attempts and network timeouts. The boundary-function-attached type 2.0 or 2.1 node issues a delayed Alert for an unsuccessful short-hold mode *reconnection*. The delayed Alert is sent only if the type 4 node with which the sender establishes a connection is the same one to which the reconnection attempt was directed; if a different type 4 node is reached, no Alert is sent.

No Alert is issued by a boundary-function-attached type 2.0 or 2.1 node for an unsuccessful initial connection, since at this point the sending node would be unknown to the network, and thus the Alert would serve no purpose. (For a description of the format that supports the establishment of a short-hold mode connection, see the Format 1 continuation of XID, in *SNA Formats*)

```
                         _________________
     ________           |                 |          __________
    |  APPN   |         |  inter_exchange |         |   APPN   |
    |  node   |_______ÿ|     network     |_______ÿ|  node    |
    |         |         |                 |         |          |
    |_________|         |_________________|         |__________|
```

```
|_______ÿ|                  |_______ÿ|            local loops


|_____________________________________ÿ|            X.21 network
```

Figure 3-14. Nodes Communicating Over an X.21 Network in a Peer Environment

Functions Provided (Peer Environment):

Figure 3-14 shows two APPN nodes communicating over a link established by means of an X.21 network.

As in the hierarchical environment, the management services support unique to X.21 link connections is limited to issuing Alerts for such conditions as unsuccessful connection attempts and network timeouts. An APPN node does not identify itself as SNA primary or SNA secondary. Both nodes on the link send Alerts on both unsuccessful reconnections or unsuccessful initial connections. If they support the option for doing so, they send held Alerts if they do not currently have a session with a focal point, otherwise they send an Alert immediately. (For a description of the format that supports the establishment of a short-hold mode connection, see the Format 3 continuation of XID, in *SNA Formats*.)

Implementation:

Alert support in the boundary-function-attached type 2.0 or 2.1 node includes Alerts issued for the X.21 network. See "EP_ALERT Optional Subset 9 (X.21 Alert)" in topic 2.10.1.13.

Format Usage:

The only format involved in the management of the links traversing X.21 networks is the Alert. See "X.21 and X.21 Short-Hold Mode Alerts" in topic A.3.1 for a list of the Alerts defined for links traversing X.21 networks.

1.3.6.1.2 SDLC Links

```
 _______________ |                            _______________
|               |                            |               |
| Type 2.1 Node |____________________________| Type 2.1 Node |
|               |         peer-to-peer       |               |
|_______________|          SDLC link         |_______________|
```

Figure 3-15. Peer-to-Peer SDLC Link Between Type 2.1 Nodes

Functions Provided:

Figure 3-15 shows a peer-to-peer SDLC link between two type 2.1 nodes. In this configuration, each type 2.1 node sends an Alert about a problem on the link between them if it has either an SSCP-PU session with a control point or an Alert focal point. If the SSCP-PU session does not exist or if the node does not have a focal point at the time the Alert was created, the node may elect to hold the Alert until it can be sent (when communication with an SSCP or a focal point is available).

Implementation:

Alert support in a type 2.1 node includes Alerts issued for the SDLC logical link with its peer type 2.1 node. See "EP_ALERT Optional Subset 8 (SDLC/LAN LLC Alert)" in topic 2.10.1.12.

Format Usage:

The only format involved in the management of the SDLC logical link between a type 2.1 node and its peer type 2.1 node is the Alert. See "SDLC Alerts" in topic A.2.2 for a list of the Alerts defined for SDLC logical links.

1.3.6.1.3 Links Traversing Local Area Networks

The next four sections discuss four types of management for local area networks (LANs):

- Medium-access control-(MAC-) level management for token-ring LANs
- MAC-level management for carrier sense multiple access with collision detection (CSMA/CD) LANs
- MAC-level management for bridged LANs; the LAN bridges may connect two token rings, two CSMA/CD busses, or one ring and one bus
- Logical link control-(LLC-) level management for logical links traversing any of the three types of LANs listed above

Management of links traversing local area networks involves management at both the MAC and LLC levels. Thus, a node responsible for managing such a link must implement the functions described in "LLC-Level Management for Links Traversing Local-Area Networks" in addition to those described for the appropriate MACs.

MAC-Level Management for Token-Ring LANs:

- RS denotes ring station
- MS session denotes either an SSCP-PU session with a CP or an active Alert focal point (accessible over the CP-CP or LU-LU sessions used by multiple-domain support)

Figure 3-16. Nodes Communicating Over a Token-Ring LAN

Functions Provided:

Figure 3-16 shows an example of a token-ring LAN and its attached nodes. Five nodes attached to the LAN are shown, each with a different role in the management process:

- Type 2.1 node with a management services session
  - Issues Alerts concerning the local LAN adapter (hardware check, programming error, and removal from the ring), wire faults, and

ring inoperative conditions. See "Problem Determination, Diagnosis, and Resolution via Alerts and Resolution Notifications" in topic 1.3.4 for details on Alerts.

Note: The Alerts reporting the ring inoperative conditions are not sent by this node if a LAN manager is managing the ring to which it is attached. Knowledge of whether a LAN manager is present on a ring is defined locally at each Alert sender on the ring; there is no mechanism for acquiring this knowledge dynamically.

- Participates in the MAC-level management functions required for management and operation of the ring. See *Token-Ring Network Architecture Reference* for details on the operation of token-ring LANs.

° Type 2.0 node:

- Participates in the MAC-level management functions required for management and operation of the ring

° Type 2.1 node (without a management services session):

- Participates in the MAC-level management functions required for management and operation of the ring

° Non-SNA node:

- Participates in the MAC-level management functions required for management and operation of the ring

° LAN manager

- Issues Alerts concerning the local LAN adapter (hardware check, programming error, and removal from the ring), wire faults, remote LAN adapter removal from the ring, ring inoperative conditions, excessive soft errors, and program errors in the LAN manager (see "Problem Determination, Diagnosis, and Resolution via Alerts and Resolution Notifications" in topic 1.3.4 for details on Alerts)

- The LAN Alert transport function is contained within the LAN manager and provides an "Alert pass-through" for sending Alerts to an SSCP or to an Alert focal point. Devices and software on the LAN that experience error conditions can build Alert messages and send them over the LAN to the LAN manager. The LAN manager receives the Alert message and sends an Alert to the SSCP or Alert focal point.

- Participates in the MAC-level management functions required for management and operation of the ring

Implementation:

Alert support in the LAN manager and in the type 2.1 nodes includes Alerts issued for the token-ring LAN link connection. See "EP ALERT Optional Subset 7 (LAN Alert)" in topic 2.10.1.11.

Format Usage:

The only format involved in the MAC-level management of a token-ring LAN is the Alert. See "Token-Ring LAN Alerts" in topic A.1.1 for a list of the Alerts defined for a node providing MAC-level management for a token-ring LAN.

MAC-Level Management for CSMA/CD LANs:

```
 ____________                                  ____________
|type 2.1 node|                               | LAN manager |
|   with MS   |                               |     node    |
|   session   |                               |             |
|             |                               |             |
|    ____     |                               |    ____     |
|   |  S  |   |                               |   |  S  |   |
|   |__ __|   |                               |   |__ __|   |
|_____|_____|          CSMA/CD bus            |_____|_____|
|_______|____ _________________ _________________ ________|________|
```

```
 ______|______      ______|______      ______|_______
|     __|__    |    |     __|__    |    |     __|__    |
|    |  S  |   |    |    |  S  |   |    |    |  S  |    |
|    |_____|   |    |    |_____|   |    |    |_____|    |
|             |    |             |    |                 |
|type 2.0 node|    |type 2.1 node|    | non_SNA node    |
|_____________|    |_____________|    |_________________|
```

° S denotes station
° MS session denotes either an SSCP-PU session with a CP or an active Alert focal point (accessible over the CP-CP or LU-LU sessions used by multiple-domain support)

Figure 3-17. Nodes Communicating Over a CSMA/CD LAN

Functions Provided:

Figure 3-17 shows an example of a CSMA/CD LAN and its attached nodes. Five nodes attached to the LAN are shown, each with a different role in the management process:

° Type 2.1 node with a management services session

- Issues Alerts concerning the local LAN adapter (hardware check, programming error, removal from the bus), lack of signal at the adapter, and bus-inoperative conditions. See "Problem Determination, Diagnosis, and Resolution via Alerts and Resolution Notifications" in topic 1.3.4 for details on Alerts.

  Note: The Alerts reporting the bus-inoperative conditions are not sent by this node if a LAN manager is managing the bus to which it is attached. Knowledge of whether a LAN manager is present on a bus is defined locally at each Alert sender on the bus; there is no mechanism for acquiring this knowledge dynamically.

- Participates in the MAC-level management functions required for management and operation of the bus

° Type 2.0 node:

- Participates in the MAC-level management functions required for management and operation of the bus

° Type 2.1 node (without a management services session):

- Participates in the MAC-level management functions required for management and operation of the bus

° Non-SNA node:

- Participates in the MAC-level management functions required for management and operation of the bus

° LAN manager

- Issues Alerts concerning the local LAN adapter (hardware check, programming error, removal from the bus), lack of signal at the adapter, bus-inoperative conditions, unauthorized adapter insertion attempts, and removal of remote adapters from the bus. See "Problem Determination, Diagnosis, and Resolution via Alerts and Resolution Notifications" in topic 1.3.4 for details on Alerts.

- The LAN Alert transport function is contained within the LAN manager and provides an "Alert pass-through" for sending Alerts to the SSCP or Alert focal point. Devices and software on the LAN that experience error conditions can build Alert messages and send them over the LAN to the LAN manager. The LAN manager receives the Alert message and sends an Alert to the focal point.

- Participates in the MAC-level management functions required for management and operation of the bus.

Implementation:

Alert support in the LAN manager and in the type 2.1 nodes includes Alerts issued for the CSMA/CD LAN link connection. See "EP_ALERT Optional Subset 7 (LAN Alert)" in topic 2.10.1.11.

Format Usage:

The only format involved in the MAC-level management of a CSMA/CD LAN is the Alert. See "CSMA/CD LAN Alerts" in topic A.1.2 for a list of the Alerts defined for a node providing MAC-level management for a CSMA/CD LAN.

MAC-Level Management for Bridged LANs:

- RS denotes ring station
- M-server denotes LAN management server
- MS session denotes either an SSCP-PU session with a CP or an active Alert focal point (accessible over the CP-CP or LU-LU sessions used by multiple-domain support)

Figure 3-18. Nodes Communicating Over a Bridged Token-Ring LAN. This figure shows a bridge between two token-rings; a bridge may also be between two CSMA/CD busses, or between a token-ring and a CSMA/CD bus.

Functions Provided:

Figure 3-18 shows an example of a bridged LAN and its attached nodes. Five types of nodes, plus the bridge, are shown attached to the LAN, but the only ones with specific responsibilities for managing the bridged LAN are the LAN manager and those nodes attached to the remote ring (or bus) that contain a LAN management server:

- A node that is attached to the remote ring or bus and contains a LAN management server; this node may, but need not be, the bridge itself

  - Assembles and analyzes MAC-level from the other nodes attached to its ring or bus, and forwards the results of this analysis to the LAN manager

- LAN manager

- Issues Alerts concerning the bridges and remote rings or busses it is responsible for, as well as its logical connections to the LAN management servers that provide it data about these bridges, rings, and busses. See "Problem Determination, Diagnosis, and Resolution via Alerts and Resolution Notifications" in topic 1.3.4 for details on Alerts.

- The LAN Alert transport service is contained within the LAN manager and provides an "Alert pass-through" for sending Alerts to the focal point. Devices and software on the LAN that experience error conditions can build Alert messages and send them over the LAN to the LAN manager. The LAN manager receives the Alert message and sends an Alert to the focal point.

- Participates in the MAC-level management functions required for management and operation of the bridged LAN

Implementation:

Alert support in the LAN manager includes Alerts issued for the bridged LAN.

Format Usage:

The only format involved in the MAC-level management of a bridged LAN is the Alert. See "Bridged LAN Alerts" in topic A.1.3 for a list of the Alerts defined for a LAN manager providing MAC-level management for a bridged LAN.

LLC-Level Management for Links Traversing Local-Area Networks

- RS denotes ring station
- BF-attached denotes boundary-function-attached
- ..... denotes a peer-to-peer (type 2.1 to type 2.1) logical link connection

Figure 3-19. Logical Links Traversing a Token-Ring LAN. While the figure shows a token-ring LAN, CSMA/CD and bridged LANs also support such logical links. In a bridged LAN, a logical link may traverse a bridge.

Functions Provided:

Figure 3-19 shows an example of a local-area network and a peer-to-peer logical link, between two type 2.1 nodes, that traverses it. The type 2.1 node with a management services session is responsible for issuing Alerts on the peer-to-peer logical link. Note that both type 2.1 nodes on a peer-to-peer logical link may have management services sessions with an Alert focal point or an SSCP.

Implementation:

Alert support in the boundary-function-attached type 2.1 node includes Alerts issued for logical link connections traversing a LAN. See "EP_ALERT Optional Subset 8 (SDLC/LAN LLC Alert)" in topic 2.10.1.12.

Format Usage:

The only format involved in the management of LAN logical link connections is the Alert. See "LAN LLC Alerts" in topic A.2.1 for a list of the Alerts defined for a node providing LLC-level management for links that traverse LANs.

1.3.6.1.4 Links Traversing X.25 Packet-Switched Data Networks

```
                          _________________
 _________               |                 |          __________
| type 4  |              | packet_switched |         |  remote  |
|  node   |_______ÿ|     | data network    |_______ÿ|   node   |
|         |              |                 |         |          |
|_________|              |_________________|         |__________|

X.25 layer

   (1)    |_______ÿ|  physical layer  |_______ÿ|            local loop
   (2)    |_______ÿ|    link layer    |_______ÿ|            access link
   (3)    |_______ÿ|   packet layer   |_______ÿ|            logical channel

SNA X.25 layer

   (3a)   |___________   LLC layer  ___________ÿ|            logical link


                              or

OSI layer

   (4)    |________  transport layer  _________ÿ|            transport


Note:  The remote node can be either a type 2.0 node or
       a boundary-function-attached type 2.1 node
```

Figure 3-20. Boundary Function Attachment Over an X.25 Network

Functions Provided:

Figure 3-20 shows a type 4 node communicating with a type 2.0 node or a type 2.1 boundary-function-attached node over a link established by means of an X.25 Packet-Switched Data Network. For an introduction to X.25 networks, see *The X.25 1984 Interface for Attaching SNA Nodes to Packet-Switched Data Networks General Information Manual* Management services support is provided for X.25 connections by providing for the issuance of Alerts.

Implementation:

The Alert support required for management services of X.25 link connections may be found in "EP_ALERT Optional Subset 11 (X.25 Alert)" in topic 2.10.1.15.

Format Usage:

The only format involved in the management of X.25 link connections is the Alert. See "Alerts for X.25 Link Connections" in topic A.4 for a list of the Alerts defined for a node that implements the X.25 protocols. The flow is of the type described in "Unsolicited Flows" in topic 1.2.2.1.

1.4 Chapter 4. Performance and Accounting Management

Performance and accounting management is the process of quantifying, measuring, reporting, and controlling the responsiveness, availability, utilization, and costs of a data processing or information system. Response-time monitoring is the only element of performance and accounting management for which SNA management services have been provided.

Subtopics:

- 1.4.1 Response-Time Monitoring (RTM)

1.4.1 Response-Time Monitoring (RTM)

Response-time monitoring (RTM) provides the capability to quantify, measure, and report end-user response times for dependent LUs. Refer to Figure 4-1 for an overview of the steps required for RTM and the components in which the steps are implemented.

```
         events
           |

 ____________________ ....................................................................
|    collection and    |_____________________________________________ |                 "
|     analysis of      |                                              |                 |
|  response_time data  |____________________                          |              LU LMS
|_________ __________|                     |                         |                 |
          |                                |                         |                 |
                                          |                         |
 ____________________ ......................|.........................|...................
|  collection of data  |                    |                         |                 "
| unique to reporting  |                    |                         |                 |
|         PU           |                    |                         |              PUMS
|_________ __________|                    |                         |                 |
          |                                |                         |                 |
                                  ___________|_________    ___________|___________    |
 ____________________             |  forwarding requests  |   |  forwarding requests to  |   |
|    reporting of      |.........|   for response_time   |...|   set or modify response_ |
|  response_time data  |         |        data           |   |    time collection or    |.......
|_________ __________|         |_______________________|   |   reporting parameters   |   "
          |                                  "                |__________________________|   |
                                            |                        "                     |
 ____________________                       |                       |                     |
|    recording and     | ________           |                       |                     |
|    retrieval of      |          |         |                       |                     |
|  response_time data  |          |         |                       |                  CPMS
|_________ __________|          |         |                       |                     |
          |                     |         |                       |                     |
          |                     |         |                       |                     |
          |                     |         |                 ______|_________            |
                                |         |          |  processing of network   |      |
 ____________________      _____|_________|_____     |  operator requests to    |      |
|   presentation of    |    |  processing of network  |   |  set or modify response_ |      |
|  response_time data  |    |  operator requests for  |   |  time collection or      |      |
| to network operator  |    |  response_time data     |   |  reporting parameters    |
|____________________|....|_________________________|...|__________________________|..........
```

Figure 4-1. Overview and Placement of Response-Time Monitoring Steps

RTM provides for measurement of response time at the workstation, allows correlation of transaction timing with application program usage, allows network owners to define how response times are to be measured, and allows the network operator to have real-time access to the response-time data.

RTM is performed automatically, although a mechanism may exist for a human operator to change the collection or reporting parameters, or to request response-time data.

RTM is only supported for dependent LUs.

Subtopics:

- 1.4.1.1 Functions Provided
- 1.4.1.2 Implementation
- 1.4.1.3 Format Usage

1.4.1.1 Functions Provided

Subtopics:

- 1.4.1.1.1 Collection and Analysis of Response-Time Data
- 1.4.1.1.2 Reporting of Response-Time Data
- 1.4.1.1.3 Recording and Retrieval of Response-Time Data
- 1.4.1.1.4 Presentation of Response-Time Data
- 1.4.1.1.5 Processing and Forwarding of Network Operator Requests

---

1.4.1.1.1 Collection and Analysis of Response-Time Data

The collection and analysis function of RTM provides the following:

- Measurement and collection of end-user response times at the workstation or controller according to definitions specified by the network operator (See "Processing and Forwarding of Network Operator Requests" in topic 1.4.1.1.5.)
- Collection of data that will identify any potential inaccuracy of the response-time data, e.g., counter overflow or the activation of a new session before the data from the previous session could be transmitted
- Collection of data that will allow the correlation of response times with application program usage, e.g., identification of primary and secondary LU
- Measurement of the total time period over which the data was collected.

1.4.1.1.2 Reporting of Response-Time Data

The reporting function of RTM provides for reporting of response-time data to the RTM focal point under the following conditions:

- When solicited by the network operator
- Unsolicited on session deactivation, if specified by a network operator command
- Unsolicited on counter overflow, if specified by a network operator command.

1.4.1.1.3 Recording and Retrieval of Response-Time Data

The recording and retrieval function of RTM is implemented at the focal point and provides the following:

- Determines if the response-time data is to be logged
- Provides data base management (e.g., storage, retrieval) of the collected data.

1.4.1.1.4 Presentation of Response-Time Data

The presentation function of RTM is implemented at the focal point and provides the formatting and presentation of response-time data to allow the network operator to view the following data:

- A summary of the data for a specific LU over a user-defined time period
- Detailed data for a specific session for a single collection period

- Long-term trend for a specific LU.

1.4.1.1.5 Processing and Forwarding of Network Operator Requests

This function provides a protocol boundary for the network operator to request the following:

- Presentation of RTM data that is either requested from the RTM historical data base or from the collection and analysis function.

- The setting of the following parameters in the collection and analysis function:

  - The conditions under which unsolicited RTM data is reported

    The following options are available:

    -- Report on session deactivation

    -- Report on counter overflow.

  - The response-time unit of measure

  - The number and size of the ranges in which response times are to be reported

    For example, range 1 could be defined as all transactions with a response time of less than one second, range 2 as transactions falling within one to two seconds, range 3 as two to four seconds, and so forth. For each transaction, the appropriate range counter is updated. Up to four response-time boundaries, defining up to five ranges, may be set, and up to five response-time counters will be supported.

  - The definition of response times

    The following measurement options are available:

    -- From the attention or action key depression to the arrival back at the LU of the first character that can alter the presentation space

    -- From the attention or action key depression until the LU is ready to accept input from its end user

    -- From the attention or action key depression to the receipt and processing back at the LU of Change Direction (CD) or End Bracket (EB)

    -- From the attention or action key depression to the receipt of the last character of the last message received prior to the next attention or action key depression.

1.4.1.2 Implementation

Support of RTM is provided in a PU by "EP_RTM_Function_Set" in topic 2.10.2. This is an optional function set implemented by type 2.0 nodes or boundary-function-attached 2.1 nodes (for dependent LUs).

1.4.1.3 Format Usage

```
NETWORK MANAGEMENT VECTOR TRANSPORT (NMVT)
   Request Response-Time Monitor MV (X'8080')
   Flow:  From SSCP to PU T2.0 (Normal)

NETWORK MANAGEMENT VECTOR TRANSPORT (NMVT)
   Response-Time Monitor MV (X'0080')
   Flow:  From PU T2.0 to SSCP (Normal)
```

Subtopics:

- 1.4.1.3.1 Setting of RTM Parameters
- 1.4.1.3.2 Requesting RTM Data and Status
- 1.4.1.3.3 Sending Unsolicited RTM Data and Status

1.4.1.3.1 Setting of RTM Parameters

A network operator can request that RTM collection and reporting parameters be set to specified values for all LUs associated with a particular PU or for a specified LU. Data necessary for the request is passed to CPMS.

CPMS formats the request into an NMVT containing a Request RTM (X'8080') major vector and sends it to PUMS. The flow is of the type described in "Request Without Reply Flows" in topic 1.2.2.2.

When PUMS receives the NMVT it does the following:

- If the request was for a single LU, PUMS sends a request to the specified LU. Refer to Figure 4-2 for an example flow.

- If the request applies to all LUs, PUMS determines the list of LUs configured at the node, and sends requests to each of them. The flow is similar to Refer to Figure 4-3 for an example flow.

```
Network
Operator              CPMS              PUMS                              LU1
  .                    .                 .                                 .
  .    request to      .                 .                                 .
  . set RTM parameters .                 .                                 .
  °____________________ÿ                 .                                 .
  .     (for LU1)      .  NMVT (X'8080')  .                                 .
  .                    °__________________ÿ                                 .
  .                    .  +RSP(NMVT)      .                                 .
  .                    ___________________                                 .
  .                    .                 .   request to set RTM parameters  .
  .                    .                 °__________________________________ÿ
  .                    .                 .                                 .
  .                    .                 .                                 .
  .                    .                 .                                 .
```

Figure 4-2. Example Flow Showing Request to Set RTM Parameters for a Single LU

```
Network
Operator              CPMS              PUMS                 LU1    LU2 ...LUn
  .                    .                 .                    .      .      .
  .    request to      .                 .                    .      .      .
  . set RTM parameters .                 .                    .      .      .
  °____________________ÿ                 .                    .      .      .
  .   (for all LUs)    . NMVT (X'8080')  .                    .      .      .
  .                    °__________________ÿ                    .      .      .
  .                    .  +RSP(NMVT)      .                    .      .      .
  .                    ___________________°                  .      .      .
  .                    .                 .   request to       .      .      .
  .                    .                 . set RTM parameters .      .      .
  .                    .                 °____________________ÿ      .      .
  .                    .                 .                    .      .      .
  .                    .                 .                    .      .      .
  .                    .                 .   request to       .      .      .
  .                    .                 . set RTM parameters .      .      .
  .                    .                 °__________________________ÿ      .
  .                    .                 .                    .      .      .
  .                    .                 .            °       .      .      .
  .                    .                 .            °       .      .      .
  .                    .                 .            °       .      .      .
  .                    .                 .                    .      .      .
  .                    .                 .   request to       .      .      .
  .                    .                 . set RTM parameters .      .      .
  .                    .                 °__________________________________ÿ
  .                    .                 .                    .      .      .
  .                    .                 .                    .      .      .
```

Figure 4-3. Example Flow Showing Request to Set RTM Parameters for All LUs

---

1.4.1.3.2 Requesting RTM Data and Status

A network operator can request RTM data from a PU for a single LU or for all LUs with accumulated data. Data necessary for the request is passed to CPMS.

CPMS formats the request into an NMVT containing a Request RTM (X'8080') major vector and sends it to PUMS. The flow is of the type described in "Request/Reply Flows for Non-Bulk Data" in topic 1.2.2.3.

When PUMS receives the NMVT it does the following:

- If the request was for a single LU, PUMS sends a request to the specified LU. If the LU currently has RTM data, both accumulated data and status are returned; if it has no RTM data, or if it is inactive, only status information is returned. Refer to Figure 4-4 for an example flow of the request for RTM data from a single LU.

- If the request applies to all LUs with accumulated data, PUMS determines the list of LUs configured at the node, and sends requests to each of them. Each LU responds to this request with its RTM status; if the LU currently has RTM data, it includes this data as well. For each LU that responds with both accumulated data and status, PUMS immediately builds an NMVT containing an RTM (X'0080') major vector and sends it to CPMS.

  PUMS discards responses from LUs that have no data, with the possible exception of the last one. An NMVT containing an RTM (X'0080') major vector with RTM status information may, if PUMS so elects, *always* be sent for the last LU, even if this LU returns no data to PUMS. By always sending a reply for the last LU, PUMS is able to send replies for preceding LUs immediately, and thereby avoid the expenses associated with lookahead; otherwise it would have to hold each reply until it had determined whether or not that reply would be the last one for the request.

  If *no* LU has accumulated RTM data, it sends a single reply containing RTM status information for its last configured LU. Refer to Figure 4-5 and Figure 4-6 for example flows.

When CPMS receives each NMVT, it optionally logs the data and passes it to the network operator.

```
Network
Operator              CPMS                PUMS                        LU1
  .                     .                   .                          .
  .    request for      .                   .                          .
  . RTM data and status .                   .                          .
  °_____________________ÿ                   .                          .
  .     (for LU1)       .                   .                          .
  .                     .   NMVT (X'8080')  .                          .
  .                     °_________________ÿ                           .
  .                     .    +RSP(NMVT)     .                          .
  .                     ___________________°     request for          .
  .                     .                   .   RTM data and status    .
  .                     .                   °___________________________ÿ
  .                     .                   .                          .
  .                     .                   .                          .
  .                     .                   .   RTM data and status    .
  .                     .                   .      or RTM status       .
  .                     .                   ___________________________°
  .                     .                   .                          .
  .                     .   NMVT (X'0080')  .                          .
  .                     _________________°                             .
  .                     .    +RSP(NMVT)     .                          .
  . RTM data and status °_____________________ÿ                         .
  .    or RTM status    .                   .                          .
  _____________________°                    .                          .
  .     (for LU1)       .                   .                          .
  .                     .                   .                          .
```

Figure 4-4. Example Flow Showing Request for RTM Data and Status for a Single LU, and the Resulting Reply

```
Network
Operator              CPMS                PUMS                     LU1     LU2...  LUn
  .                     .                   .                       .       .       .
  .    request for      .                   .                       .       .       .
  . RTM data and status .                   .                       .       .       .
  °_____________________ÿ                   .                       .       .       .
  .    (for all LUs)    .   NMVT (X'8080') .                        .       .       .
  .                     °_________________ÿ                        .       .       .
  .                     .    +RSP(NMVT)    .                        .       .       .
  .                     ___________________°     request for       .       .       .
```

```
  .                    .                 .  RTM data and status  .       .       .
  .                    .                 º_______________________ÿ       .       .
  .                    .                 .                       .       .       .
  .                    .                 .                       .       .       .
  .                    .                 .  RTM data and status  .       .       .
  .                    .   NMVT (X'0080') _______________________º       .       .
  .                    _________________º                       .       .       .
  .                    .    +RSP(NMVT)   .                       .       .       .
  .                    º_________________ÿ                       .       .       .
  . RTM data and status .                 .                       .       .       .
  ____________________º                  .                       .       .       .
  .    (for LU 1)      .                 .                       .       .       .
  .                    .                 .    request for        .       .       .
  .                    .                 .  RTM data and status  .       .       .
  .                    .                 º_______________________________ÿ       .
  .                    .                 .                       .       .       .
  .                    .                 .    RTM status         .       .       .
  .                    .                 _______________________________º       .
  .         º          .                 .          º            .       .       .
  .         º          .                 .          º            .       .       .
  .         º          .                 .     request for       .       .       .
  .                    .                 .  RTM data and status  .       .       .
  .                    .                 º_____________________________________ÿ
  .                    .                 .                       .       .       .
  .                    .                 .    RTM status¹        .       .       .
  .                    .                 _____________________________________º
  .                    .   NMVT (X'0080') .                      .       .       .
  .                    _________________º                       .       .       .
  .                    .    +RSP(NMVT)   .                       .       .       .
  .                    º_________________ÿ                       .       .       .
  .    RTM  status     .                 .                       .       .       .
  ____________________º                  .                       .       .       .
  .    (for LU n)      .                 .                       .       .       .
```

¹ The architecture allows a PU to reply with status only (no data) for its last defined LU if that LU has no accumulated RTM data. This prevents an implementation from having to buffer the data it has collected from all its LUs in order to determine whether the reply it is sending is the last one for the request.

Figure 4-5. Example Flow Showing Request for RTM Data and Status for All LUs, and the Resulting Replies

```
Network
Operator              CPMS               PUMS                    LU1     LU2...  LUn
  .                    .                 .                       .       .       .
  .    request for     .                 .                       .       .       .
  . RTM data and status .                 .                       .       .       .
  º_____________________ÿ                 .                       .       .       .
  .   (for all LUs)    .                 .                       .       .       .
  .                    .                 .                       .       .       .
  .                    .  NMVT (X'8080') .                       .       .       .
  .                    º_________________ÿ                       .       .       .
  .                    .    +RSP(NMVT)   .                       .       .       .
  .                    _________________º                       .       .       .
  .                    .                 .                       .       .       .
  .                    .                 .     request for       .       .       .
  .                    .                 .  RTM data and status  .       .       .
  .                    .                 º_______________________ÿ       .       .
  .                    .                 .     RTM status        .       .       .
  .                    .                 _______________________º       .       .
  .                    .                 .     request for       .       .       .
  .                    .                 .  RTM data and status  .       .       .
  .                    .                 º_____________________________ÿ       .
  .                    .                 .     RTM status        .       .       .
  .                    .                 _____________________________º       .
  .                    .                 .          º            .       .       .
  .                    .                 .          º            .       .       .
  .                    .                 .     request for       .       .       .
  .                    .                 .  RTM data and status  .       .       .
  .                    .                 º_____________________________________ÿ
  .                    .                 .                       .       .       .
  .                    .                 .     RTM status        .       .       .
  .                    .                 _____________________________________º
  .                    .   NMVT (X'0080') .                      .       .       .
  .                    _________________º                       .       .       .
  .                    .    +RSP(NMVT)   .                       .       .       .
  .                    º_________________ÿ                       .       .       .
  .                    .                 .                       .       .       .
  .    RTM  status     .                 .                       .       .       .
  ____________________º                  .                       .       .       .
  .     (for LU n)     .                 .                       .       .       .
  .                    .                 .                       .       .       .
```

Figure 4-6. Example Flow Showing Request for RTM Data and Status for All

LUs, When No LUs Have Accumulated Data. PUMS returns status only (no data) for its last defined LU.

1.4.1.3.3 Sending Unsolicited RTM Data and Status

RTM data and status can be sent unsolicited to a control point under the following conditions, if specified by the RTM reporting parameters:

- When session deactivation is detected
- When a response-time counter overflow is detected

Upon detection of one of the above conditions, the LU LMS sends RTM data and status to PUMS. PUMS formats the information into an NMVT containing an RTM (X'0080') major vector and sends it to CPMS. In the case of counter overflow, PUMS and the LU LMS may also be configured to send an Alert reporting the condition.

When CPMS receives the NMVT, it optionally logs the data and passes it to the network operator. The flow is of the type described in "Unsolicited Flows" in topic 1.2.2.1. Refer to Figure 4-7 for an example flow.

```
Network
Operator                 CPMS                PUMS                       LU
  .                       .                   .                         .
  .                       .                   .                         .
  .                       .                   .   RTM data and status   .
  .                       .                   __________________________°
  .                       .   NMVT (X'0080')  .                         .
  .                      _________________°                            .
  .                       .   +RSP(NMVT)      .                         .
  .                       °___________________ÿ                         .
  . RTM data and status   .                   .                         .
 ________________________°                   .                         .
  .                       .                   .                         .
  .                       .                   .                         .
  .                       .                   .                         .
```

Figure 4-7. Example Flow Showing Unsolicited Return of RTM Data and Status

---

1.5 Chapter 5. Configuration Management

Configuration management is the control of information necessary to identify network resources. This physical identification includes information such as machine type and serial number (to identify hardware), program number and release level (to identify software), and port number and power-on status of port-attached devices. Query product ID is the only element of configuration management for which SNA management services have been provided.

Subtopics:

- 1.5.1 Query Product ID (QPI)

1.5.1 Query Product ID (QPI)

Query Product ID (QPI) provides the capability to retrieve product-identification information from a specified PU. The information can be used to facilitate problem determination or problem diagnosis. Another use of the information is querying the identification of hardware and software products in a network for inventory purposes. This latter function is sometimes referred to as *network asset management*.

Refer to Figure 5-1 for an overview of the steps required for QPI and the component that they are implemented within.

```
 ..............................................................................
                                                                            "
                                                                            |
```

```
                                                                  Physical resources
                                                                    Manager  LMS
        ____________________                                            |
       |                    |                                           |
       |    collection      |
       |     of product_    |.........................................................
       | identification data |_______________________                 "
       |_________ __________|                       |                   |
                 |                                 |                   |
                 |                                 |                  PUMS
                                                  |                   |
     ________________________        _____________|______              |
     |   reporting of product_ |       | forwarding requests  |
...  |   identification data   |...... | for product-id data  |  ........................
     |_____________ ___________|       |______________________|                "
                 |                                   "                  |
                                                     |                  |
     ________________________                        |                  |
     | recording and retrieval |_________________    |                  |
     |       of product_       |                 |   |                  |
     |  identification data    |                 |   |                  |
     |_____ ___________________|                 |   |                 CPMS
          |                                      |   |                  |
          |                                      |   |                  |
          |                                      |   |                  |
          |                                      |   |                  |
          |                                      |   |                  |
                                               |   |                  |
 ________________________                  ____________|_______|___________      |
| presentation of product_ |           |  processing and forwarding of   |      |
|  identification data     |           |  network operator requests to   |      |
| to requesting network    |           | retrieve product_identification |      |
|       operator           |           |  data for a specified PU        |
|__________________________|..........|_________________________________|...................
```

Figure 5-1. Overview and Placement of Query Product Identification (QPI) Steps

Subtopics:

- 1.5.1.1 Functions Provided
- 1.5.1.2 Implementation
- 1.5.1.3 Format Usage

1.5.1.1 Functions Provided

QPI is composed of the following steps.

Subtopics:

- 1.5.1.1.1 Collection of Product-Identification Data
- 1.5.1.1.2 Forwarding Requests for Product-Identification data
- 1.5.1.1.3 Reporting of Product-Identification Data
- 1.5.1.1.4 Recording and Retrieval of Product-Identification Data
- 1.5.1.1.5 Presentation of Product-Identification Data
- 1.5.1.1.6 Processing and Forwarding of Network Operator Requests

---

1.5.1.1.1 Collection of Product-Identification Data

The collection step of QPI provides both hardware and software (if present) identification for the specified PU and port-attached devices when requested.

° The following data is provided for each hardware product:

- Machine type
- Model number (1)
- Serial number or repair ID number, consisting of a plant of manufacture identifier and a sequence number
- Optionally, the engineering change (EC) level of the microcode in the product
- Optionally, hardware product common name

- Emulated product identifier, if the product emulates another hardware product

° The following data is provided for each software product:

- Software product serviceable component identifier. This data is required for IBM software products assigned a component ID by the IBM National Service Division (NSD).

- Software product program number. This field is required for IBM products not assigned a component ID by the IBM NSD.

- Software product common level. This field is required for IBM products not assigned a component ID by the IBM NSD.

- Software product common name

- Either the software product customization identifier or the software product customization date and time, if the product is customer modifiable

° The following data is optionally provided for each hardware product and software product:

- The name of the vendor, if the product is not manufactured by IBM

The collection step of QPI provides the following configuration description for each port-attached device:

° The port number the device is attached to

° Whether the device is powered on or off

° Whether the device was powered on since the last QPI solicitation

The following product set description data is optionally provided for the PU and each port-attached device:

° Physical location of the product set (e.g., city, room, and building)

° LAN universally-assigned address (2)

° One or more user-defined product set attributes. Each attribute consists of a label and data (e.g., "LESSOR" and "XYZ CORPORATION").

---

(1) The model number must be provided if it is required, in conjunction with machine type and serial number to uniquely identify a product instance. If the model number is not needed to uniquely identify a product instance, its inclusion, for the purpose of providing additional information, is optional.

(2) I.e., a LAN address that is assigned, administered, and guaranteed to be unique across all local area networks by the IEEE.

1.5.1.1.2 Forwarding Requests for Product-Identification data

This function requests product-identification data from the physical resources manager LMS for the SNA node and port-attached devices.

1.5.1.1.3 Reporting of Product-Identification Data

The reporting step of QPI provides for the reporting of product-identification information from the specified PU to the requesting control point.

1.5.1.1.4 Recording and Retrieval of Product-Identification Data

The recording and retrieval step of QPI is implemented at the requesting control point and provides for data base management (e.g., storage,

retrieval) of collected data.

1.5.1.1.5 Presentation of Product-Identification Data

The presentation step of QPI is implemented at the requesting control point and presents product-identification information to a requesting network operator.

1.5.1.1.6 Processing and Forwarding of Network Operator Requests

The processing and forwarding step of QPI is implemented at the requesting control point and provides a mechanism for accepting network operator requests for product-identification information, and forwarding those requests to the specified PU.

1.5.1.2 Implementation

Support of QPI is provided in a PU by "EP_QPI Function Set" in topic 2.10.3. This is an optional function set implemented by type 2.0 or boundary-function-attached type 2.1 nodes.

1.5.1.3 Format Usage

NETWORK MANAGEMENT VECTOR TRANSPORT (NMVT)
Request Product Set ID (X'8090') major vector
Flow: From SSCP to PU (Normal)

NETWORK MANAGEMENT VECTOR TRANSPORT (NMVT)
Reply Product Set ID (X'0090') major vector
Flow: From PU to SSCP (Normal)

A network operator can request product identification information for a specified PU. Data necessary for the request is passed to CPMS.

CPMS formats the request into an NMVT containing a Request Product Set ID (X'8090') major vector and sends it to PUMS. When PUMS receives this NMVT, it forwards the request to the physical resources manager LMS. The LMS passes the requested data back to PUMS. PUMS formats the data into one or more NMVTs containing a Reply Product Set ID (X'0090') major vector and sends it to its controlling CPMS.

When CPMS receives the NMVT, it optionally logs the data and passes it to the network operator. The flow is of the type described in "Request/Reply Flows for Non-Bulk Data" in topic 1.2.2.3. Refer to Figure 5-2 for an example flow.

```
                                                                  Physical
        Network                                                   Resources Mgr
        Operator                 CPMS              PUMS              LMS
           .                       .                 .                 .
           .  request for product  .                 .                 .
           .  identification data  .                 .                 .
           º_______________________ÿ                 .                 .
           .                       .  NMVT (X'8090')  .                 .
           .                       º__________________ÿ                 .
           .                       .                 .                 .
           .                       .    +RSP(NMVT)   .                 .
           .                       _________________º    request for product    .
           .                       .                 .     identification data    .
           .                       .                 º_____________________________ÿ
           .                       .                 .           product          .
           .                       .                 .     identification data    .
           .                       .                 .     for PU                 .
           .                       .                 _____________________________º
           .                       .  NMVT (X'0090')  .                 .
           .                       _________________º                   .
           .                       .    +RSP(NMVT)   .                 .
           .      product          º__________________ÿ                 .
           .  identification data  .                 .                 .
           _______________________º                 .                 .
           .                       .                 .    first port-attached     .
           .                       .                 .    device's product        .
           .                       .                 .    identification data     .
           .                       .                 _____________________________º
```

```
.                       .  NMVT (X'0090')  .                          .
.                       _______________   °             °            .
.                       .    +RSP(NMVT)    .              °            .
.      product          °_________________ÿ              °            .
.  identification data  .        °         .                           .
_______________________°        °        .                           .
.                       .        °         .    last port-attached      .
.                       .                  .    device's product        .
.              °        .                  .    identification data     .
.              °        .                  _____________________________°
.              °        .  NMVT (X'0090')  .                           .
.                       _______________   °                           .
.                       .    +RSP(NMVT)    .                           .
.      product          °_________________ÿ                           .
.  identification data  .                  .                           .
_______________________°                  .                           .
.                       .                  .                           .
.                       .                  .                           .
```

Figure 5-2. Example Flow Showing QPI Request to a PU and the Subsequent Replies. The flows beginning with the one labeled "first device's product identification data" occur only if the network operator queried for the identification of port-attached products as well as the products making up the SNA node.

1.6 Chapter 6. Change Management

Change management is the process of planning and controlling changes in a network. A *change* is defined as an alteration (addition, deletion, or modification) of one or more information system components, of one of the following types: hardware (may include microcode), or software (either system or application, vendor-supplied or user-written). Current architecture provides for management of changes that can be distributed electronically (not changes to hardware circuitry). For an overview of the elements of change management and their relationships to one another, refer to Figure 6-1.

```
 _______________________
|                       |
|    Change Planning    |________
|                       |______  |
|_______________________|      |  |
            |                  |  |
                              |  |
°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°      |  |
°                            °      |  |
°     Change Control         °______|  |
°                            °         |
°°°°°°°°°°°°°°°°°°°°°°°°°°°°°         |
            |                         |
            |                         |
                                     |
°°°°°°°°°°°°°°°°°°°°°°°°°°°°°         |
°                            °         |
°     Node Activation        °_________|
°                            °
°°°°°°°°°°°°°°°°°°°°°°°°°°°°°
```

Note: SNA management services are provided to assist with elements bounded by °°°°°.

Figure 6-1. Overview of Flow Between Change Management Elements

Change management, as described here and in subsequent chapters, defines the protocols followed and the formats that flow between nodes implementing this architecture. These descriptions are not intended to address a common user interface or programming interface.

Subtopics:

- 1.6.1 Change Planning
- 1.6.2 Change Control
- 1.6.3 Node Activation
- 1.6.4 Example Change Management Plans

1.6.1 Change Planning

Change planning is the element of change management that encompasses all the activities required to take place before changes can be distributed and installed. Refer to Figure 6-2 for an overview of the steps required for change planning.

```
                    _____________________
                   | change preparation  |
                   |_____________________|
                            |
                            |
                            |

                    _____________________
                   |  change approval    |
                   |_____________________|
                            |
                            |
                            |

                    _____________________
                   |  change scheduling  |
                   |_____________________|
```

Figure 6-2. Overview of Change Planning Steps

Subtopics:

- 1.6.1.1 Overview
- 1.6.1.2 Change Preparation
- 1.6.1.3 Change Approval
- 1.6.1.4 Change Scheduling

1.6.1.1 Overview

A *network planner* is a person or program responsible for planning the configuration (and changes) of all or part of a network.

The need for changes to the network may arise for a variety of reasons. A non-exhaustive list of examples would include:

° Problem diagnosis identifies a required fix

° A vendor distributes preventive maintenance

° A vendor distributes general software

° Users require configuration changes

When a network planner is aware of the need for a change to be made to the network, change planning begins. Change planning is composed of the following steps.

1.6.1.2 Change Preparation

For a general introduction to the SNA/MS *entry point* and *focal point* roles, refer to "Management Services Roles" in topic 1.1.2.2.

Entry points are typically remote and unattended. An entry point may be a large system or a small one, an intelligent workstation, a fixed function device, or a control unit. One of the entry points may serve as the preparation site for *change files*, that is, files containing component replacements or updates and any necessary instructions to install them. Change files can contain one of the following types of data: microcode, microcode customizing data, software, software customizing data, applications data, procedures, or documentation. The change file preparation entry point, if required, is typically located at the central site with the focal point, where it can be attended. The preparation site can also be at a focal point rather than at an entry point, or even in a system that serves neither a focal point nor an entry point role (in that case, the prepared change files must be introduced by some other means, for example, distribution tapes, at either a focal point or an entry point). It is the responsibility of the preparer of the change file to include in it any necessary prerequisite information, to be checked by the target entry points when the change is installed. However, corequisites (a group of change files to be installed together) may be specified by the network planner.

1.6.1.3 Change Approval

Once a change file has been prepared for distribution and installation, approvals must be obtained from management. This may include various parts of the organization, such as network users and network management

staff.

1.6.1.4 Change Scheduling

When change files have been approved for distribution and installation, the network planner can create a *change management plan*. The functions provided to the network planner that can be scheduled are described in more detail in "Change Control" in topic 1.6.2.

1.6.2 Change Control

Change control is the element of change management that distributes change files to entry points, and installs them there. Refer to Figure 6-3 for an overview of the steps required for change control.

```
 _____________________
| change distribution |
|_____________________|
          |
          |
          |

 _____________________
| change installation |
|_____________________|
          |
          |
          |

 _____________________
|   change tracking   |
|_____________________|
```

Figure 6-3. Overview of Change Control Steps

Subtopics:

- 1.6.2.1 Overview
- 1.6.2.2 Change Distribution
- 1.6.2.3 Change Installation
- 1.6.2.4 Change Tracking
- 1.6.2.5 Functions Provided
- 1.6.2.6 Implementation
- 1.6.2.7 Format Usage

1.6.2.1 Overview

Change control is composed of the following steps.

1.6.2.2 Change Distribution

SNA/File Services (SNA/FS) architecture is used by management services for bulk data transport. SNA/FS moves files (that can be bulk data) from one location to another, provides a cross-enterprise-unique naming scheme (the SNA/FS *global name*), and defines the fetching and storing services. It uses enhanced SNA/Distribution Services formats, *format set 2*. For a detailed description of this architecture, refer to *SNA/File Services Reference* and *SNA/Distribution Services Reference*.

SNA/MS defines a naming convention for change files. The name used is the SNA/FS global name, and SNA/MS specifies what values the tokens in that name can take.

SNA/File Services capability is symmetric with respect to focal point and entry point roles. That is, nodes in either role can both send files to, and retrieve files from, other nodes. Change management commands, however, are issued only from a focal point. A summary of focal point and entry point roles with respect to change distribution and change management functions is provided in Table 6-1 in topic 1.6.2.5.11.

SNA/DS is the transport mechanism used for the delivery of all commands and replies flowing between the focal point and entry point. Nodes between the focal point and the entry point may perform the SNA/DS intermediate role to provide a connectionless delivery service and fan-out (that is, one copy coming into the intermediate node can be replicated and forwarded to several subsequent nodes for ultimate routing to the

destinations).

1.6.2.3 Change Installation

Once change files have been distributed to an entry point, they can be installed there. In fact, distribution and installation can be done with a single request if desired. See "Functions Provided" in topic 1.6.2.5 for a description of the installation functions.

1.6.2.4 Change Tracking

SNA/MS, SNA/FS, and SNA/DS reports are all tracked at the focal point. This allows a network planner to view the distribution and installation status and history for each entry point under control of the focal point.

1.6.2.5 Functions Provided

Provision for the following change control requests is made by an SNA/MS change management focal point at its interface with the network planner, or entry point at its interface with the local operator:

Subtopics:

- 1.6.2.5.1 Retrieve
- 1.6.2.5.2 Send
- 1.6.2.5.3 Delete
- 1.6.2.5.4 Install
- 1.6.2.5.5 Initiate
- 1.6.2.5.6 Send-and-Install
- 1.6.2.5.7 Send-and-Initiate
- 1.6.2.5.8 Remove
- 1.6.2.5.9 Accept
- 1.6.2.5.10 Cancel
- 1.6.2.5.11 Role Summary
- 1.6.2.5.12 Testing Features

1.6.2.5.1 Retrieve

Obtains a change file prepared at an entry point or at another focal point, for storage at the focal point. If issued from an entry point, it obtains a change file from a focal point or another entry point, for storage at the entry point.

1.6.2.5.2 Send

Distributes a change file from the focal point to one or more entry points, or other focal points. If issued from an entry point, it distributes a change file from the entry point to one or more focal points, or other entry points.

1.6.2.5.3 Delete

Deletes a change file at one or more entry points or focal points.

1.6.2.5.4 Install

Uses a change file and its corequisites, if any, to alter at one or more entry points, or at the local node if locally issued, all components necessary to effect the change. The entry point can perform such alteration in a removable manner if requested, that is, so that a subsequent request (*Remove*) can return all those components to their condition prior to the alteration. The network planner can request testing, before and/or after the installation process is performed by the entry point. For example, an entry point can test a new version of a configuration file for validity before deleting the old version. Also, automatic removal of changes (if the tests or installation fail) or automatic acceptance (see *Accept* below) is possible.

The network planner can designate components altered by the installation process for *trial* activation, or alternately, *production* activation. This conditions how the entry point is later reactivated.

The network planner may prohibit the entry point from altering active components as part of the installation process. When doing so, a subsequent reactivation is required to effect the installed changes.

If the entry point supports them, two additional capabilities exist:

- The entry point can perform the installation by date and time (either local time or Greenwich Mean Time).

- The entry point can automatically accept the change after a specified delay (expressed in days).

1.6.2.5.5 Initiate

Causes a command procedure to be initiated at one or more entry points. It may be issued by a network planner at a focal point or by a local operator at the local node.

1.6.2.5.6 Send-and-Install

The same as Install, except that a change file is sent in the same request. It may be issued only by the network planner at a focal point.

1.6.2.5.7 Send-and-Initiate

The same as Initiate except that a command procedure is sent in the same request. It may be issued only by the network planner at a focal point who also has the ability to prevent or allow destruction of the command procedure following its completion.

1.6.2.5.8 Remove

Returns all components previously altered in connection with a change to their condition prior to the installation of the change. This is possible only for changes previously installed in a removable manner. It may be issued by a network planner at a focal point or by a local operator at the local site.

The network planner can request testing after the removal process is performed at the entry point and can also require that a subsequent reactivation of the entry point be performed before the prior level of the altered components is applied to the running code.

If the entry point supports it, the capability exists to perform the removal by date and time (either local time or Greenwich Mean Time).

1.6.2.5.9 Accept

Relinquishes resources at an entry point required to maintain removability of a change. This cancels the removability of a change previously installed in a removable manner. It may be issued by a network planner at a focal point or by a local operator at the local node.

If the entry point supports it, the capability exists to perform the acceptance by date and time (either local time or Greenwich Mean Time).

Focal point implementations may provide the network planner the ability to aggregate a series of requests, and specific scheduling and conditioning rules for their execution, into a *change management plan*.

1.6.2.5.10 Cancel

Issued by the network planner to request the focal point to send a command to one or more entry points to cause cancelation of another command at the entry point. It may also be issued by a local operator to cause cancelation of a command at the local node. The cancel request is also used in the operations management category of management services. See "Operations Management Function Set" in topic 1.7.3 and "EP_OPERATIONS_MGMT Function Set" in topic 2.10.6 for more information on its use in that category.

1.6.2.5.11 Role Summary

The following is a summary of focal point and entry point roles with respect to the change distribution and change management functions described above.

Table 6-1. Functions and the Roles to Which They Apply

Function	FP Source EP Target	EP Source FP Target	FP Source FP Target	EP Source EP Target
RETRIEVE	Yes	Yes	Yes	Yes
SEND	Yes	Yes	Yes	Yes
DELETE	Yes	Yes	Yes	Yes
INSTALL	Yes	No	No	No
INITIATE	Yes	No	No	No
SEND_AND_INSTALL	Yes	No	No	No
SEND_AND_INITIATE	Yes	No	No	No
REMOVE	Yes	No	No	No
ACCEPT	Yes	No	No	No
CANCEL	Yes	No	No	No

1.6.2.5.12 Testing Features

Without the explicit testing features provided on the Install request, the only testing would be done by entry point user(s) over some period of time after the change was installed. If problems were encountered, the central-site network planner would need to be consulted, because only he would know about the previous installation of changes and be in a position to issue the Remove commands.

Two kinds of explicitly requested tests are required *as part of the installation process*, so that the central-site network planner can be informed immediately, as part of the installation report, about the results of certain diagnostic tests:

- *Pre-tests* -- Tests made before the programming components are altered

- *Post-tests* -- Tests made after the programming components are altered but before the installation report is made

Some reasons that automatic testing of changes is desirable:

- Possible corruption of the change between its initial development and installation at the entry point

- Possible sensitivity of the change to differences between the maintenance level of the target system and the maintenance level of those systems where the change has already been tested

- Possible sensitivity of the change to differences between the configuration of the target system and the configuration of those systems where the change has already been tested

- Possible sensitivity of the change to differences between functions or applications present at the target system and those present at systems where the change has already been tested

The *pre-test* is performed (if requested) by the entry point by examining

the change file *before* components are altered. The change file contents may be examined for self-consistency, and consistency with the entry point's configuration, maintenance level, or application set. For example, an altered version of a file that contains input data to a routine can be checked to see if it contains inconsistent specifications.

If the pre-test fails, then no attempt is made to alter the components and the installation report is made to the requester with the test results.

The *post-test* is performed (if requested) by the entry point *after* components are altered, but before the installation report is made. Altered versions of the components are tested directly -- for example, by executing diagnostic routines. Such routines or test instructions can be distributed along with the change.

If the post-test fails, then the components are returned to their unaltered condition if another parameter, *automatic removal*, is specified. In any event, the installation report is sent with the test results, and the requester is informed immediately. In this way, the requester has the opportunity to remove the change so that the impact on end users is minimized.

Implementations may choose to include testing procedures in the change files, so that the general procedures do not have to be developed and included in advance at the target entry points.

1.6.2.6 Implementation

Support of controlling changes is provided in an entry point by "EP_CHANGE_MGMT Function Set" in topic 2.10.4. This is an optional function set.

A series of example change mangement flows is provided in "Example Flows" in topic 2.10.4.13. These examples show, in detail, the data that flows between the focal point and entry point as well as the interaction between SNA/MS, SNA/DS and SNA/FS at each node.

1.6.2.7 Format Usage

```
SNA/FS AGENT OBJECT
   FS_Agent_Request GDS Variable (X'1530')
   FS_Agent_Report GDS Variable (X'154A')
   Flow:  From CP to PU (over LU-LU sessions used by SNA/DS)
          From CP to CP (over LU-LU sessions used by SNA/DS)
```

This format is used for the Retrieve, Send, and Delete functions that exploit the FS-capability of the SNA/MS agent. A CP-MSU does not flow in the agent object.

Figure 6-4 shows an example flow for the retrieve function. The flow is of the type described in "Request/Reply Flows for Bulk Data" in topic 1.2.2.4. The flow consists of the following stages:

1. A network planner requests the focal point SNA/MS to retrieve a change file from an entry point. The request is passed to SNA/MS in the focal point.

2. The focal point SNA/MS formats an SNA/FS request for the entry point to fetch and return a change file. The focal point SNA/MS sends the command to the entry point SNA/MS over the LU-LU sessions used by SNA/DS.

3. SNA/MS passes the request to the SNA/FS server.

4. The SNA/FS server fetches the requested change file and sends it to the focal point SNA/MS over an LU-LU session used by SNA/DS.

5. The SNA/FS server at the focal point stores the change file and SNA/DS notifies SNA/MS that it has arrived.

6. SNA/MS notifies the network planner that the change file has arrived.

```
                    ________ Focal Point _________        _________ Entry Point _________
```

```
   Network          SNA/MS        SNA/FS      SNA/DS      SNA/DS        SNA/FS       SNA/MS
   Planner                        server                                server
      .               .             .           .           .             .             .
      .               .             .           .           .             .             .
      .  Retrieve     .             .           .           .             .             .
      .  request      .             .           .           .             .             .
(1)   °_______________ÿ             .           .           .             .             .
      .               .  SNA/FS     .           .           .             .             .
      .               .  request    .           .           .             .             .
(2)   .               °_________________________ÿo__________ÿo___________________________ÿ
      .               .             .           .           .             .  SNA/FS     .
      .               .             .           .           .             .  request    .
(3)   .               .             .           .           .             _____________°
      .               .             .            change file                .             .
(4)   .               .             ___________o__________o___________°            .
      .               .  signal     .           .           .             .             .
(5)   .               _____________°           .           .            .            .
      .  reply        .             .           .           .             .             .
(6)   _______________°             .           .           .            .            .
      .               .             .           .           .             .             .
      .               .             .           .           .             .             .
```

Figure 6-4. The Request/Reply Flow for Retrieving Change Files

CONTROL POINT MANAGEMENT SERVICES UNIT (CP-MSU)
Request Change Control MV (X'8050')
Change Control MV (X'0050')
Flow: From CP to PU (over LU-LU sessions used by SNA/DS)
From CP to CP (over LU-LU sessions used by SNA/DS)

This format is used for the Install, Remove, and Accept functions. A CP-MSU containing the appropriate SNA/MS major vector flows in the agent object.

Figure 6-5 shows a request/reply flow to send a change to an entry point and install it there. The flow is of the type described in "Request/Reply Flows for Bulk Data" in topic 1.2.2.4. The flow consists of the following stages:

1. A network planner requests that a change be sent to an entry point. An Install function is also requested. The planner's request is passed to SNA/MS in the focal point.

2. The focal point SNA/MS formats a request CP-MSU containing a Request Change Control (X'8050') major vector. It invokes its SNA/FS server to fetch the change file from storage.

3. The SNA/FS server fetches the file, and it flows together with the CP-MSU on a SNA/DS conversation over an LU-LU session between the focal point and the entry point.

4. The SNA/FS server in the entry point stores the change file and SNA/DS notifies SNA/MS that it has arrived, passing the CP-MSU and SNA/FS parameters in the agent object.

5. SNA/MS installs the change file, and builds a reply CP-MSU containing a Change Control (X'0050') major vector. Then it sends the report to the focal point SNA/MS on a SNA/DS conversation over an LU-LU session.

6. After the report has reached the focal point SNA/MS, it is passed to the network planner that made the request.

```
                     _________ Focal Point _________        __________ Entry Point ________

   Network          SNA/MS        SNA/FS      SNA/DS      SNA/DS        SNA/FS       SNA/MS
   Planner                        server                                server
      .               .             .           .           .             .             .
      .               .             .           .           .             .             .
       Send-and-Install             .           .           .             .             .
      .  request      .             .           .           .             .             .
(1)   °_______________ÿ             .           .           .             .             .
      .               .             .           .           .             .             .
      .               CP-MSU (X'8050')          .           .             .             .
      .             and SNA/FS parameters       .           .             .             .
(2)   .               °_____________ÿ           .           .             .             .
      .               .             .           .           .             .             .
      .               .             .        CP-MSU (X'8050')             .             .
      .               .             .        and a change file            .             .
(3)   .               .             °__________ÿo__________ÿo____________ÿ             .
      .               .             .           .           .             .             .
      .               .             .           .           .             CP-MSU (X'8050')
      .               .             .           .           .           and SNA/FS parameters
(4)   .               .             .           .           .             °_____________ÿ
      .               .             .           .  report   .             .             .
(5)   .               ______________________________o__________o___________________________°
      .  report       .             .           .           .             .             .
```

```
(6)   _______________°            .          .          .          .          .
      .              .            .          .          .          .          .
      .              .            .          .          .          .          .
```

Figure 6-5. The Request/Reply Flow for Send-and-Install

Figure 6-6 shows a request/reply flow to remove a change already installed removably at an entry point. The flow is of the type described in "Request/Reply Flows for Bulk Data" in topic 1.2.2.4. The flow consists of the following stages:

1. A network planner requests that a change be removed at an entry point. The planner's request is passed to SNA/MS in the focal point.

2. The focal point SNA/MS formats a request CP-MSU containing a Request Change Control (X'8050') major vector. The SNA/FS server is invoked to build the SNA/FS control information to identify the change file to be removed.

3. The CP-MSU and control information is sent on an SNA/DS conversation over an LU-LU session between the focal point and the entry point.

4. The SNA/FS server in the entry point decodes the control information, and SNA/DS notifies SNA/MS that the request has arrived, passing the CP-MSU and SNA/FS parameters in the agent object.

5. SNA/MS removes the change file, and builds a reply CP-MSU containing a Change Control (X'0050') major vector. Then it sends the report to the focal point SNA/MS on a SNA/DS conversation over an LU-LU session.

6. After the report has reached the focal point SNA/MS, it is passed to the network planner that made the request.

```
                    __________ Focal Point ________        _________ Entry Point _________

   Network          SNA/MS        SNA/FS      SNA/DS      SNA/DS        SNA/FS       SNA/MS
   Planner                        server                                server
      .               .            .           .           .            .            .
      .               .            .           .           .            .            .
      .  Remove       .            .           .           .            .            .
      .  request      .            .           .           .            .            .
(1)   °_______________ÿ            .           .           .            .            .
      .               .            .           .           .            .            .
      .               CP-MSU (X'8050')         .           .            .            .
      .             and SNA/FS parameters      .           .            .            .
(2)   .               °_____________ÿ          .           .            .            .
      .               .            .           .           .            .            .
      .               .            .       CP-MSU (X'8050') and         .            .
      .               .            .     SNA/FS control inormation      .            .
(3)   .               .            °__________ÿo__________ÿo____________ÿ            .
      .               .            .           .           .            .            .
      .               .            .           .           .            CP-MSU (X'8050')
      .               .            .           .           .          and SNA/FS parameters
(4)   .               .            .           .           .            °_____________ÿ
      .               .            .           .  report   .            .            .
(5)   .               ___________________________o___________o___________________________°
      .  report       .            .           .           .            .            .
(6)   _______________°            .           .           .            .            .
      .               .            .           .           .            .            .
      .               .            .           .           .            .            .
```

Figure 6-6. The Request/Reply Flow for Remove

Other key examples (similar to that just described for Remove, and so not separately shown) include:

- An Install request acting on a change file that was sent previously

- An Accept request

1.6.3 Node Activation

Subtopics:

- 1.6.3.1 Functions Provided
- 1.6.3.2 Implementation
- 1.6.3.3 Format Usage

1.6.3.1 Functions Provided

MS change management provides for the following request at its focal point protocol boundary with the network planner or at its entry point protocol boundary with the local operator:

Subtopics:

- 1.6.3.1.1 Activate

1.6.3.1.1 Activate

Causes reactivation of an entry point. This request may be issued by a network planner at a focal point or by a local operator at the local node. Such reactivation uses changes installed on a trial basis in addition to those installed in production. In addition, the network planner can request that the entry point either not attempt or force reactivation if it is not in a quiesced state.

If the entry point supports them, two additional capabilities exist:

° The entry point can perform the activation by date and time (either local time or Greenwich Mean Time).

° The entry point can wait a specified period of time to quiesce before either rejecting or forcing the activation request.

The Activate command provides a network planner the ability to reactivate an entry point in a change management plan. For example, an initial microprogram load (IML) of the node may be performed. A reply is returned by the entry point prior to reactivation, to indicate acceptance of the Activate command.

If the activation acceptance reply indicates so, an activation report can be returned by the entry point to report the results of the activation attempt. No reply is returned by a base entry point after reactivation, due to lack of non-volatile storage capability in small entry-point implementations.

Unlike other change management functions, Activate does not refer to bulk data and so the SNA/File Services server is not invoked by SNA/Distribution Services in the process of distributing requests and reports.

The Activate command is also used in the operations management category of management services. See "Operations Management Function Set" in topic 1.7.3 and "EP_OPERATIONS_MGMT Function Set" in topic 2.10.6 for more information on its use in that category.

The Change Management Activation Use Parameter: The Activate request contains a parameter that causes the entry point to activate both the trial and the production versions of altered entry point components. This parameter of the Install request causes the entry point to install the indicated changes for *trial activation* or *production activation*. Entry points implement both of the following types of local reactivation:

° Use of both trial and production components, and

° Use of production components only.

Changes that cannot be tested fully or that have a strong potential to affect the entry point to focal point communication path are best installed on trial. After activation and a period of successful operation, the changes may be installed again in production.

Of course, to provide the trial activation capability, an entry point must be capable of installing a change removably. That is, the entry point must be able to keep a copy of the production-level system. Storage capabilities at the entry points may preclude support of trial activation in some cases. If so, the installation will be refused if activation use of *trial* is specified.

The parameters on the *Install* and *Activate* requests reflect very specific reliability requirements for entry point implementations. While the basics of the Activate request provide the ability to reactivate an entry point after changes have been installed (for example, by loading altered microcode), there is a need to provide a way to use only unaltered versions of components during local activation. Without such capability,

reactivation of the entry point could result in the use of a change so destructive that the path to the focal point cannot be maintained. Repair (in the form of further change distribution and installation) cannot be triggered by the focal point in this case. Reactivation must be triggered at the entry point through human intervention. Reactivation of the (working) production level of components can restore communication with the focal point and allow the network planner to repair the components.

Hence, the ability to store both a production and a trial set of entry point components is necessary to allow operator intervention to be simple, and to avoid the requirement for both change management skills and awareness of network planning activities at the entry point. Through support of a default local activation of the production system, an entry point implementation can provide hardware externals that are very simple for non-technical users of entry points.

1.6.3.2 Implementation

Support of activating nodes is provided in an entry point by "EP_CHANGE_MGMT Function Set" in topic 2.10.4. This is an optional function set.

1.6.3.3 Format Usage

```
CONTROL POINT MANAGEMENT SERVICES UNIT (CP-MSU)
   Request Activation MV (X'8066')
   Activation Acceptance MV (X'0066')
   Activation MV (X'0067')
   Flow:  From CP to PU (over LU-LU sessions used by SNA/DS)
          From CP to CP (over LU-LU sessions used by SNA/DS)
```

This format is used for the Activate function. A CP-MSU containing the appropriate SNA/MS major vector flows in the agent object.

Figure 6-7 shows an example flow for the activate function. The flow is of the type described in "Request/Reply Flows for Bulk Data" in topic 1.2.2.4. The flow consists of the following stages:

1. A network planner requests the focal point SNA/MS to activate an entry point.

2. The focal point SNA/MS formats a request CP-MSU. The focal point then invokes SNA/DS to send the CP-MSU on a conversation over an LU-LU session between the focal point and the entry point.

3. SNA/MS builds a CP-MSU indicating that activation of the entry point will be performed and sends the CP-MSU to the focal point.

4. SNA/MS indicates activation acceptance to the network planner. SNA/MS causes the entry point (including SNA/MS itself) to be reactivated, and the LU-LU session with the focal point is terminated.

5. SNA/MS builds a CP-MSU indicating that activation of the entry point was performed, establishes another conversation with the focal point, and sends the CP-MSU to it.

6. SNA/MS indicates activation to the network planner.

```
                    __________ Focal Point ________        _________ Entry Point _________

   Network          SNA/MS        SNA/FS      SNA/DS      SNA/DS        SNA/FS       SNA/MS
   Planner                        server                                server
      .               .            .           .           .            .             .
      .               .            .           .           .            .             .
      .  Activate     .            .           .           .            .             .
      .  request      .            .           .           .            .             .
(1)   °_________________ÿ          .           .           .            .             .
      .               .            .      CP-MSU (X8066')              .             .
(2)   .               °_________________________ÿo__________ÿo___________________________ÿ
      .               .            .      CP-MSU (X'0066')             .             .
(3)   .               __________________________o__________o___________________________°
      .  report       .            .           .           .            .             .
(4)   _________________°            .           .           .            .             .
                                                   .        .....
                                                 (entry point is reactivated)
                                                   .        .....
      .               .            .      CP-MSU (X'0067')             .             .
(5)   .               __________________________o__________o___________________________°
```

```
        .  report      .            .          .          .          .          .
(6)     _______________°            .          .          .          .          .
        .              .            .          .          .          .          .
```

Figure 6-7. The Request/Reply Flow for Activate

1.6.4 Example Change Management Plans

Subtopics:

- 1.6.4.1 Centralized Customization of Microcode
- 1.6.4.2 Centralized Distribution and Control of Application Software

---

1.6.4.1 Centralized Customization of Microcode

Refer to Figure 6-8.

A network planner at focal point 1 wants to retrieve a microcode customizing data file that was developed at a central site control unit (entry point 2), then send, test, and install it at a remote control unit (entry point 3), and (one week later) accept it.

Network Planner

"Retrieve 'x' from node 2 ¹
Send and install 'x' removably at node 3; test before installation;
install on trial" ¹
Activate trial and production versions (immediately) ¹

(At 00:00 on 12/1/89:)
Accept 'x' at node 3"

```
   |
   |
   |
   |
  _|_____________                                 _____________
 | |             |                               |             |
 |             |                               |             |
 |            __|_______________________________|ÿ    entry    |
 |              |                               |     point 3  |
 |              |                               |_____________|
 |              |                               Remote Control Unit
 |              |
 |              |
 |              |
 |              |
 |              |
 |       focal  |
 |   "   point 1 |
 |   |          |
 |___|__________|
     |
     |
     |
    _|_________
   | |         |
   |          |  Central Site Control Unit
   |   entry  |
   |   point 2 |
   |__________|
```

¹ If this is not successful, the next function will not be performed
and the planner is notified that the plan ended

Figure 6-8. Centralized Customization of Microcode

1.6.4.2 Centralized Distribution and Control of Application Software

Refer to Figure 6-9.

In this example, a user at development node (entry point 1) prepares an application change (in this case, a new tax-witholding table) and sends it to a focal point (2) in an unsolicited manner. The network planner writes a plan to send and install it at three production nodes (entry points 3, 4, and 5) in the manner indicated. The installation is timed for midnight on New Years' Eve, at the local time of each node, since that is when the tax witholding table needs to be changed.

Network Planner

```
"Send and install 'x' at nodes 3, 4, and 5 in production removably
   at 00:00 on 1/1/90 local time with post_test, automatically remove
   if the install or post_test fails, and automatically accept in 30 days"
```

Figure 6-9. Centralized Distribution and Control of Application Software

1.7 Chapter 7. Operations Management

Subtopics:

- 1.7.1 Operations Management
- 1.7.2 Common Operations Services Function Set
- 1.7.3 Operations Management Function Set

1.7.1 Operations Management

This category provides an architecture to both control and query network resources. The function sets within this category are based on the network management philosophy of skill centralization. With this philosophy, skilled operators are located at a central site and can operate and query the distributed resources under their control.

The operations management category consists of two function sets: EP_COMMON_OPERATIONS_SERVICES and EP_OPERATIONS_MGMT. The functions of this category are manifested in either architecturally defined structures or "envelopes" that enclose implementation-specific commands and data.

The following discussion compares the two function sets.

Subtopics:

- 1.7.1.1 Command Enveloping
- 1.7.1.2 Sending Messages to Operators
- 1.7.1.3 Routing to Second-level Application Programs
- 1.7.1.4 MSU Differences

1.7.1.1 Command Enveloping

Both function sets provide the ability to send either architecturally defined commands or "envelopes" that carry resource-specific,

implementation-defined commands. The envelope may contain completely unstructured data. Alternatively, the envelope may provide some indications as to the context of the data contained within it. Multiple envelope structures have been defined and cover a broad range of possible scenarios.

1.7.1.2 Sending Messages to Operators

Remote application programs may need to send a message to a centralized operator. Both function sets provide this capability, each using either architecturally defined commands or command envelopes.

1.7.1.3 Routing to Second-level Application Programs

Common operations services and operations management offer the capability to route commands to application programs served by either the focal-point or entry-point function sets themselves. Such application programs are referred to as either *second-level* or *served* application programs, for they are served by, and exist hierarchically above, the focal-point or entry-point function sets.

A command may originate from an application program served by the common operations services or operations management focal point function set. It flows first to the proper focal point, based on second-level routing parameters in the request, then to the entry point. From there, the request may flow to a named target application program or resource (again, based on second-level routing parameters), which may be more than one hierarchical level away from that entry-point function set. The target application program associated with such a request simply reverses the second-level origin and destination application program names found in the request to send any replies and reports.

1.7.1.4 MSU Differences

The operations management function set is designed to take advantage of the MDS transport. As a result, the operations management function set may be used only in implementations of CPMS, having the CP-MSU within the MDS-MU as the transport structure. Common operations services, while still supporting PUMS implementations (and therefore the NMVT structure), may exist in CPMS implementations also, sending and receiving data in the CP-MSU/MDS-MU format.

1.7.2 Common Operations Services Function Set

Common operations services are a set of services available to all of the major categories of network management. The architecture for common operations services provides a way of managing types of resources not explicitly addressed by the architecture defined for the individual categories. It does this by providing a general mechanism that allows a network operator to communicate with specialized network management application programs; these application programs, in turn, provide functions not currently provided by SNA management services.

A specific set of common operations services has been defined to assist a network operator with the task of controlling resources in the network. In order to manage a network, an operator requires the ability both to know about the various resources present in the network and to control those resources. Much of the necessary information about network resources is provided by the unsolicited messages defined by the management services architecture, e.g., the Alert. Other information is available via architecturally defined requests, such as QPI.

The common operations services for resource control provide for the collection of the additional information necessary to operate a network. They also provide a mechanism by which an operator can exert control over network resources, by directing commands to them.

The architecture for common operations services defines a number of capabilities to be used by application programs that provide network management services. The structures defined by this architecture serve network management application programs distributed in a network. Their role is to facilitate the transfer of information between two portions of a network management application program, or between a network operator and a remote network management application program. In this way it is possible for functions and/or resources not explicitly covered by the management services architecture to be brought within its overall network

management scheme.

Subtopics:

- 1.7.2.1 Functions Provided
- 1.7.2.2 Testing a Resource
- 1.7.2.3 Implementation
- 1.7.2.4 Format Usage

1.7.2.1 Functions Provided

The common operations services structures for resource control provide for the transport of information between network management application programs, or between a network operator and one such application program. The meaning of the information being transported is entirely defined by the application programs using the supporting services. The common operations services architecture provides, at the highest level, for two functions:

- The enveloping of uninterpreted data for transport through an SNA network

- A means of identifying a particular network management application program, or a particular (human or programmed) network operator, at the destination node as the intended recipient of this data.

Subtopics:

- 1.7.2.1.1 Enveloping of Data
- 1.7.2.1.2 Routing to Served Application Programs and Network Operators
- 1.7.2.1.3 Parameter Major Vectors
- 1.7.2.1.4 Commands
- 1.7.2.1.5 Messages to an Operator
- 1.7.2.1.6 Querying for Information on a Resource

1.7.2.1.1 Enveloping of Data

Two of the structures defined by common operations services for resource control are entirely free-form. The data provided by the served application programs is enveloped within subvectors that identify how the data is encoded (e.g., as character data in a specified coded graphic character set), but say nothing about the nature of the data. The data is then transported through the management services transport network, on the same sessions that carry such messages as Alerts.

The other three structures defined by common operations services identify the nature of the data they transport (for example, a query for information about a resource), as well as how it is encoded.

1.7.2.1.2 Routing to Served Application Programs and Network Operators

When a network operator enters a common operations services command, the name of a destination application program is also specified. Similarly, when an application program sends a message to a network operator, it also identifies the operator. Since the network operator in these cases can be programmed as well as human, the capability is provided for a network management application program at a host to communicate with a network management application program in the network, and vice versa.

See "Routing a Request to a Specified Application Program" in topic 1.2.5.3 for a complete discussion of this capability.

1.7.2.1.3 Parameter Major Vectors

As noted earlier in "Introduction to Management Services Formats," the common operations services architecture for resource control exploits the capability of the NMVT and CP-MSU to transport multiple major vectors. In addition to the usual management services major vector, certain NMVTs and CP-MSUs defined for common operations services for resource control contain one or more *management services parameter* major vectors, which appear after the usual major vector in the NMVT or CP-MSU. MS parameter

major vectors contain and group subvectors that carry the data associated with a single resource, or they indicate resource groupings. Figure 7-1 illustrates the two ways in which parameter major vectors are used by common operations services.

```
Single Parameter Major Vector:

|___________________NMVT or CP-MSU_______________________ÿ|
 ________ ________________________________ ______________
|(Header)| Common Operations Services MV | Parameter MV |
|________|________________________________|______________|
                                          .
                                          .
                                          ....Key = X'1300', X'1307', or X'1309'


Multiple Parameter Major Vectors:

|_______________________________NMVT or CP-MSU_________________________________ÿ|
 ________ ________________________________ ____________ _ _ _ _ _ ____________
|(Header)| Common Operations Services MV | X'130A' MV | X'1307'MVs| X'130B' MV |
|________|________________________________|____________| _ _ _ _ _ |____________|
                                                        .
                                                        .
                                                        ....zero or more instances
```

Figure 7-1. Parameter Major Vectors in Common Operations Services Encodings

Three MS parameter major vectors have been defined to handle the three types of data to be carried:

- Character strings (the Text Data major vector)
- Identified values (the Structured Data major vector)
- An undefined format (the Transparent Coded Datastream major vector).

Unlike other major vectors, the last of these MS parameter major vectors has no subvectors defined for it. The definition of its content is left totally up to the application programs using it.

The *Resource Data* subvector serves as the general carrier of identified values within the Structured Data major vector. This subvector carries a resource identifier in the form of a name or address and one of several defined types of data (character, integer, hexadecimal or bit-string) for each separately identified item of data reported for the resource.

Two additional MS parameter major vectors have been defined, to allow the limits of the set of MS parameter major vectors within a single NMVT or CP-MSU to be identified. The Begin Data Parameters MS parameter major vector is used to indicate that one or more Structured Data MS parameter major vectors may follow. Depending on what common operations services major vector it accompanies, the Begin Data Parameters may or may not have any Structured Data MS parameter major vectors following it. The End Parameter Data MS parameter major vector terminates all MS parameter major vectors. The use of these two parameter major vectors is illustrated in Figure 7-1.

Use of parameter major vectors provides the ability for a served application program to transport a large amount of data in a single reply. As a result, it is possible for a single reply to exceed the maximum allowable NMVT size. When this happens on an SSCP-PU session, the reply may span multiple NMVTs, where each NMVT contains a physical slice of the total reply. See "Overview of the Management Services Unit (MSU)" in topic 1.2.1.1.1 for a discussion of this support.

The capability of transporting data that spans multiple NMVTs is strictly limited to reply flows. All command and request flows, whether or not they contain parameter major vectors, must be fully contained in a single NMVT and are subject to a size limitation of 256 bytes. Unsolicited messages to an operator are similarly constrained but have an NMVT size limitation of 512 bytes.

1.7.2.1.4 Commands

The Execute Command MS major vector provides for transport of a command from a human or programmed network operator to a specified network management application program. The Execute Command major vector may carry with it one of three MS parameter major vectors, or the command may be placed within the Execute Command MS major vector itself.

The Reply to Execute Command major vector carries with it one of three MS parameter major vectors, and is routed to the originator of the Execute Command.

1.7.2.1.5 Messages to an Operator

The Send Message to Operator MS major vector provides for transport of a message from a network management application program to a specified (human or programmed) network operator. Its function is thus complementary to that of the Execute Command major vector. One of three MS parameter major vectors is carried with the Send Message to Operator major vector.

1.7.2.1.6 Querying for Information on a Resource

The explicit request for information relating to a named resource (or multiple named resources) is provided in the Query Resource Data MS major vector. The request always contains the name of one or more resources.

The Reply to Query Resource Data MS major vector is followed by the Begin Data Parameters and End Parameter Data pair of MS parameter major vectors and one Structured Data MS parameter major vector for each resource.

1.7.2.2 Testing a Resource

The Test Resource MS major vector provides the same list of names as Query Resource Data, plus a Test Request subvector. Reply to Test Resource uses the same MS parameter major vector structure for reporting information as does Reply to Query Resource Data. The Reply to Test Resource MS major vector carries additional information in the Test Result subvector.

The only type of test identified in the request is a "Self Test." The number of times the test is executed may be specified. A summary indication of the results of the test is returned, along with detailed information on each resource included in the test.

Subtopics:

- 1.7.2.2.1 Analysis of a Resource

1.7.2.2.1 Analysis of a Resource

The Analyze Status MS major vector is very specifically an extension of the Alert function. It uses the same list of resource names in the request as the Query Resource Data and Test Resource major vectors, and returns a Structured Data MS parameter major vector series similar to those returned by the Reply to Query Resource Data and Reply to Test Resource major vectors, but without the detailed data. Instead, the Reply to Analyze Status major vector returns a Probable Causes subvector identical to that found in an Alert. The Probable Causes subvector is returned in the Begin Data Parameters MS parameter major vector, and applies to the entire set of resources identified in the request. The Structured Data major vectors for each resource then identify the individual resources by name and type.

The Analyze Status MS major vector is sent to an application program that manages a resource suspected of failure, but for which there has been no Alert received.

1.7.2.3 Implementation

Support of common operations services for resource control is provided in an entry point by "EP_COMMON_OPERATIONS_SERVICES Function Set" in topic 2.10.5. This is an optional function set implemented by type 2.0, LEN end nodes, and APPN nodes.

1.7.2.4 Format Usage

```
NETWORK MANAGEMENT VECTOR TRANSPORT (NMVT)
   Execute Command MV (X'8061')
   Analyze Status MV (X'8062')
   Query Resource Data MV (X'8063')
   Test Resource MV (X'8064')
   Text Data Parameter MV (X'1300')
   Structured Data Parameter MV (X'1307')
   Transparent Coded Datastream Parameter MV (X'1309')
   Flow:  From SSCP to PU T2 (Normal)


NETWORK MANAGEMENT VECTOR TRANSPORT (NMVT)
   Reply to Execute Command MV (X'0061')
   Reply to Analyze Status MV (X'0062')
   Reply to Query Resource Data MV (X'0063')
   Reply to Test Resource MV (X'0064')
   Send Message to Operator MV (X'006F')
   Text Data Parameter MV (X'1300')
   Structured Data Parameter MV (X'1307')
   Transparent Coded Datastream Parameter MV (X'1309')
   Begin Data Parameters Parameter MV (X'130A')
   End Parameter Data Parameter MV (X'130B')
   Flow:  From PU T2 to SSCP (Normal)


CONTROL POINT MANAGEMENT SERVICES UNIT (CP-MSU)
   Execute Command MV (X'8061')
   Analyze Status MV (X'8062')
   Query Resource Data MV (X'8063')
   Test Resource MV (X'8064')
   Reply to Execute Command MV (X'0061')
   Reply to Analyze Status MV (X'0062')
   Reply to Query Resource Data MV (X'0063')
   Reply to Test Resource MV (X'0064')
   Send Message to Operator MV (X'006F')
   Text Data Parameter MV (X'1300')
   Structured Data Parameter MV (X'1307')
   Transparent Coded Datastream Parameter MV (X'1309')
   Begin Data Parameters Parameter MV (X'130A')
   End Parameter Data Parameter MV (X'130B')
   Flow:  From CP to CP
```

A network operator can send a common operations services command to a specified served application program at a specified PU or CP. Data necessary for creating the request is passed to the requesting CPMS.

CPMS formats the request into one of four common operations services MS major vectors:

- Execute Command (X'8061')
- Analyze Status (X'8062')
- Query Resource Data (X'8063')
- Test Resource (X'8064').

CPMS may also create one of three parameter major vectors when it formats the request into the Execute Command (X'8061') MS major vector. CPMS then constructs an NMVT containing the major vector, and sends it to PUMS.

When CPMS or PUMS receives the request, it passes it to the served application program identified in the Name List (X'06') subvector.

After performing the requested function, the application program to which the request was passed creates one of the following four common operations services major vectors, as appropriate, in response to the request:

- Reply to Execute Command (X'0061')
- Reply to Analyze Status (X'0062')
- Reply to Query Resource Data (X'0063')
- Reply to Test Resource (X'0064').

It also creates one or more parameter major vectors. The application program passes the major vectors it has created to CPMS or PUMS. PUMS envelopes the major vectors into one or more NMVTs and sends them to CPMS where any necessary concatenation of NMVTs is performed. CPMS optionally logs the reply and passes it to the network operator. The flow is of the type described in "Request/Reply Flows for Non-Bulk Data" in topic 1.2.2.3. Refer to Figure 7-2 and Figure 7-3 for example flows.

```
        Network                                                   Served
        Operator                 CPMS               PUMS         Application
           .                       .                  .                .
           .   command and served  .                  .                .
           .   application name    .                  .                .
           º_______________________ÿ                  .                .
           .                       .  NMVT (X'806x')³ .                .
           .                       º__________________ÿ                .
           .                       .                  .                .
           .                       .    +RSP(NMVT)    .                .
           .                       __________________º                .
           .                       .                  .      command   .
           .                       .                  º________________ÿ
           .                       .                  .                .
           .                       .                  .  requested data .
           .                       .                  __________________º
           .                       .  NMVT (X'006x')¹².                .
           .                       __________________º                .
           .                       .    +RSP(NMVT)²   .                .
          .                       º__________________ÿ                .
           .     requested data    .                  .                .
           _______________________º                  .                .
           .                       .                  .                .
           .                       .                  .                .
```

¹Plus one or more parameter major vectors.
²When the requested data exceeds the maximum NMVT size, this NMVT/response exchange occurs once for each NMVT.
³May also contain a parameter major vector for X'8061'.

Figure 7-2. Example Flow Showing Common Operations Services Request and Reply. The NMVTs are transported between the CP and the type 2.0 (or boundary-function-attached type 2.1 node) over the SSCP-PU session.

```
 Network               Requesting             Destination                Served
 Operator                CPMS                    CPMS                  Application
   .                      .                       .                        .
   .  command and served  .                       .                        .
   .   application name   .                       .                        .
   º______________________ÿ                       .                        .
   .                      .    CP MSU (X'806x')²  .                    .
   .                      º______________________ÿ                        .
   .                      .                       .                        .
   .                      .                       .         command        .
   .                      .                       º______________________ÿ
   .                      .                       .                        .
   .                      .                       .     requested data     .
   .                      .                       _______________________º
   .                      .    CP_MSU (X'006x')¹  .                        .
   .                      _______________________º                       .
   .    requested data    .                       .                        .
  _______________________º                       .                        .
   .                      .                       .                        .
   .                      .                       .                        .
```

¹Plus one or more parameter major vectors.
²May also contain a parameter major vector for X'8061'.

Figure 7-3. Example Flow Showing Common Operations Services Request and Reply for an APPN Node

A served application program can also send a text message to a network operator at a specified node. It builds a Send Message to Operator (X'006F') major vector, plus one of three parameter major vectors (Text Data (X'1300'), Structured Data (X'1307'), or Transparent Coded Datastream (X'1309')); then it passes these major vectors to CPMS or PUMS in its node. CPMS envelopes the major vector in a CP-MSU, which it sends to the destination CPMS over the CP-CP or LU-LU sessions used by multiple-domain support. PUMS envelopes the major vector in an NMVT and sends it to CPMS over the SSCP-PU session. CPMS optionally logs the message and passes it to the network operator. The network operator then passes the message to the individually-identified operator to whom it was directed. The flow is of the type described in "Unsolicited Flows" in topic 1.2.2.1. Refer to Figure 7-4 and Figure 7-5 for example flows.

```
        Network                                                   Served
        Operator                 CPMS               PUMS         Application
```

```
              .                       .                  .                  .
              .                       .                  .    message and   .
              .                       .                  .  operator's name .
              .                       .                  _________________°
              .                       .  NMVT (X'006F')¹ .                  .
              .                       _______________°                     .
              .                       .    +RSP(NMVT)    .                  .
              .                       °_________________ÿ                  .
              .         message       .                  .                  .
             _______________________°                   .                  .
              .                       .                  .                  .
              .                       .                  .                  .
```

¹Plus one parameter major vector.

Figure 7-4. Example Flow Showing Common Operations Services Send Message to Operator

```
 Network               Destination              Origin                    Served
 Operator                CPMS                    CPMS                   Application
   .                       .                       .                        .
   .                       .                       .      message and       .
   .                       .                       .     operator's name    .
   .                       .                       _______________________°
   .                       .    CP_MSU (X'006F')¹  .                        .
   .                       _______________________°                        .
   .        message        .                       .                        .
  _____________________°                           .                        .
   .                       .                       .                        .
   .                       .                       .                        .
```

¹Plus one parameter major vector.

Figure 7-5. Example Flow Showing Common Operations Services Send Message to Operator for an APPN Node

---

1.7.3 Operations Management Function Set

Managers of large enterprise networks require the ability to operate systems in the network from a central site. There are several benefits to this approach:

- Labor costs are reduced, since the requirement to attend systems locally can be eliminated (or at least the required skill level of those people attending the local systems can be reduced).

- Operations management commands and procedures are made uniform across the network.

Operations management extends the functions of common operations services and change management, improving a network manager's ability to control distributed system resources. Additionally, if an interface between the operations management and problem management focal-point operators is implemented, then operations management commands may be used to assist in problem bypass and recovery. See "EP ALERT Optional Subset 14 (Operations Management Support)" in topic 2.10.1.18 for more information.

Subtopics:

- 1.7.3.1 Functions Provided
- 1.7.3.2 Implementation
- 1.7.3.3 Format Usage

1.7.3.1 Functions Provided

Control of remote resources involves many different functions.

Requests, Replies, and Reports:

The operations management architecture reduces the major functions into five requests with corresponding replies and reports.

In the context of operations management, a request initiates an action at the target. A report may indicate acceptance or rejection of the request or it may indicate the status of the execution of the request, if in fact the request was accepted. An acceptance or rejection report is referred to as a reply where the function provides a unique CP-MSU for this initial reporting. A status report is part of the same second-level application program unit of work as the request and reply (acceptance/rejection report/reply). The relationship between MDS request/reply flows and operations management requests, replies, and reports is explained in "MDS Message Types and Application-Level Flows" in topic 1.7.3.3.1.

- Request Activation CP-MSU

  Activating resources includes such functions as power-on and initial microcode load (IML), the extent of the activation being based on a configuration profile stored at the receiver of the Activate request (and named in the request itself).

- Request Deactivation CP-MSU

  Deactivation performs the opposite of activation, with the target resource being reduced from being a communicating entity to being logically inoperative.

- Set Clock CP-MSU

  This function allows a central site to synchronize the clocks in a network and to indicate offsets from different timing sources.

- Request Cancelation CP-MSU

  The architecture allows for a single, specific command to be canceled, or for *all* commands meeting a wide variety of criteria to be canceled. The commands in question can be those that are held (awaiting execution by the target), or, alternatively, a command that is currently executing at the target location.

- Operate CP-MSU

  The ability to transport implementation-specific commands (commands not defined by SNA/MS) is provided by this structure.

Each of these commands is paired with reporting functions that indicate whether or not the command is accepted by the target. If the command is accepted, additional reports can be sent back to the operator to indicate the success or failure of the execution. If the command is not accepted (or a failure occurs during execution), the event is described in a report containing appropriate sense data.

Second-Level Routing:

Second-level routing refers to the process of routing MSUs to application programs that are served by an architecturally defined management services application program; in this case, this means routing to application programs served by EP_OPERATIONS_MGMT and the operations management focal point.

Operators also need to know about command routing errors, at both the transport and second-level application program layers. Transport-level error reporting is accomplished by MULTIPLE_DOMAIN_SUPPORT, using the format described in "The MDS Error Message" in topic 1.2.3.2.2, while errors in second-level routing are handled by operations management's Routing/Parsing Report CP-MSU. More detail can be found in "The Routing and Targeting Instructions GDS Variable" in topic 1.7.3.3.1 and in "Routing a Request to a Specified Application Program" in topic 1.2.5.3.

The SNA Condition Report (X'1532') GDS variable (SNACR) within the Routing/Parsing Report CP-MSU contains the details of the error that has occurred.

Execution Window Timing:

The Execution Window Timing (X'0B') MS Common Subvector provides a means to specify a "window of time" during which the execution of a command is to begin. When multiple execution windows are scheduled, they are fixed in time regardless of the beginning or ending of the command executions

with which they are associated. In other words, the beginning and end of an execution window is *not* based on the time that a prior command execution took place.

If both the Execution Window Timing subvector and a Force Indication subfield are present in the command, and the Force Indication subfield specifies an amount of time to wait for the target to enter a quiesce state before execution is to begin, then the Execution Window Timing subvector governs the *beginning* of the waiting period. (1)

The window specified by the Execution Window Timing subvector is defined by the time values in either an Execute Not Before Time subfield, an Execute Not After Time subfield, or the time values of both subfields together. Optionally, a Time Interval and an Execution Count subfield may be included within the subvector to indicate that a command is to be executed multiple times, as explained below:

° If the Time Interval subfield is present and the Execution Count subfield is *not* present, then the command is executed repetitively until a Cancel command is sent to cancel the request. The time interval is added to the Execute Not After Time subfield value, or the Execute Not Before Time subfield value (or both values if they are present in the request) to establish a window for the next iteration.

° If the Time Interval subfield is present and the Execution Count subfield *is* present, the execution count specifies the total number of times the request is to be executed. The time interval is added to the Execute Not After Time subfield value, or the Execute Not Before Time subfield value (or both values if they are present in the request) to establish a window for the next iteration.

The Execute Not Before Time subfield or the Execute Not After Time subfield, or both, and the Time Interval subfield must be present in order to include the Execution Count subfield.

The time window established by the Execution Window Timing subvector is either an *implicit* window or an *explicit* window. If only an Execute Not After Time subfield or an Execute Not Before Time subfield is specified (with or without a time interval or execution count), then the window is an *implicit* window, as explained below:

° If only an Execute Not After Time subfield is present in the subvector, the implicit beginning of the time window is the time at which the target node receives the request. The end of the window is defined by the value of the Execute Not After Time subfield.

° If a Time Interval subfield is present with only an Execute Not After Time subfield, the implicit beginning of the initial time window is the time at which the target node receives the request. Refer to Figure 7-6.

The implicit beginning of the second window is the time specified by the Execute Not After Time subfield of the initial window. The end of the second window is calculated by adding the value in the Time Interval subfield to the value in the Execute Not After Time subfield.

After the establishment of the second window, the time values for each of the windows 3-n ("n" being determined by the Execution Count subfield or the reception of a Cancel command) are calculated by adding the value in the Time Interval subfield to the beginning time value of the previous window.

Note that when an Execute Not Before Time subfield is not present, the beginning time of each window after the initial window occurs no earlier than the time the prior window ends.

```
            Window 1     Window 2        Window 3        Window n


        T(r)       T(na)           T(i)            T(i)            T(i)
  time
  °°°°°°ÿ|__________|_______________|_______________|_______________|°°°°ÿ
        .          .               .               .               .
        .          .               .               .               .
        .          .   interval    .   interval    .   interval    .
        .          ._______________ÿ._______________ÿ._______________ÿ.
        .          .               .               .               .
```

Figure 7-6. Time Windows: Execute Not After Time and Time Interval.
Note that the windows in this figure are *implicit* windows.

Legend:

T(r) Time, at the receiving node, when the request is received. Defines the beginning of the first window.

T(na) Time value in the Execute Not After Time subfield. Defines the end of the first window and the beginning of the second window.

T(i) Time, after interval value applied. The first occurrence of T(i) defines the end of the second window and the beginning of the third window.

° If only an Execute Not Before Time subfield is present in the subvector, the beginning of the time window is the value in the Execute Not Before Time subfield. The end of the window is determined by either of two methods:

- An implementation-specific time-out value defines the latest time that the command can be considered for execution. If the request has not been executed by this time (or the requester has received no reply), an implementation may choose to send the command again. An example of such a timer is the one used to determine the maximum time that MDS will wait for an MDS reply sent in response to an MDS request.

- A cancel command is sent, canceling the request, effectively ending the time window.

° If a Time Interval subfield is present with only an Execute Not Before Time subfield, the implicit end of the initial execution timing window is calculated by adding the value in the Time Interval subfield to the time specified by the Execute Not Before Time subfield. The ending times of the remaining windows are calculated by adding the interval time to the ending time of the previous window. Refer to Figure 7-7.

Note that while an Execute Not After Time subfield is not present, the ending time of each window after the first window occurs no later than the time the next window begins.

```
                            Window 1        Window 2        Window n


       T(r)            T(nb)           T(i)            T(i)            T(i)
 time
 °°°°°°ÿ|°°°°°°°°°°°°°°°|_______________|_______________|_______________|°°°°ÿ
       .               .               .               .               .
       .               .               .               .               .
       .               .   interval    .   interval    .   interval    .
       .               ._______________ÿ._______________ÿ._______________ÿ.
       .               .               .               .               .
```

Figure 7-7. Time Windows: Execute Not Before Time and Time Interval.
Note that the windows in this figure are *implicit* windows.

Legend:

T(r) Time, at the receiving node, when the request is received

T(nb) Time value in the Execute Not Before Time subfield. Defines the beginning of the first window.

T(i) Time, after interval value applied. The first occurrence of T(i) defines the end of the first window and the beginning of the second window.

An *explicit* window is one that is established by having both the Execute

Not Before Time and the Execute Not After Time subfields present in the request. If a Time Interval subfield is present, then its value is added to the values in both the Execute Not Before Time and Execute Not After Time, resulting in a the definition of the second window. For windows 3-n, the interval value is added to the beginning and ending time values of the previous window. The beginning time of each window after the first window occurs no earlier than the ending time of the prior window. Refer to Figure 7-8.

```
                        Window 1                                Window 2


        T(r)        T(nb1)          T(na1)                  T(nb2)          T(na2)
 time
 °°°°°°ÿ|°°°°°°°°°°°°|_______________|°°°°°°°°°°°°°°°°°°°°°°°|_______________|°°°°°°ÿ
                     .               .                       .               .
                     .               .                       .               .
                     .            interval                   .               .
                     ._________________________________________ÿ.              .
                     .               .                       .               .
                                     .                                       .
                                     .               interval                .
                                     ._________________________________________ÿ.
                                     .                                       .
```

Figure 7-8. Time Windows: Execute Not Before/Execute Not After/Time Interval. Note that the windows in this figure are *explicit* windows.

Legend:

T(r) Time, at the receiving node, when the request is received

T(nb1) Time value in the Execute Not Before Time subfield. Defines the beginning of the first window.

T(na1) Time value in the Execute Not After Time subfield. Defines the end of the first window.

T(nb2) Time value, calculated by adding the value of the Time Interval subfield to T(nb1). Defines the beginning of the second window.

T(na2) Time value, calculated by adding the value of the Time Interval subfield to T(na1). Defines the end of the second window.

An explicit window offers the flexibility of having an execution window time span that is less than a time interval span. For example, an implementation may choose to tightly control the time allowed for a request execution to take place. Perhaps a window of only 15 minutes is required. An additional requirement may be that the request is to be executed once every 30 days. Unless an explicit window is encoded, there is no way to satisfy both requirements.

| *Operations Management Use of the Context-Identified Values (X'1323') GDS variable*:

| ° Context-Identified Values (X'1323') GDS variable

| The Context-Identified Values GDS variable allows information to be enveloped in a way such that its meaning is evident from the context in which it appears. In other words, the parameters that are carried are to be interpreted only within the context of the fields that enclose those parameters.

| In operations management, the Context-Identified Values GDS variable is used to transport implementation-specific commands. The format of these commands follow this basic structure:

| - Verb
| - Operand name
| - Operand value

| The operand name is to be interpreted only within the context of the verb. The operand value is to be interpreted only within the context

of the operand name.

The sequence of *operand name/operand value*, within the verb specification, may be repeated several times if necessary. For example, an operator may want to send a command causing the target to return the contents of two different types of registers (64-bit and 32-bit), each type having three individual registers (numbered 0 through 2). The *operand name/operand value* sequence would be repeated for each register.

To encode a command with repeating operand names and values, the concept of *nesting* is introduced. Nesting allows repeating structures to be specified within one single, enveloping field, as shown below:

```
Verb           = Display
Operand Name   = 64-bit Data
Operand Value  = (nested operand names and values)

                 Operand Name = 64-bit register

                 Operand Value = 0

                 Operand Name = 64-bit register

                 Operand Value = 1

                 Operand Name = 64-bit register

                 Operand Value = 2

Operand Name   = 32-bit Data
Operand Value  = (nested operand names and values)

                 Operand Name = 32-bit register

                 Operand Value = 0

                 Operand Name = 32-bit register

                 Operand Value = 1

                 Operand Name = 32-bit register

                 Operand Value = 2
```

The example Context-Identified Values structure above would be interpreted as "Display the data associated with 64-bit registers 0 through 2 and 32-bit registers 0 through 2."

A more efficient method can be used to accomplish the same task, using the *array* concept. An enveloping field can be encoded as a one-dimensional array of values, each value in the array being equal in length to all other values in the array. Thus, the example above could be encoded as follows:

```
Verb           = Display

Operand Name   = Data

Operand Value  = (nested operand names and values)

                 Operand Name:  64-bit register

                 Operand Value:  array=64BR0.64BR1.64BR2

                 Operand Name = 32-bit register

                 Operand Value:  array=32BR0.32BR1.32BR2
```

For clarity in the example above, a period separates the array elements. These periods would not appear in the actual encoding.

Table 7-1 illustrates the above concepts using the actual Context-Identified Values structures:

Table 7-1. Example Context-Identified Values (X'1323') GDS variable

Hex	DContents	Nested ID/T	Nested Contents
1323	Context-Identified Values		
13	4 Context Identifier Grou		
	325 Context Identifier. " since no Value Proces (X'1328') present)	isplay" (enco ing Method	ed in hex
13	6 Value Group		
	32A Hexadecimal Value Des (encoded in hex)	riptor. "Dat	,"
	32B Nested Value Processi	g Method	
	01 Value Type. Indica of nested 1326s	es 1329 is co	posed
	329 Value (contents in Ne	t1326D/T colu	nValue>Group
		132A	Hexadecimal Value Desc "64-bit register" (enc
		132B	Nested Value Processin
		01	Value Type. Indicat
		03	Array Entry Length. is an array.
		1329	Value. Array, "64BR064BR164BR2," in (array of all 64-bit r
		132A	Hexadecimal Value Desc "32-bit register" (enc
		132B	Nested Value Processin
		01	Value Type. Indicat
		03	Array Entry Length. is an array.
		1329	Value. Array, "32BR032BR132BR2," in (array of all 32-bit r

The Context-Identified Values is a structured enveloping method. It allows for data to be identified within a certain specified context, thus ensuring that the sender and receiver interpret the enveloped data in the same way. The actual Context-Identified Values structures follow:

- Value Processing Method (X'1328'): indicates how the Context Identifier (X'1325') and any Value (X'1329') GDS variables are encoded. Required only if the default processing method is different from hexadecimal.

  -- Value Type (X'01')

  - Hexadecimal (X'01')

  - Character (X'02')

  - Unsigned decimal integer (X'03')

  - Bit string (X'04')

  - Processing method implied by context ID (X'05')

  -- Coded Character Set ID (X'02')

  If Value Type is character, the coded graphics character set is identified here.

  -- National Language ID (X'12')

  Identifies the national language in which a text string in written.

```
|     -   Context Identifier Group (X'1324')

|         --  Value Processing Method (X'1328'):  indicates how the Context
|             Identifier (X'1325') GDS variable within this X'1324' is
|             encoded.  Required only if processing method of context
|             identifier is different from default processing method.

|             °   Value Type (X'01')

|                 -   Hexadecimal (X'01')

|                 -   Character (X'02')

|                 -   Unsigned decimal integer (X'03')

|                 -   Bit string (X'04')

|                 -   Processing method implied by context ID (X'05')

|             °   Coded Character Set ID (X'02')

|                 If Value Type is character, the coded graphics character
|                 set is identified here.

|             °   National Language ID (X'12')

|                 Identifies the national language in which a text string in
|                 written.

|         --  Context Identifier (X'1325'):  the "verb" of the command.

|     -   Value Group (X'1326')

|         --  Value descriptor:  the "operand" of the command

|             °   Character Value Descriptor (X'1327'):  if operand is in
|                 character format (coded graphics character set
|                 01134-00500)

|             °   Hexadecimal Value Descriptor (X'132A'):  if operand is in
|                 hexadecimal format

|         --  Nested Value Processing Method (X'132B'):  indicates how the
|             Value (X'1329') contained within this X'1326' is encoded.
|             Required only if processing method of value is different from
|             default processing method.

|             °   Nested Value Type (X'01')

|                 -   Hexadecimal (X'01')

|                 -   Character (X'02')

|                 -   Unsigned decimal integer (X'03')

|                 -   Bit string (X'04')

|                 -   Processing method implied by context ID (X'05')

|                 -   Value contains one or more nested value groups (X'06')

|             °   Coded Character Set ID (X'02')

|                 If Nested Value Type is character, the coded graphics
|                 character set is identified here.

|             °   Array Entry Length (X'03')

|                 -   Presence indicates that the X'1329' is an array.  The
|                     length specified is the length of each element in the
|                     array, all elements being equal in length.

|             °   National Language ID (X'12')
```

```
|                 -   Identifies the national language in which a text
|                     string in written.

|         --  Value Instance Identifier (X'1331')

|             °   An implementation-defined token that identifies a
|                 particular Value Group (X'1326') GDS variable.

|         --  Value (X'1329')

|             Composed of one of the following:

|             °   A single scalar value

|             °   One or more Value Group (X'1326') GDS variables

|             °   A one-dimensional array of values, each value being equal
|                 in length to all other values within the array
```

Operations Management Use of the SNA Condition Report (SNACR):

The SNACR (X'1532') can be thought of as a menu of structures that can be used as an architecture or implementation requires. It is not necessary to include every SNACR substructure when reporting an error condition. See *SNA Formats* for a complete description of the SNACR encodings.

If, for example, a request is sent and the receiver (either EP_OPERATIONS_MGMT or a second-level application program that EP_OPERATIONS_MGMT serves) does not recognize the name in the Destination Instance Identifier (X'70') subfield within the Name List (X'06') MS Common subvector of the Routing and Targeting Instructions (X'154D') GDS variable, a Routing/Parsing Report (X'0077') CP-MSU is sent by the receiving application program back to the origin of the request. A SNACR identifying the unrecognized subfield is structured as shown in Table 7-2.

Table 7-2. An Example SNACR Structure

Structure Name	ID/T	Contents
SNA_CONDITION_REPORT	X'1532'	
SNA_REPORT_CODE	X'7D'	X'100B000C' ("Format exception, unknown or unsupported data value")
STRUCTURE_REPORT	X'01'	
STRUCTURE_STATE	X'01'	X'01' (Present)
PARENT_SPEC	X'03'	
PARENT_ID_OR_T	X'01'	X'06' (key of Name List MS Common Subvector)
PARENT_CLASS	X'02'	X'02' (X'06' is type LT)
STRUCTURE_SPEC	X'04'	
STRUCTURE_ID_OR_T	X'01'	X'70' (key of Destination Instance Identifier)
STRUCTURE_CLASS	X'02'	X'02' (X'70' is type LT)

The STRUCTURE_STATE tells whether the structure being reported on is absent or present. In this example, it is present.

The PARENT_SPEC identifies the parent structure of the structure being reported on.

The PARENT_ID_OR_T field specifies the key of the parent. In this case, the parent is the Name List (X'06') MS Common subvector.

PARENT_CLASS indicates what kind of structure the parent is (e.g., LLID, LT, Delimited LT) The Name List is an LT structure, indicated by X'02'.

The STRUCTURE_SPEC identifies the structure being reported on.

The STRUCTURE_ID_OR_T field specifies the actual key of the structure in question. In this case, the key is X'70', which corresponds to the Destination Instance Identifier subfield of the Name List subvector.

The STRUCTURE_CLASS indicates the class of the structure in question. The Destination Instance Identifier is an LT structure, indicated by X'02'.

If a particular structure is not present, an SNACR structure similar to the one above would be used; however, the STRUCTURE_STATE would indicate "absent" (X'02') and there would be a different SNA_REPORT_CODE. The structure in question would be identified in the same manner as an unrecognized structure.

In summary, the SNACR allows implementers to use generalized report codes to define errors. The details of the error location and associated information can be placed in the various SNACR sub-structures, therefore allowing a more flexible reporting vehicle while controlling the number of new report codes defined.

Correlating Requests, Replies, and Reports:

MDS Requests and replies are related by using the Agent Unit-of-Work Correlator (X'1549') GDS variable (AUOWC) present in the MDS header. See Table 2-4 in topic 1.2.3.2.2 for a description of the Agent Unit of Work. Both the entry-point and focal-point application programs and application programs served by the entry points and focal points ("second-level" application programs) may make use of the Agent Unit-of-Work Correlator (X'1549') GDS variable for correlation purposes. At the level of the entry-point and focal-point application programs, the AUOWC enables MDS to correlate requests with replies. However, the second-level application programs also use the AUOWC, or an implementation-specific correlator, to correlate their requests, replies, and reports. This second-level correlation is transparent to MDS. See "MDS Message Types and Application-Level Flows" in topic 1.7.3.3.1 for more information.

Operations Management Entry-Point MS Application Program Name:

The MS entry point application program name for operations management is:

° EP_OPERATIONS_MGMT (X'23F0F1F6')

---

(1)

A Force Indication (X'10') subfield, if present in the command subvector, specifies whether or not the command is to be executed based on the quiesce state of the target. The optional delay time parameter of the Force Indication subfield reflects how long the command issuer is willing to wait for the target to enter a quiesce state. Depending on the implementation, the end of the delay period may occur after the end of an execution timing window. For this reason, implementations should carefully plan multiple execution timing windows and force indication delay times such that command executions are scheduled in a nonoverlapping manner. More information concerning the Force Indication subfield can be found in the descriptions of the operations management request CP-MSUs.

Subtopics:

- 1.7.3.1.1 Operations Management Requests, Replies, and Reports
- 1.7.3.1.2 Request Activation CP-MSU
- 1.7.3.1.3 Activation Acceptance CP-MSU
- 1.7.3.1.4 Activation CP-MSU
- 1.7.3.1.5 Request Deactivation CP-MSU
- 1.7.3.1.6 Deactivation Acceptance CP-MSU
- 1.7.3.1.7 Deactivation CP-MSU
- 1.7.3.1.8 Set Clock CP-MSU
- 1.7.3.1.9 Set Clock Report CP-MSU
- 1.7.3.1.10 Request Cancelation CP-MSU
- 1.7.3.1.11 Cancelation CP-MSU
- 1.7.3.1.12 Operate CP-MSU
- 1.7.3.1.13 Operate Report CP-MSU
- 1.7.3.1.14 Routing/Parsing Report CP-MSU

1.7.3.1.1 Operations Management Requests, Replies, and Reports

| Described below are requests and their parameters, acceptance report (or
| reply) contents, and status report contents. Each request, reply, and report is referred to as a CP-MSU structure, and is qualified by the name of the major vector that flows within it. For example, the CP-MSU structure that carries the Request Activation major vector is referred to as the *Request Activation CP-MSU*.

Each description is preceded by a table showing the names and hexadecimal keys of the formats used, as well as the number of times each format may occur within its "parent" (enclosing) format. An occurrence of "0-1" means that the structure is optionally present one time within its parent. An occurrence of "1" means that the structure is required to be present once within its parent.

Following each request, reply or report is a table containing an example of that structure from the CP-MSU level down.

Refer to *SNA Formats* for byte-level descriptions of the encodings.

1.7.3.1.2 Request Activation CP-MSU

The structure of the Request Activation CP-MSU is shown in Table 7-3.

Table 7-3. The Request Activation CP-MSU

Structure Name	ID/T	Occurrences Within Parent
CP-MSU	1212	__
Routing and Targeting Instructions	154D	0-1
Context-Identified Values	1323	0-1
Request Activation	8066	1
Reporting Level	09	0-1
Configuration Name	07	0-1
Execution Window Timing	0B	0-1
Execute Not Before Time	10	0-1
Execute Not After Time	20	0-1
Time Interval	30	0-1
Execution Count	40	0-1
Activate Command Subvector	81	1
Force Indication	10	1
Change Management Activation Use	20	0-1

The architecture for activation relies on the notion of a *configuration name*. Entry-point implementations can use this name to identify either a particular resource (the simple case) or a file containing a set of resources and default instructions. In the latter case, it is assumed that overriding of those defaults can be done either using one of the transport envelopes from the focal point or by using an implementation of the architecturally defined Send and Install function (see "VERB: Send_and_Install" in topic B.5.1) against a customizing data file. The format of the configuration name is designed to fit into one of the identification tokens in the SNA/FS global name of the data object as specified by a change management network planner (up to 16 bytes, EBCDIC, CGCSID 01134-00500). Note that the EBCDIC coded graphics character set ID 01134-00500 is equivalent to the ASCII code page 00850. For further information on FS global names, refer to Appendix C, "SNA/FS File Names Defined by SNA/MS" in topic C.0.

The Activate command is a request to an entry point to activate a system resource under its control. Successful activation implies that all functions relevant to that component are active. This includes both hardware and software components in the resource being activated. Defaults must be available at the entry point for this to be feasible. The architecture allows for reporting to be staged if the execution is successful (see "Activation CP-MSU" in topic 1.7.3.1.4 for more information).

The parameters specified in the Reporting Level (X'09') MS Common Subvector determine whether or not replies and reports (as well as the number of reports) are returned to the request originator.

Required Parameters

- Force Indication (X'10') Activate subfield

- Force(No): Reject command if target not quiesced

  -- Option: Specify a maximum number of seconds, minutes, or hours to wait, after the command is received, before rejecting the command. The Execution Window Timing (X'0B') MS common subvector, if present in the request, specifies when to start the waiting period. If the target is not quiesced during the waiting period, the command is rejected.

- Force(Yes): Execute command

  -- Option: Specify a maximum number of seconds, minutes, or hours to wait, after the command is received, before executing the command. The Execution Window Timing (X'0B') MS common subvector, if present in the request, specifies when to start the waiting period. If the target is not quiesced during the waiting period, the command is still executed.

- If the waiting period option is not encoded, it is implied that the waiting period will be 0. The command will be rejected/executed immediately, depending on whether Force(No) or Force(Yes) was specified.

Optional Parameters

° Change Management Activation Use (X'20') Activate subfield

Both change management and operations management use this subfield for the same purpose: to control which versions of entry-point software components are used in an activation. This subfield occurs here because an implementation may choose to do activation via the Operate command.

This subfield causes the entry point to base its activation on *trial* and/or *production* versions of software components. Alternatively, the network operator may specify that the versions used in the previous (last) activation are to be used in the present activation.

Changes that cannot be tested fully or those having a strong potential to disrupt entry-point processing are best installed on trial. After activation and a period of successful operation, the activation may be issued again, specifying that production versions are to be used.

Note: This parameter is conditionally present. It is always present unless the requester knows by some means that the target has not implemented change management or change management activation use (and is therefore incapable of interpreting the meaning of this subfield). If this subfield is present and the target of the request does *not* support change management or this subfield, then the target of the request sends an Activation Acceptance CP-MSU (containing a SNACR with details of the error) to the request originator.

° Reporting Level (X'09') MS common subvector

This subvector indicates whether or not acceptance replies and status reports are to be generated by the request target. If status reports are requested, then the type of status report is also specified.

- When indicating acceptance/rejection:

  Rejection The target will not execute the associated command.
  Acceptance The target will attempt to execute the associated command.

- When indicating status after acceptance:

  Success The target successfully completed execution of the command.
  Failure The target detected an error while attempting to execute the command. There is no attempt to re-execute the command.
  Intermediate The target has status about the associated command, but the command has not finished executing.

For each report, a requirement parameter is given. These parameters are:

Required The target is required to return a report of this category.
Allowed The target may, at its discretion, return a report of this category.
Not Allowed The target shall not return a report of this category.

If this subvector is not included in the request, the target sends its reply and reports according to the defaults, listed below:

- Rejection replies are required to be sent.
- Acceptance replies are allowed to be sent (but are not required).
- Success reports are required to be sent.
- Failure reports are required to be sent.
- Intermediate reports are not allowed to be sent.

Each use of this subvector applies only to the command in which it is contained. The defaults are not replaced but only overridden for the duration of each command request/report sequence.

° Execution Window Timing (X'0B') MS common subvector: the earliest or latest (or both earliest and latest) time that the command may be executed.

Optionally, a Time Interval subfield and an Execution Count subfield may be included to indicate that the command is to be executed multiple times. See "Execution Window Timing" in topic 1.7.3.1 for more information.

° Configuration Name (X'07') MS common subvector: if not specified, then reactivation of the entire receiving node is implied.

° Routing and Targeting Instructions (X'154D') GDS variable: identifies a second-level application program target and optionally the path to that target. If the originator of the request is a second-level application program, then it is also identified here. In addition, this GDS variable contains an indication of how the Agent Unit of Work GDS variable is to be interpreted by the second-level application programs involved. See "The Routing and Targeting Instructions GDS Variable" in topic 1.7.3.3.1 and "MDS Message Types and Application-Level Flows" in topic 1.7.3.3.1 for more information.

° Context-Identified Values (X'1323') GDS variable: see page 1.7.3.1 for details of this GDS variable. It may be used to transport additional, implementation-specific information related to the request.

Table 7-4 provides an example of a Request Activation CP-MSU.

Table 7-4. Example Request Activation CP-MSU

Structure Name	ID/T	Contents
CP-MSU	1212	__
Routing and Targeting Instructions	154D	__
Name List	06	__
Cascaded Resource Name List	10	*APPL1, APPL2.* (path is APPL1 to APPL2)
Destination Application Name	50	*APPL3.* (target application program name, request delivered from APPL2)
Origin Application Name	60	*APPL7.* (request origin, instance named below)
Destination Instance Identifier	70	*APPL3A.* (named instance of target, APPL3)
Origin Instance Identifier	80	*APPL7A.* (named instance of originator, APPL7)
Correlator Control	0D	__
Application Correlator Control	01	Byte-offset 2 bit-settings: B'10000000'
Request Activation	8066	__
Reporting Level	09	__
Report Requirements	10	Required to send acceptance and failure reports
Configuration Name	07	Named configuration file at target
Execution Window Timing	0B	__

Execute Not Before Time	10	Execute not before 09:00:00, 12/31/91
Activate Command Subvector	81	__
Force Indication	10	Force(no); do not execute if target not quiesced
Change Management Activation Use	20	(last); that which was used on the last activation

1.7.3.1.3 Activation Acceptance CP-MSU

The structure of the Activation Acceptance CP-MSU is shown in Table 7-5.

Table 7-5. The Activation Acceptance CP-MSU

Structure Name	ID/T	Occurrences Within Parent
CP-MSU	1212	__
Routing and Targeting Instructions	154D	0-1
SNA Condition Report	1532	0-1
Activation Acceptance	0066	1
Date/Time	01	1
Activation Acceptance Subvector	82	1
Attempt Status	10	1
Detailed Data	98	°0

This reply contains the following:

Required Parameters

° Date/Time (X'01') MS common subvector: the date and time the reply was built

° Attempt Status (X'10') Activation Acceptance subfield: whether an attempt to activate the target will or will not be made

Optional Parameters

° Error indication if command will not be attempted (transported in an SNA Condition Report, X'1532' GDS variable)

° Detailed Data (X'98') subvector: additional, implementation-specific data included for the network operator. One or more Detailed Data Extended (X'85') subfields or one or more Detailed Data (X'82') subfields may be contained within this subvector. The format of this subvector is like that of the Detailed Data (X'98') Alert MS subvector. The only difference is that this subvector does not contain any Qualified Message Data (X'01') subfields.

° Routing and Targeting Instructions (X'154D') GDS variable: identifies a second-level application program target of the reply/report. If the originator of the reply/report is a second-level application program, then it is also identified here. In addition, this GDS variable contains an indication of how the Agent Unit-of-Work Correlator (X'1549') GDS variable is to be interpreted by the second-level application programs involved. See "The Routing and Targeting Instructions GDS Variable" in topic 1.7.3.3.1 and "MDS Message Types and Application-Level Flows" in topic 1.7.3.3.1 for more information.

Table 7-6 provides an example of an Activation Acceptance CP-MSU.

Table 7-6. Example Activation Acceptance CP-MSU

Structure Name	ID/T	Contents
CP-MSU	1212	__
Routing and Targeting Instructions	154D	__
Name List	06	__
Cascaded Resource Name List	10	*APPL2, APPL1.* (path is APPL2 to APPL1, opposite of req.)
Destination Application Name	50	*APPL7.* (target application program name, delivered from APPL1)
Origin Application Name	60	*APPL3.* (request origin, instance named below)
Destination Instance Identifier	70	*APPL7A.* (named instance of target, APPL7)
Origin Instance Identifier	80	*APPL3A.* (named instance of originator, APPL3)
Correlator Control	0D	__
Application Correlator Control	01	Byte-offset 2 bit-settings: B'00000000'
Activation Acceptance	0066	__
Date/Time	01	09:01:46 (local), 12/31/91
Activation Acceptance	82	__
Attempt Status	10	Will attempt to activate the target
Detailed Data subvector	98	__
Detailed Data subfield	82	Additional, product-specific data

1.7.3.1.4 Activation CP-MSU

The structure of the Activation CP-MSU is shown in Table 7-7.

Table 7-7. The Activation CP-MSU

Structure Name	ID/T	Occurrences Within Parent
CP-MSU	1212	__
Routing and Targeting Instructions	154D	0-1
SNA Condition Report	1532	0-1
Activation	0067	1
Date/Time	01	1
Reporting Activation Subvector	82	1
Activation Status	10	1
Activation Stage	18	0-1
Change Management Activation Use Status	20	0-1
Detailed Data	98	°0

This report contains the following:

Required Parameters

° Date/Time (X'01') MS common subvector: the date and time the reply was built

° Activation Status (X'10') Reporting Activation subfield: successful or not successful, or intermediate

Optional Parameters

° Change Management Activation Use Status (X'20') Reporting Activation subfield (present only if activation successful). Possible indications are:

- Trial and production components were used in the activation.

- Only production components were used in the activation.

- Components used in the previous activation were used in this activation.

- Change management or change management activation use are not supported.

A special condition may also be reported if "Trial and Production" was requested but only production components were found. Another reportable special condition is that which exists when a production activation has been performed as a result of an unsuccessful trial activation.

° Error indication if not successful (transported in an SNA Condition Report X'1532' GDS variable).

° Activation Stage (X'18') Reporting Activation subfield

- Initial connectivity is complete; target is at least capable of communicating.

- Target hardware is active; target application software is not yet active.

- Target application function has reached a point where a quiesce of that function is required prior to any disruptive command being executed.

- The target has been fully activated in compliance with the configuration parameters supplied (or optionally referenced) in the activation request.

° Detailed Data (X'98') subvector: additional, implementation-specific data included for the network operator. One or more Detailed Data Extended (X'85') subfields or one or more Detailed Data (X'82') subfields may be contained within this subvector. The format of this subvector is like that of the Detailed Data (X'98') Alert MS subvector. The only difference is that this subvector does not contain any Qualified Message Data (X'01') subfields.

° Routing and Targeting Instructions (X'154D') GDS variable: identifies a second-level application program target of the reply/report. If the originator of the reply/report is a second-level application program, then it is also identified here. In addition, this GDS variable contains an indication of how the Agent Unit-of-Work Correlator (X'1549') GDS variable is to be interpreted by the second-level application programs involved. See "The Routing and Targeting Instructions GDS Variable" in topic 1.7.3.3.1 and "MDS Message Types and Application-Level Flows" in topic 1.7.3.3.1 for more information.

Table 7-8 provides an example of an Activation CP-MSU.

Table 7-8. Example Activation CP-MSU

Structure Name	ID/T	Contents
CP-MSU	1212	__
Routing and Targeting Instructions	154D	__
Name List	06	__
Cascaded Resource Name List	10	*APPL2, APPL1.* (path is APPL2 to APPL1, opposite of req.)
Destination Application Name	50	*APPL7.* (target

		application program, delivered from APPL1)
Origin Application Name	60	*APPL3.* (request origin, instance named below)
Destination Instance Identifier	70	*APPL7A.* (named instance of target, APPL7)
Origin Instance Identifier	80	*APPL3A.* (named instance of originator, APPL3)
Correlator Control	0D	__
Application Correlator Control	01	Byte-offset 2 bit-settings: B'01100000'
SNA Condition Report	1532	SNA sense data code point describing failure
Activation	0067	__
Date/Time	01	09:05:01 (local), 12/31/91
Reporting Activation	82	__
Activation Status	10	Not successful
Detailed Data subvector	98	__
Detailed Data subfield	82	Additional, product-specific data

1.7.3.1.5 Request Deactivation CP-MSU

The structure of the Request Deactivation CP-MSU is shown in Table 7-9.

Table 7-9. The Request Deactivation CP-MSU

Structure Name	ID/T	Occurrences Within Parent
CP-MSU	1212	__
Routing and Targeting Instructions	154D	0-1
Context-Identified Values	1323	0-1
Request Deactivation	8071	1
Reporting Level	09	0-1
Configuration Name	07	0-1
Execution Window Timing	0B	0-1
Execute Not Before Time	10	0-1
Execute Not After Time	20	0-1
Time Interval	30	0-1
Execution Count	40	0-1
Deactivate Command Subvector	81	1
Force Indication	10	0-1

This command requests an entry point to deactivate a network resource under its control. Successful deactivation implies that all functions relevant to that resource are inactive. Deactivation accomplishes the opposite of activation.

Required Parameters

- Deactivate Command (X'81') subvector

Optional Parameters

- Reporting Level (X'09') MS common subvector

This subvector indicates whether or not acceptance replies and status reports are to be generated by the request target. If status reports are requested, then the type of status report is also specified.

- When indicating acceptance/rejection:

  Rejection The target will not execute the associated command.
  Acceptance The target will attempt to execute the associated command.

- When indicating status after acceptance:

  Success The target successfully completed execution of the command.
  Failure The target detected an error while attempting to execute the command. There is no attempt to re-execute the command.
  Intermediate The target has status about the associated command, but the command has not finished executing.

For each report, a requirement parameter is given. These parameters are:

Required The target is required to return a report of this category.
Allowed The target may, at its discretion, return a report of this category.
Not Allowed The target shall not return a report of this category.

If this subvector is not included in the request, the target sends its reply and reports according to the defaults, listed below:

- Rejection replies are required to be sent.
- Acceptance replies are allowed to be sent (but are not required).
- Success reports are required to be sent.
- Failure reports are required to be sent.
- Intermediate reports are not allowed to be sent.

| Each use of this subvector applies only to the command in which it is
| contained. The defaults are not replaced but only overridden for the
| duration of each command request/report sequence.

° Force Indication (X'10') Deactivate subfield

- Force(No): Reject command if target not quiesced (default)

  -- Option: Specify a maximum number of seconds, minutes, or hours to wait, after the command is received, before rejecting the command. The Execution Window Timing (X'0B') MS common subvector, if present in the request, specifies when to start the waiting period. If the target is not quiesced during the waiting period, the command is rejected.

- Force(Yes): Execute command

  -- Option: Specify a maximum number of seconds, minutes, or hours to wait, after the command is received, before executing the command. The Execution Window Timing (X'0B') MS common subvector, if present in the request, specifies when to start the waiting period. If the target is not quiesced during the waiting period, the command is still executed.

- If the waiting period option is not encoded, it is implied that the waiting period will be 0. The command will be rejected/executed immediately, depending on whether Force(No) or Force(Yes) was specified.

° Execution Window Timing (X'0B') MS common subvector: the earliest or latest (or both earliest and latest) time that the command may be executed.

Optionally, a Time Interval subfield and an Execution Count subfield may be included to indicate that the command is to be executed multiple times. See "Execution Window Timing" in topic 1.7.3.1 for more information.

° Configuration Name (X'07') MS common subvector: if not specified, deactivation of the entire receiving node is implied.

° Routing and Targeting Instructions (X'154D') GDS variable: identifies a second-level application program target and optionally the path to that target. If the originator of the request is a second-level

application program, then it is also identified here. In addition, this GDS variable contains an indication of how the Agent Unit of Work GDS variable is to be interpreted by the second-level application programs involved. See "The Routing and Targeting Instructions GDS Variable" in topic 1.7.3.3.1 and "MDS Message Types and Application-Level Flows" in topic 1.7.3.3.1 for more information.

° Context-Identified Values (X'1323') GDS variable: see page 1.7.3.1 for details of this GDS variable. It may be used to transport additional, implementation-specific information related to the request.

Table 7-10 provides an example of a Request Deactivation CP-MSU.

Table 7-10. Example Request Deactivation CP-MSU

Structure Name	ID/T	Contents
CP-MSU	1212	__
Routing and Targeting Instructions	154D	__
Name List	06	__
Cascaded Resource Name List	10	*APPL1.*
Destination Application Name	50	*APPL3.* (target application program name, delivered from APPL1)
Origin Application Name	60	*APPL7.* (request origin, instance named below)
Origin Instance Identifier	80	*APPL7A.* (named instance of originator, APPL7)
Correlator Control	0D	__
Application Correlator Control	01	Byte-offset 2 bit-settings: B'10000000'
Context-Identified Values	1323	Implementation-specific information associated with deactivation
Request Deactivation	8071	__
Reporting Level	09	__
Report Requirements	10	Required to send acceptance, intermediate, and failure reports
Configuration Name	07	Named configuration file at target
Execution Window Timing	0B	__
Execute Not After Time	10	Execute not after 09:00:00, 12/31/91
Deactivate Command Subvector	81	__
Force Indication	10	Force(yes); execute even if target not quiesced

1.7.3.1.6 Deactivation Acceptance CP-MSU

The structure of the Deactivation Acceptance CP-MSU is shown in Table 7-11.

Table 7-11. The Deactivation Acceptance CP-MSU

Structure Name	ID/T	Occurrences Within Parent
CP-MSU	1212	__
Routing and Targeting Instructions	154D	0-1
SNA Condition Report	1532	0-1
Deactivation Acceptance	0071	1

Date/Time	01	1
Deactivation Acceptance Subvector	82	1
Attempt Status	10	1
Detailed Data	98	°0

This reply contains the following:

Required Parameters

° Date/Time (X'01') MS common subvector: the date and time the reply was built

° Attempt Status (X'10') Deactivation Acceptance subfield: will attempt or will not attempt

Optional Parameters

° Error indication if command will not be attempted (transported in an SNA Condition Report (X'1532') GDS variable)

° Detailed Data (X'98') subvector: additional, implementation-specific data included for the network operator. One or more Detailed Data Extended (X'85') subfields or one or more Detailed Data (X'82') subfields may be contained within this subvector. The format of this subvector is like that of the Detailed Data (X'98') Alert MS subvector. The only difference is that this subvector does not contain any Qualified Message Data (X'01') subfields.

° Routing and Targeting Instructions (X'154D') GDS variable: identifies a second-level application program target of the reply/report. If the originator of the reply/report is a second-level application program, then it is also identified here. In addition, this GDS variable contains an indication of how the Agent Unit-of-Work Correlator (X'1549') GDS variable is to be interpreted by the second-level application programs involved. See "The Routing and Targeting Instructions GDS Variable" in topic 1.7.3.3.1 and "MDS Message Types and Application-Level Flows" in topic 1.7.3.3.1 for more information.

Table 7-12 provides an example of a Deactivation Acceptance CP-MSU.

Table 7-12. Example Deactivation Acceptance CP-MSU

Structure Name	ID/T	Contents
CP-MSU	1212	__
Routing and Targeting Instructions	154D	__
Name List	06	__
Cascaded Resource Name List	10	*APPL1*
Destination Application Name	50	*APPL7*. (target application program name, delivered from APPL1)
Origin Application Name	60	*APPL3*. (request origin)
Destination Instance Identifier	70	*APPL7A*. (named instance of target, APPL7)
Correlator Control	0D	__
Application Correlator Control	01	Byte-offset 2 bit-settings: B'00000000'
Deactivation Acceptance	0071	__
Date/Time	01	09:01:46 (local), 12/31/91
Deactivation Acceptance	82	__
Attempt Status	10	Will attempt

1.7.3.1.7 Deactivation CP-MSU

The structure of the Deactivation CP-MSU is shown in Table 7-13.

Table 7-13. The Deactivation CP-MSU

Structure Name	ID/T	Occurrences Within Parent
CP-MSU	1212	__
Routing and Targeting Instructions	154D	0-1
SNA Condition Report	1532	0-1
Deactivation	0072	1
Date/Time	01	1
Reporting Deactivation Subvector	82	1
Deactivation Status	10	1
Detailed Data	98	°0

This report contains the following:

Required Parameters

° Date/Time (X'01') MS common subvector: the date and time the reply was built

° Deactivation Status (X'10') subfield: successful or not successful, or intermediate

Optional Parameters

° Error indication if not successful (transported in an SNA Condition Report (X'1532') GDS variable)

° Detailed Data (X'98') subvector: additional, implementation-specific data included for the network operator. One or more Detailed Data Extended (X'85') subfields or one or more Detailed Data (X'82') subfields may be contained within this subvector. The format of this subvector is like that of the Detailed Data (X'98') Alert MS subvector. The only difference is that this subvector does not contain any Qualified Message Data (X'01') subfields.

° Routing and Targeting Instructions (X'154D') GDS variable: identifies a second-level application program target of the reply/report. If the originator of the reply/report is a second-level application program, then it is also identified here. In addition, this GDS variable contains an indication of how the Agent Unit-of-Work Correlator (X'1549') GDS variable is to be interpreted by the second-level application programs involved. See "The Routing and Targeting Instructions GDS Variable" in topic 1.7.3.3.1 and "MDS Message Types and Application-Level Flows" in topic 1.7.3.3.1 for more information.

Table 7-14 provides an example of a Deactivation CP-MSU.

Table 7-14. Example Deactivation CP-MSU

Structure Name	ID/T	Contents
CP-MSU	1212	__
Routing and Targeting Instructions	154D	__
Name List	06	__
Cascaded Resource Name List	10	*APPL1.*
Destination Application Name	50	*APPL7.* (target application program name, delivered from APPL1)

Origin Application Name	60	*APPL3.* (request origin, instance named below)
Destination Instance Identifier	70	*APPL7A.* (named instance of target, APPL7)
Correlator Control	0D	__
Application Correlator Control	01	Byte-offset 2 bit-settings: B'01100000'
Deactivation	0072	__
Date/Time	01	09:05:01 (local), 12/31/91
Reporting Deactivation	82	__
Deactivation Status	10	Successful

1.7.3.1.8 Set Clock CP-MSU

The structure of the Set Clock CP-MSU is shown in Table 7-15.

Table 7-15. The Set Clock CP-MSU

Structure Name	ID/T	Occurrences Within Parent
CP-MSU	1212	__
Routing and Targeting Instructions	154D	0-1
Context-Identified Values	1323	0-1
Set Clock	8075	1
Reporting Level	09	0-1
Execution Window Timing	0B	0-1
Execute Not Before Time	10	0-1
Execute Not After Time	20	0-1
Time Interval	30	0-1
Execution Count	40	0-1
Set Clock Command Subvector	81	1
Force Indication	10	0-1
Clock Name	15	0-1
Timing Source Name	20	0-1
Date/Time to Set	30	0-1
GMT Offset	40	0-1
Increment/Decrement Interval	50	0-1

This command may be used to accomplish the following:

- Set an existing clock at the target:
  - From a named timing source
  - From date and time values encoded in the Set Clock command

  The above methods are mutually exclusive.

- Increment one of the following:
  - The existing clock time at the target
  - A timing source time value, which, after being modified by this command, may be used later to set a clock. The timing source itself is not modified; rather, its time value is used as a base upon which the modification is made.

- The Greenwich Mean Time offset at the target

- Both the clock time and the Greenwich Mean Time offset (by the same value)

Required Parameters

° Set Clock Command (X'81') subvector

Optional Parameters

° Reporting Level (X'09') MS common subvector

| This subvector indicates whether or not acceptance reports and status reports are to be generated by the request target. If status reports are requested, then the type of status report is also specified.

- When indicating acceptance/rejection:

Rejection The target will not execute the associated command.
Acceptance The target will attempt to execute the associated command.

- When indicating status after acceptance:

Success The target successfully completed execution of the command.
Failure The target detected an error while attempting to execute the command. There is no attempt to re-execute the command.
Intermediate The target has status about the associated command, but the command has not finished executing.

For each report, a requirement parameter is given. These parameters are:

Required The target is required to return a report of this category.
Allowed The target may, at its discretion, return a report of this category.
Not Allowed The target shall not return a report of this category.

If this subvector is not included in the request, the target sends its reply and reports according to the defaults, listed below:

- Rejection replies are required to be sent.
- Acceptance replies are allowed to be sent (but are not required).
- Success reports are required to be sent.
- Failure reports are required to be sent.
- Intermediate reports are not allowed to be sent.

| Each use of this subvector applies only to the command in which it is
| contained. The defaults are not replaced but only overridden for the
| duration of each command request/report sequence.

° Force Indication (X'10') Set Clock subfield

- Force(No): Reject command if target not quiesced (default)

  -- Option: Specify a maximum number of seconds, minutes, or hours to wait, after the command is received, before rejecting the command. The Execution Window Timing (X'0B') MS common subvector, if present in the request, specifies when to start the waiting period. If the target is not quiesced during the waiting period, the command is rejected.

- Force(Yes): Execute command

  -- Option: Specify a maximum number of seconds, minutes, or hours to wait, after the command is received, before executing the command. The Execution Window Timing (X'0B') MS common subvector, if present in the request, specifies when to start the waiting period. If the target is not quiesced during the waiting period, the command is still executed.

- If the waiting period option is not encoded, it is implied that the waiting period will be 0. The command will be rejected/executed immediately, depending on whether Force(No) or

Force(Yes) was specified.

- Clock Name (X'15') subfield: an implementation-defined name of a clock to be set. If not present, a single (only) target clock is implied. This subfield can also be used to name a Timing Source to be set. Like a clock, a timing source may be set from another timing source (named in the Timing Source Name subfield) or from the time specified in the Date/Time to Set subfield.

- Timing Source Name (X'20') subfield: an implementation-defined name of a timing source to use to set a clock or another timing source. Not present if Date/Time to Set (X'30') or Greenwich Mean Time Offset (X'40') subfields are present.

- Date/Time to Set (X'30') subfield: the year, month, day, hour, and minute that the target clock or timing source is to be set to. An indication of whether the time is local time or Greenwich Mean Time is also given.

- Greenwich Mean Time Offset (X'40') subfield: an adjustment indicator, specifying one of the following:

  - Adjustment to be added to local time to obtain Greenwich Mean Time, or subtracted from Greenwich Mean Time to obtain local time.

  - Adjustment to be subtracted from local time to obtain Greenwich Mean Time, or added to Greenwich Mean Time to obtain local time.

- Increment/Decrement Interval (X'50') subfield: number of hours, minutes, seconds to increment/decrement the current time in Clock Name (X'15') or Timing Source Name (X'20') subfield. An indication is given to set either the clock (or timing source) named in the Clock Name subfield, or the GMT Offset value, or both values. Not present if Date/Time to Set (X'30') or Greenwich Mean Time Offset (X'40') subfield is present.

- Execution Window Timing (X'0B') MS common subvector: the earliest or latest (or both earliest and latest) time that the command may be executed.

  Optionally, a Time Interval subfield and an Execution Count subfield may be included to indicate that the command is to be executed multiple times. See "Execution Window Timing" in topic 1.7.3.1 for more information.

- Routing and Targeting Instructions (X'154D') GDS variable: identifies a second-level application program target and optionally the path to that target. If the originator of the request is a second-level application program, then it is also identified here. In addition, this GDS variable contains an indication of how the Agent Unit of Work GDS variable is to be interpreted by the second-level application programs involved. See "The Routing and Targeting Instructions GDS Variable" in topic 1.7.3.3.1 and "MDS Message Types and Application-Level Flows" in topic 1.7.3.3.1 for more information.

- Context-Identified Values (X'1323') GDS variable: see page 1.7.3.1 for details of this GDS variable. It may be used to transport additional, implementation-specific information related to the request.

Table 7-16 provides an example of a Set Clock CP-MSU.

Table 7-16. Example Set Clock CP-MSU

Structure Name	ID/T	Contents
CP-MSU	1212	__
Routing and Targeting Instructions	154D	__
Name List	06	__
Cascaded Resource Name List	10	*APPL2*.
Destination Application Name	50	*APPL3*. (target application program name, delivered from APPL2)
Origin Application Name	60	*APPL7*. (request origin, instance named below)
Destination Instance Identifier	70	*APPL3B*. (named instance of target, APPL3)

Correlator Control	0D	__
Application Correlator Control	01	Byte-offset 2 bit-settings: B'10000000'
Set Clock	8075	__
Reporting Level	09	__
Report Requirements	10	Required to send acceptance and failure reports
Execution Window Timing	0B	__
Execute Not Before Time	10	Execute not before 09:00:00, 12/31/91
Execute Not After Time	20	Execute not after 09:05:00, 12/31/91
Time Interval	30	One-day interval
Execution Count	40	Execute 31 times
Set Clock Command Subvector	81	__
Force Indication	10	Force(yes); execute even if target not quiesced
Clock Name	15	Name of clock to be set
Timing Source Name	20	Name of radio signal or other source to set clock from

---

1.7.3.1.9 Set Clock Report CP-MSU

The structure of the Set Clock Report CP-MSU is shown in Table 7-17.

Table 7-17. The Set Clock Report CP-MSU

Structure Name	ID/T	Occurrences Within Parent
CP-MSU	1212	__
Routing and Targeting Instructions	154D	0-1
SNA Condition Report	1532	0-1
Set Clock Report	0075	1
Date/Time	01	0-1
Reporting Clock Set Subvector	82	1
Set Clock Status	10	1
Detailed Data	98	°0

This report contains the following:

Required Parameters

° Set Clock Status (X'10') subfield: accepted or not accepted, successful or not successful, or intermediate

Optional Parameters

° Date/Time (X'01') MS common subvector: the date and time the reply was built

° Error indication if not successful or not accepted (transported in an SNA Condition Report (X'1532') GDS variable)

° Detailed Data (X'98') subvector: additional, implementation-specific data included for the network operator. One or more Detailed Data Extended (X'85') subfields or one or more Detailed Data (X'82') subfields may be contained within this subvector. The format of this

subvector is like that of the Detailed Data (X'98') Alert MS subvector. The only difference is that this subvector does not contain any Qualified Message Data (X'01') subfields.

° Routing and Targeting Instructions (X'154D') GDS variable: identifies a second-level application program target of the reply/report. If the originator of the reply/report is a second-level application program, then it is also identified here. In addition, this GDS variable contains an indication of how the Agent Unit-of-Work Correlator (X'1549') GDS variable is to be interpreted by the second-level application programs involved. See "The Routing and Targeting Instructions GDS Variable" in topic 1.7.3.3.1 and "MDS Message Types and Application-Level Flows" in topic 1.7.3.3.1 for more information.

Table 7-18 provides an example of a Set Clock Report CP-MSU reporting acceptance of a Set Clock CP-MSU.

Table 7-18. Example Set Clock Report CP-MSU (Acceptance)

Structure Name	ID/T	Contents
CP-MSU	1212	__
Routing and Targeting Instructions	154D	__
Name List	06	__
Cascaded Resource Name List	10	*APPL2.*
Destination Application Name	50	*APPL7.*
Origin Application Name	60	*APPL3.* (reply origin, instance named below)
Origin Instance Identifier	60	*APPL3B.* (instance of reply origin named above)
Correlator Control	0D	__
Application Correlator Control	01	Byte-offset 2 bit-settings: B'00000000'
Set Clock Report	0075	__
Date/Time	01	09:05:01 (local), 12/31/91
Reporting Clock Set	82	__
Set Clock Status	10	Accepted

Table 7-19 provides an example of a Set Clock Report CP-MSU reporting the successful execution of a Set Clock CP-MSU.

Table 7-19. Example Set Clock Report CP-MSU (Success)

Structure Name	ID/T	Contents
CP-MSU	1212	__
Routing and Targeting Instructions	154D	__
Name List	06	__
Cascaded Resource Name List	10	*APPL2.*
Destination Application Name	50	*APPL7.*
Origin Application Name	60	*APPL3.* (reply origin, instance named below)
Origin Instance Identifier	60	*APPL3B.* (instance of reply origin named above)
Correlator Control	0D	__
Application Correlator Control	01	Byte-offset 2 bit-settings: B'01100000'
Set Clock Report	0075	__
Date/Time	01	09:06:01 (local), 12/31/91

Reporting Clock Set	82	__
Set Clock Status	10	Successful

1.7.3.1.10 Request Cancelation CP-MSU

The structure of the Request Cancelation CP-MSU is shown in Table 7-20.

Table 7-20. The Request Cancelation CP-MSU

Structure Name	ID/T	Occurrences Within Parent
CP-MSU	1212	__
Routing and Targeting Instructions	154D	0-1
Context-Identified Values	1323	0-1
Request Cancelation	8076	1
Reporting Level	09	0-1
Cancel Command Subvector	81	1
Cancel Specific Tasks	10	0-2
Cancel All	20	0-2
Cancel All Deferred	30	0-2
Agent Unit-of-Work Correlator	1549	°0

This command requests that the target stop the execution of requests sent/received earlier. Specific requests or all requests meeting certain criteria may be canceled. The target is responsible for attempting this in a manner that does not disrupt the processing of requests received, but not affected by the Cancel request. The requests that are to be canceled may have been completely processed by the time the Request Cancelation arrives at the target. The target, however, is not required to ensure that the effects of those completed requests are reversed. The target is free, however, to reject the request if the effect of cancelation is undesirable.

EP_OPERATIONS_MGMT, once it receives a Cancel request, performs one of the following actions:

° Processes the request itself if it is the request target

° Passes the request (entire MDS-MU) to the second-level application program named in the Routing and Targeting Instructions (RTI)

° Passes the CP-MSU containing the request to EP_CHANGE_MGMT (in the same node) over MS application protocol boundary BB (see "Protocol Boundary BB--Cancelation From Operations Management" in topic 2.9.1.18). This particular action occurs only when the Cancel request specifies that requests sent to EP_CHANGE_MGMT are to be canceled. The program supervisor component of the change management entry point then begins the cancelation process described in "EP_CHANGE_MGMT Optional Subset 7 (Cancel Command Support)" in topic 2.10.4.11.

Before EP_OPERATIONS_MGMT forwards a Cancel request to EP_CHANGE_MGMT, the MDS and RTI routing information is saved. After EP_CHANGE_MGMT processes the request, it builds at least one report, and possibly more:

- A Cancelation CP-MSU is always built and sent to EP_OPERATIONS_MGMT over MS protocol boundary CC (see "Protocol Boundary CC--Reporting Cancelation to Operations Management" in topic 2.9.1.19). EP_OPERATIONS_MGMT then encloses the CP-MSU within an MDS-MU (using the MDS and RTI routing information saved from the Request Cancelation) and sends the MDS-MU to the Request Cancelation originator.

- One or more reports are built by EP_CHANGE_MGMT and returned to the change management focal point for each request that was affected by the cancelation process. Each report carries a sense data value that indicates that the related request was canceled due to operator intervention.

  If the cancelation process found no requests to be canceled, then EP_CHANGE_MGMT does not send any reports to the change management

focal point.

Both the first and third options above may be executed if the Cancel request specifies that requests sent to both operations management and change management entry points are to be canceled.

The sender of the Cancel request may be either the operations management focal point, or a second-level application program (involved in the network operator function) that is served by the operations management focal point. Furthermore, particular instances of these second-level application programs may send Cancel requests.

Required Parameters:

One of three mutually exclusive subfields:

° Cancel Specific Tasks (X'10')

- Cancel requests by matching the one or more AUOWCs (which must be present in the CP-MSU in this case) with the correlators of the requests to be canceled.

- Cancel the request currently executing under control of the agent identified in the Routing and Targeting Instructions GDS variable of the Cancel request.

- Cancel the currently running request sent by the named application program. The named application may be the either the operations management or change management focal-point application program, or it may be some other non-focal-point program.

- Cancel the currently running request received by the named application program. The named application program may be either EP_OPERATIONS_MGMT, EP_CHANGE_MGMT, or it may be some other non-SNA/MS-registered program name.

° Cancel All (X'20')

Cancel all requests (both those that are currently active and those that have been accepted but have not yet executed) characterized as follows:

- All requests received by the receiver of the Cancel request

- All requests from the sender of the Cancel request. If an Origin Instance Identifier (X'80') subfield of the Name List (X'06') subvector is present in the request, it identifies the sender. Information from MDS concerning the origin is not used in this attempt to match.

- All requests from the sender of the Cancel request. Even if an Origin Instance Identifier subfield is present in the request, just the Origin Application Name (X'60') subfield is used to identify the sender. Information from MDS concerning the origin is not used in this attempt to match.

- All requests from the sender of the Cancel request. If an Origin Instance Identifier (X'80') subfield of the Name List (X'06') subvector is present in the request, it identifies the sender. The MDS header NETID, NAU name, and MS application program name of a candidate request must match those same parameters of the sender of the Cancel request.

- All requests from the sender of the Cancel request. Even if an Origin Instance Identifier subfield is present in the request, just the Origin Application Name (X'60') subfield is used to identify the sender. The MDS header NETID, NAU name, and MS application program name of a candidate request must match those same parameters of the sender of the Cancel request.

- All requests received by a named application program at the target. The named application may be either EP_OPERATIONS_MGMT, EP_CHANGE_MGMT, or it may be some other non-SNA/MS-registered name.

- All requests sent by a named application program at the sender. The named application may be either the the operations management or change management focal-point application program, or it may be some other non-focal-point program.

- All requests received by both EP_OPERATIONS_MGMT and

EP_CHANGE_MGMT

- All requests sent by both the change management and operations management focal-point application programs

- All requests received by any application program at the target. This is limited to all requests received by all of the following:

  -- EP_OPERATIONS_MGMT

  -- All second-level application programs served by EP_OPERATIONS_MGMT

  -- EP_CHANGE_MGMT (in the same node)

° Cancel All Deferred (X'30')

Cancel all of those requests that are pending execution and are characterized as follows:

- All deferred requests received by the receiver of the Cancel request

- All deferred requests from the sender of the Cancel request. If an Origin Instance Identifier (X'80') subfield of the Name List (X'06') subvector is present in the request, it identifies the sender. Information from MDS concerning the origin is not used in this attempt to match.

- All deferred requests from the sender of the Cancel request. Even if an Origin Instance Identifier subfield is present in the request, only the Origin Application Name (X'60') subfield is used to identify the sender. Information from MDS concerning the origin is not used in this attempt to match.

- All deferred requests from the sender of the Cancel request. If an Origin Instance Identifier (X'80') subfield of the Name List (X'06') subvector is present in the request, it identifies the sender. The MDS header NETID, NAU name, and MS application program name of a candidate request must match those same parameters of the sender of the Cancel request.

- All deferred requests from the sender of the Cancel request. Even if an Origin Instance Identifier subfield is present in the request, only the Origin Application Name (X'60') subfield is used to identify the sender. The MDS header NETID, NAU name, and MS application program name of a candidate request must match those same parameters of the sender of the Cancel request.

- All deferred requests received by a named application program at the target. The named application may be either EP_OPERATIONS_MGMT, EP_CHANGE_MGMT, or it may be some other non-SNA/MS-registered program name.

- All deferred requests sent by a named application program at the sender The named application may be either the the operations management or change management focal-point application program, or it may be some other non-focal-point program.

- All deferred requests received by both EP_OPERATIONS_MGMT and EP_CHANGE_MGMT

- All deferred requests sent by both the change management and operations management focal-point application programs

- All deferred requests received by any application program at the target. This is limited to all deferred requests received by all of the following:

  -- EP_OPERATIONS_MGMT

  -- All second-level application programs served by EP_OPERATIONS_MGMT

  -- EP_CHANGE_MGMT (in the same node)

Optional Parameters

° Reporting Level (X'09') MS common subvector

This subvector indicates whether or not acceptance reports and status reports are to be generated by the request target. If status reports are requested, then the type of status report is also specified.

- When indicating acceptance/rejection:

  Rejection The target will not execute the associated command.
  Acceptance The target will attempt to execute the associated command.

- When indicating status after acceptance:

  Success The target successfully completed execution of the command.
  Failure The target detected an error while attempting to execute the command. There is no attempt to re-execute the command.
  Intermediate The target has status about the associated command, but the command has not finished executing.

For each report, a requirement parameter is given. These parameters are:

Required The target is required to return a report of this category.
Allowed The target may, at its discretion, return a report of this category.
Not Allowed The target shall not return a report of this category.

If this subvector is not included in the request, the target sends its reply and reports according to the defaults, listed below:

- Rejection replies are required to be sent.
- Acceptance replies are allowed to be sent (but are not required).
- Success reports are required to be sent.
- Failure reports are required to be sent.
- Intermediate reports are not allowed to be sent.

Each use of this subvector applies only to the command in which it is contained. The defaults are not replaced but only overridden for the duration of each command request/report sequence.

° Routing and Targeting Instructions (X'154D') GDS variable: identifies a second-level application program target and optionally the path to that target. If the originator of the request is a second-level application program, then it is also identified here. In addition, this GDS variable contains an indication of how the Agent Unit of Work GDS variable is to be interpreted by the second-level application programs involved. See "The Routing and Targeting Instructions GDS Variable" in topic 1.7.3.3.1 and "MDS Message Types and Application-Level Flows" in topic 1.7.3.3.1 for more information. Within each of the Cancel command subfields, the presence of the RTI may be required for second-level application program identification.

° Context-Identified Values (X'1323') GDS variable: see page 1.7.3.1 for details of this GDS variable. It may be used to transport additional, implementation-specific information related to the request.

° Agent Unit-of-Work Correlator (X'1549') GDS variable:

See Table 2-4 in topic 1.2.3.2.2 for a description of the Agent Unit of Work. MDS Requests and replies are related by using the Agent Unit-of-Work Correlator (X'1549') GDS variable (AUOWC) present in the MDS header.

Table 7-21 provides an example of a Request Cancelation CP-MSU.

Table 7-21. Example Request Cancelation CP-MSU

Structure Name	ID/T	Contents
CP-MSU	1212	__
Routing and Targeting Instructions	154D	__
Name List	06	__
Cascaded Resource Name List	10	*APPL2.*
Destination Application Name	50	*APPL3.* (target application program name,

		delivered from APPL2)
Origin Application Name	60	*APPL7.* (request origin, instance named below)
Destination Instance Identifier	70	*APPL3B.* (named instance of target, APPL3)
Correlator Control	0D	__
Application Correlator Control	01	Byte-offset 2 bit-settings: B'10000000'
Request Cancelation	8076	__
Reporting Level	09	__
Report Requirements	10	Required to send acceptance and failure reports
Cancel Command Subvector	81	__
Cancel Specific Tasks	10	Cancel request identified by AUOWC in this CP-MSU (below)
Agent Unit Of Work Correlator	1549	Identifier of request to be canceled

---

1.7.3.1.11 Cancelation CP-MSU

The structure of the Cancelation CP-MSU is shown in Table 7-22.

Table 7-22. The Cancelation CP-MSU

Structure Name	ID/T	Occurrences Within Parent
CP-MSU	1212	__
Routing and Targeting Instructions	154D	0-1
SNA Condition Report	1532	0-1
Cancelation	0076	1
Date/Time	01	0-1
Reporting Cancelation Subvector	82	1
Cancelation Status	10	1
Detailed Data	98	°0

This report contains the following:

Required Parameters

° Cancelation Status (X'10') subfield: accepted or not accepted, successful or not successful, or intermediate

Optional Parameters

° Date/Time (X'01') MS common subvector: the date and time the reply was built

° Error indication if not successful or not accepted (transported in an SNA Condition Report (X'1532') GDS variable)

° Detailed Data (X'98') subvector: additional, implementation-specific data included for the network operator. One or more Detailed Data Extended (X'85') subfields or one or more Detailed Data (X'82') subfields may be contained within this subvector. The format of this subvector is like that of the Detailed Data (X'98') Alert MS subvector. The only difference is that this subvector does not contain any Qualified Message Data (X'01') subfields.

- Routing and Targeting Instructions (X'154D') GDS variable: identifies a second-level application program target of the reply/report. If the originator of the reply/report is a second-level application program, then it is also identified here. In addition, this GDS variable contains an indication of how the Agent Unit-of-Work Correlator (X'1549') GDS variable is to be interpreted by the second-level application programs involved. See "The Routing and Targeting Instructions GDS Variable" in topic 1.7.3.3.1 and "MDS Message Types and Application-Level Flows" in topic 1.7.3.3.1 for more information.

Table 7-23 provides an example of a Cancelation CP-MSU. reporting acceptance of a Request Cancelation CP-MSU.

Table 7-23. Example Cancelation CP-MSU (Acceptance)

Structure Name	ID/T	Contents
CP-MSU	1212	__
Routing and Targeting Instructions	154D	__
Name List	06	__
Cascaded Resource Name List	10	*APPL2.*
Destination Application Name	50	*APPL7.*
Origin Application Name	60	*APPL3.* (reply origin, instance named below)
Origin Instance Identifier	60	*APPL3B.* (instance of reply origin named above)
Correlator Control	0D	__
Application Correlator Control	01	Byte-offset 2 bit-settings: B'00000000'
Cancelation	0076	__
Date/Time	01	09:05:01 (local), 12/31/91
Reporting Cancelation	82	__
Cancelation Status	10	Accepted

Table 7-24 provides an example of a Cancelation CP-MSU reporting the successful execution of a Request Cancelation CP-MSU.

Table 7-24. Example Cancelation CP-MSU (Success)

Structure Name	ID/T	Contents
CP-MSU	1212	__
Routing and Targeting Instructions	154D	__
Name List	06	__
Cascaded Resource Name List	10	*APPL2.*
Destination Application Name	50	*APPL7.*
Origin Application Name	60	*APPL3.* (reply origin, instance named below)
Origin Instance Identifier	60	*APPL3B.* (instance of reply origin named above)
Correlator Control	0D	__
Application Correlator Control	01	Byte-offset 2 bit-settings: B'01100000'
Cancelation	0076	__
Date/Time	01	09:06:01 (local), 12/31/91
Reporting Cancelation	82	__
Cancelation Status	10	Successful

1.7.3.1.12 Operate CP-MSU

The structure of the Operate CP-MSU is shown in Table 7-25.

Table 7-25. The Operate CP-MSU

Structure Name	ID/T	Occurrences Within Parent
CP-MSU	1212	__
Routing and Targeting Instructions	154D	0-1
Operate	8070	1
Reporting Level	09	0-1
Configuration Name	07	0-1
Default Character Set ID	32	0-1
Execution Window Timing	0B	0-1
Execute Not Before Time	10	0-1
Execute Not After Time	20	0-1
Time Interval	30	0-1
Execution Count	40	0-1
Operate Command Subvector	81	1
Force Indication	10	0-1
Change Mgmt Activation Use	20	0-1
Text Command	1320	0-1
Self-Defining Text Message	31	°1
Partial Format Processing Method	1321	0-1
Transparent Coded Datastream	1309	0-1
Display Datastream	1322	0-1
Context-Identified Values	1323	0-2

A new set of SNA/MS formats is described in this architecture for transport of data unique to entry point implementations. These are *partial* SNA/MS formats in that SNA/MS specifies only header information to help organize the structure of the data, and does not attempt to interpret its meaning or its detailed syntax. Refer to *SNA Formats* while reading this section.

Required Parameters:

The following GDS variables (X'1320', X'1309', and X'1322') are mutually exclusive within the CP-MSU:

° Text Command (X'1320') GDS variable

- Includes the Self-Defining Text Message (X'31'), which contains both a text command and the identification of the coded graphics character set and the national language of the text

° Transparent Coded Datastream (X'1309') parameter major vector

- Format type, identified by the Partial Format Processing Method (X'1321') GDS variable, which indicates what format type the receiving entry point is expected to support (up to 16 characters in coded graphics character set 01134-00500)

- Datastream (meaning undefined by MS)

° Display Datastream (X'1322') GDS variable

- Display datastream type, identified by the Partial Format Processing Method (X'1321') GDS variable (up to 16 characters in coded graphics character set 01134-00500)

- Display datastream (meaning undefined by MS)

The following GDS variable (X'1323') may be included twice within the CP-MSU to allow for the specification of two distinct sets of implementation-specific parameters. However, If either of the three preceding mutually exclusive GDS variables is present, only one X'1323' GDS variable may be present.

| ° Context-Identified Values (X'1323') GDS variable: see page 1.7.3.1 for details of this GDS variable. It may be used to transport additional, implementation-specific information related to the request.

Optional Parameters

° Reporting Level (X'09') MS common subvector

| This subvector indicates whether or not acceptance reports and status reports are to be generated by the request target. If status reports are requested, then the type of status report is also specified.

- When indicating acceptance/rejection:

Rejection	The target will not execute the associated command
Acceptance	The target will attempt to execute the associated command

- When indicating status after acceptance, or ancillary information:

Success	The target successfully completed execution of the command.
Failure	The target detected an error while attempting to execute the command. There is no attempt to re-execute the command.
Intermediate	The target has status about the associated command, but the command has not finished executing.
Ancillary	The execution having completed, the target is sending information as a result of having entered some state caused by that execution. Only the Operate may solicit a report classification of "ancillary." Consequently, only the Operate Report is allowed to be an ancillary report.

For each report, a requirement parameter is given. These parameters are:

Required	The target is required to return a report of this category.
Allowed	The target may, at its discretion, return a report of this category.
Not Allowed	The target shall not return a report of this category.

If this subvector is not included in the request, the target sends its reply and reports according to the defaults, listed below:

- Rejection replies are required to be sent.
- Acceptance replies are allowed to be sent (but are not required).
- Success reports are required to be sent.
- Failure reports are required to be sent.
- Intermediate reports are not allowed to be sent.
- Ancillary reports are not allowed to be sent.

| Each use of this subvector applies only to the command in which it is contained. The defaults are not replaced but only overridden for the duration of each command request/report sequence.

° Execution Window Timing (X'0B') MS common subvector: the earliest or latest (or both earliest and latest) time that the command may be executed.

Optionally, a Time Interval subfield and an Execution Count subfield may be included to indicate that the command is to be executed multiple times. See "Execution Window Timing" in topic 1.7.3.1 for more information.

° Default Character Set ID (X'32') MS common subvector: the default coded character set to be used in character-string interpretation whenever a character set is not explicitly specified. The national language in which the character string is written is also identified.

° Force Indication (X'10') Operate subfield

- Force(No): Reject command if target not quiesced (default)

  -- Option: Specify a maximum number of seconds, minutes, or hours to wait, after the command is received, before rejecting the command. The Execution Window Timing (X'0B') MS common subvector, if present in the request, specifies when to start the waiting period. If the target is not quiesced during the waiting period, the command is rejected.

- Force(Yes): Execute command

  -- Option: Specify a maximum number of seconds, minutes, or hours to wait, after the command is received, before executing the command. The Execution Window Timing (X'0B') MS common subvector, if present in the request, specifies when to start the waiting period. If the target is not quiesced during the waiting period, the command is still executed.

- If the waiting period option is not encoded, it is implied that the waiting period will be 0. The command will be rejected/executed immediately, depending on whether Force(No) or Force(Yes) was specified.

° Change Management Activation Use (X'20') Activate subfield

Both change management and operations management use this subfield for the same purpose: to control which versions of entry-point software components are used in an activation. This subfield occurs here because an implementation may choose to do activation via the Operate command.

This subfield causes the entry point to base its activation on *trial* and/or *production* versions of software components. Alternatively, the network operator may specify that the versions used in the previous (last) activation are to be used in the present activation.

Changes that cannot be tested fully or those having a strong potential to disrupt entry-point processing are best installed on trial. After activation and a period of successful operation, the activation may be issued again, specifying that production versions are to be used.

Note: This parameter is conditionally present. It is always present unless the requester knows by some means that the target has not implemented change management or change management activation use (and is therefore incapable of interpreting the meaning of this subfield). If this subfield is present and the target of the request does *not* support change management or this subfield, then the target of the request sends an Activation Acceptance CP-MSU (containing a SNACR with details of the error) to the request originator.

° Configuration Name (X'07') MS common subvector: encoded in character graphics ID 01134-00500. Even though an implementation may choose to send a product-specific activation command in this GDS variable (instead of using the Request Activation CP-MSU), the Configuration Name subvector may be used to name a resource or file of resources that are affected by the activation.

If not specified, then it is implied that the command is addressed to the entire receiving node.

° Routing and Targeting Instructions (X'154D') GDS variable: identifies a second-level application program target and optionally the path to that target. If the originator of the request is a second-level application program, then it is also identified here. In addition, this GDS variable contains an indication of how the Agent Unit of Work GDS variable is to be interpreted by the second-level application programs involved. See "The Routing and Targeting Instructions GDS Variable" in topic 1.7.3.3.1 and "MDS Message Types and Application-Level Flows" in topic 1.7.3.3.1 for more information.

Table 7-26 provides an example of an Operate CP-MSU.

Table 7-26. Example Operate CP-MSU

Structure Name	ID/T	Contents
CP-MSU	1212	__
Routing and Targeting Instructions	154D	__

Name List	06	__
Cascaded Resource Name List	10	*APPL2.*
Destination Application Name	50	*APPL3.* (target application program name, delivered from APPL2)
Origin Application Name	60	*APPL7.* (request origin, instance named below)
Destination Instance Identifier	70	*APPL3B.* (named instance of target, APPL3)
Correlator Control	0D	__
Application Correlator Control	01	Byte-offset 2 bit-settings: B'10000000'
Operate	8070	__
Reporting Level	09	__
Report Requirements	10	Required to send acceptance and failure reports
Default Character Set ID	32	Unless otherwise specified, use 01134-00500
Execution Window Timing	0B	__
Execute Not Before Time	10	Execute not before 09:00:00, 12/31/91
Execute Not After Time	20	Execute not after 09:05:00, 12/31/91
Time Interval	30	One-day interval
Operate Command Subvector	81	__
Force Indication	10	Force (no); do not execute if target not quiesced
Text Command	1320	__
Self-Defining Text Message	31	Implementation-specific command (01134-00500) for each X'32' subvector

1.7.3.1.13 Operate Report CP-MSU

The structure of the Operate Report CP-MSU is shown in Table 7-27.

Table 7-27. The Operate Report CP-MSU

Structure Name	ID/T	Occurrences Within Parent
CP-MSU	1212	__
Routing and Targeting Instructions	154D	0-1
SNA Condition Report	1532	0-1
Operate Report	0070	1
Date/Time	01	0-1
Reporting Operation Subvector	82	1
Operation Status	10	0-1
Change Mgmt Act. Use Status	20	0-1
Detailed Data	98	°0
Default Character Set ID	32	0-1
Text Data	1300	0-1
Self-Defining Text Message	31	°0
Qualified Message	0A	°0
Partial Format Processing Method	1321	0-1
Transparent Coded Datastream	1309	0-1

Display Datastream	1322	0-1
Context-Identified Values	1323	0-1

This report contains the following:

An unsolicited Operate Report can also be used by the entry-point operator to send a message to the focal-point operator. Such a report would be a type of ancillary report (see "Operate Report" in topic 2.10.6.5.5).

Required Parameters:

The following GDS variables (X'1300', X'1309', X'1322', and X'1323') are mutually exclusive within the CP-MSU: Any of them can be specified in an Operate Report, regardless of which was specified in a request.

- Text Data (X'1300') parameter major vector containing either:
  - One or more Self-Defining Text Message MS common subvectors containing a text string, or
  - One or more Qualified Message MS common subvectors containing the message ID of a text message stored at the receiver and, optionally, one or more items of replacement text.
- Transparent Coded Datastream (X'1309') parameter major vector
  - Format type, identified by the Partial Format Processing Method (X'1321') GDS variable, which indicates what format type the receiving entry point is expected to support (up to 16 characters in coded graphics character set 01134-00500)
  - Datastream (meaning undefined by SNA/MS)
- Display Datastream (X'1322') GDS variable,
  - Display datastream type, identified by the Partial Format Processing Method (X'1321') GDS variable (up to 16 characters in coded graphics character set 01134-00500)
  - Display datastream (meaning undefined by SNA/MS)
- Context-Identified Values (X'1323') GDS variable, the same structure as shown in the Operate request

Optional Parameters

- Operation Status (X'10') Reporting Operation subfield: indicates whether the request is accepted or not accepted, successful or not successful, or is an intermediate or ancillary report. If this subfield is not included, the report is assumed to be an ancillary report.
- Date/Time (X'01') MS common subvector: the date and time the reply was built
- Error indication (transported in an SNA Condition Report (X'1532') GDS variable)
- Detailed Data (X'98') subvector: additional, implementation-specific data included for the network operator. One or more Detailed Data Extended (X'85') subfields or one or more Detailed Data (X'82') subfields may be contained within this subvector. The format of this subvector is like that of the Detailed Data (X'98') Alert MS subvector. The only difference is that this subvector does not contain any Qualified Message Data (X'01') subfields.
- Default Character Set ID (X'32') MS common subvector: establishes a default code page, character set, and national language ID if one is not specified elsewhere in the the report.
- Change Management Activation Use Status (X'20') Reporting Activation subfield (present only if activation successful). Possible indications are:

- Trial and production components were used in the activation.

- Only production components were used in the activation.

- Components used in the previous activation were used in this activation.

- Change management or change management activation use are not supported.

A special condition may also be reported if "Trial and Production" was requested but only production components were found. Another reportable special condition is that which exists when a production activation has been performed as a result of an unsuccessful trial activation.

° Routing and Targeting Instructions (X'154D') GDS variable: identifies a second-level application program target of the reply/report. If the originator of the reply/report is a second-level application program, then it is also identified here. In addition, this GDS variable contains an indication of how the Agent Unit-of-Work Correlator (X'1549') GDS variable is to be interpreted by the second-level application programs involved. See "The Routing and Targeting Instructions GDS Variable" in topic 1.7.3.3.1 and "MDS Message Types and Application-Level Flows" in topic 1.7.3.3.1 for more information.

Table 7-28 provides an example of an Operate Report CP-MSU reporting acceptance of an Operate CP-MSU.

Table 7-28. Example Operate Report CP-MSU (Acceptance)

Structure Name	ID/T	Contents
CP-MSU	1212	__
Routing and Targeting Instructions	154D	__
Name List	06	__
Cascaded Resource Name List	10	*APPL2.*
Destination Application Name	50	*APPL7.*
Origin Application Name	60	*APPL3.* (reply origin, instance named below)
Origin Instance Identifier	60	*APPL3B.* (instance of reply origin named above)
Correlator Control	0D	__
Application Correlator Control	01	Byte-offset 2 bit-settings: B'00000000'
Operate Report	0070	__
Date/Time	01	09:05:01 (local), 12/31/91
Reporting Operation	82	__
Operations Status	10	Accepted

Table 7-29 provides an example of an Operate Report CP-MSU reporting the successful execution of an Operate CP-MSU.

Table 7-29. Example Operate Report CP-MSU (Success)

Structure Name	ID/T	Contents
CP-MSU	1212	__
Routing and Targeting Instructions	154D	__
Name List	06	__
Cascaded Resource Name List	10	*APPL2.*
Destination Application Name	50	*APPL7.*
Origin Application Name	60	*APPL3.* (reply origin, instance named below)

Origin Instance Identifier	60	*APPL3B*. (instance of reply origin named above)
Correlator Control	0D	__
Application Correlator Control	01	Byte-offset 2 bit-settings: B'01100000'
Operate Report	0070	__
Date/Time	01	09:06:01 (local), 12/31/91
Reporting Operation	82	__
Operation Status	10	Successful
Partial Format Processing Method	1321	__
Coded Datastream Processing Method	10	Format of binary data in Transparent Coded Datastream
Transparent Coded Datastream	1309	Binary datastream (format defined in Partial Format Processing Method)

1.7.3.1.14 Routing/Parsing Report CP-MSU

The structure of the Routing/Parsing Report CP-MSU is shown in Table 7-30.

Table 7-30. The Routing/Parsing Report CP-MSU

Structure Name	ID/T	Occurrences Within Parent
CP-MSU	1212	_
Routing and Targeting Instructions	154D	0_1
SNA Condition Report	1532	1
Routing/Parsing Report	0077	1

The Routing/Parsing Report major vector serves three purposes:

- To indicate that an error has occurred while routing a request from a focal point or entry point to a second-level application program. It is also used to report that an error has occurred while routing a request *between* second-level application programs.

- To indicate that either zero or more than one major vector was included in a request (only one major vector may be present in any operations management request)

- To indicate that the major vector in the request is not recognized or is unsupported.

In each of the cases above, the details of the error are contained within a SNA Condition Report (X'1532') GDS variable carried in the report. See *SNA Formats* for the applicable sense data. See "Operations Management Use of the SNA Condition Report (SNACR)" in topic 1.7.3.1 for more information.

This report is not to be used for reporting MDS-related routing errors, which are handled by the MDS Router itself (see "The MDS Error Message" in topic 1.2.3.2.2).

An example of a secondary routing failure would be an unrecognized name in the Cascaded Resource Name List or Destination Application Name. Another example would occur when a specified Destination Instance Identifier is unrecognized. A Routing and Targeting Instructions GDS variable that is not present but required would be another example.

In each case, the proper SNA sense data would be returned to the originating application program named in the Routing and Targeting Instructions GDS variable (RTI) if one was present in the request. If an RTI was not present, the request is simply returned to the focal-point application with no further routing to be done.

Required Parameters

- SNA Condition Report (X'1532') GDS variable: provides detail about the error

Optional Parameters

- Routing and Targeting Instructions (X'154D') GDS variable: identifies a second-level application program target of the report. If the originator of the report is a second-level application program, then it is also identified here. In addition, this GDS variable contains an indication of how the Agent Unit-of-Work Correlator (X'1549') GDS variable is to be interpreted by the second-level application programs involved. A Routing and Targeting Instructions GDS variable within a Routing/Parsing Report always indicates that the Routing/Parsing Report comprises the last use of the AUOWC. See "The Routing and Targeting Instructions GDS Variable" in topic 1.7.3.3.1 and "MDS Message Types and Application-Level Flows" in topic 1.7.3.3.1 for more information.

Table 7-31 provides an example of a Routing/Parsing Report CP-MSU reporting a second-level routing or parsing error.

Table 7-31. Example Routing/Parsing Report CP-MSU

Structure Name	ID/T	Contents
CP-MSU	1212	__
Routing/Parsing Report	0077	__
Routing and Targeting Instructions	154D	__
Name List	06	__
Cascaded Resource Name List	10	*APPL2.*
Destination Application Name	50	*APPL7.*
Origin Application Name	60	*APPL3.* (reply origin, instance named below)
Origin Instance Identifier	60	*APPL3B.* (instance of reply origin named above)
Correlator Control	0D	__
Application Correlator Control	01	Byte-offset 2 bit-settings: B'01100000'
SNA Condition Report	1532	SNA sense data related to routing or parsing error

1.7.3.2 Implementation

Support of operations management is provided in an entry point by "EP_OPERATIONS_MGMT Function Set" in topic 2.10.6. This is an optional function set implemented by APPN nodes.

1.7.3.3 Format Usage

```
CONTROL POINT MANAGEMENT SERVICES UNIT (CP-MSU)
   Request Activation major vector (X'8066')
   Activation Acceptance major vector (X'0066')
   Activation major vector (X'0067')
   Operate major vector (X'8070')
   Operate Report major vector (X'0070')
   Request Deactivation major vector (X'8071')
   Deactivation Acceptance major vector (X'0071')
   Deactivation major vector (X'0072')
   Set Clock major vector (X'8075')
   Set Clock Report major vector (X'0075')
   Request Cancelation major vector (X'8076')
   Cancelation major vector (X'0076')
   Routing/Parsing Report major vector (X'0077')
```

Text Data parameter major vector (X'1300')
Transparent Coded Datastream parameter major vector (X'1309')
Text Command GDS variable (X'1320')
Partial Format Processing Method GDS variable (X'1321')
Display Datastream GDS variable (X'1322')
Context-Identified Values GDS variable (X'1323')
Context Identifier Group GDS variable (X'1324')
Context Identifier GDS variable (X'1325')
Value Group GDS variable (X'1326')
Character Value Descriptor GDS variable (X'1327')
Value Processing Method GDS variable (X'1328')
Value GDS variable (X'1329')
Hexadecimal Value Descriptor GDS variable (X'132A')
Nested Value Processing Method GDS variable (X'132B')
Value Instance Identifier GDS variable (X'1331')
SNA Condition Report GDS variable (X'1532')
Agent Unit of Work GDS variable (X'1549')
Routing and Targeting Instructions GDS variable (X'154D')
Flow: From CP to CP

Subtopics:

- 1.7.3.3.1 Operations Management Flows

1.7.3.3.1 Operations Management Flows

A network operator can send an operations management command to a specified application program at a specified CP. The command may be created as the result of following an operations plan. Alternatively, it may be created as the result of an Alert focal point forwarding an Alert to the operations management focal point. The latter case occurs when an operations management command should be sent to correct an Alert condition. See Chapter 3, "Problem Management" in topic 1.3 and "EP_ALERT Optional Subset 14 (Operations Management Support)" in topic 2.10.1.18 for more detail.

Data necessary for creating the request is passed to CPMS. CPMS formats the request into one of five operations management MS major vectors (MVs):

- Request Activation (X'8066') MV
- Operate (X'8070') MV
- Request Deactivation (X'8071') MV
- Set Clock (X'8075') MV
- Request Cancelation (X'8076') MV

CPMS also creates additional GDS variables and parameter major vectors necessary to carry extended routing and target-specific information. The requesting CPMS then encapsulates the major vector and additional structures (if present) in a CP-MSU, which is sent to the destination CPMS.

When the destination CPMS receives the request, it passes it to EP_OPERATIONS_MGMT, which in turn searches the request for a Routing and Targeting Instructions (X'154D') GDS variable (RTI). Target application program names may be transported in the Destination Application Name (X'50') Name List subfield (DAN) of the RTI. If a DAN is present, EP_OPERATIONS_MGMT passes the request to the application program named in the DAN.

The RTI may also include names of resources that constitute the "path" of resources that the request should take to get to the intended target (transported in the Cascaded Resource Name List (X'10') Name List subfield).

If multiple instances of a destination application program are active at the target system, then the Destination Instance Identifier (X'70') Name List subfield is used to name a specific instance. See "The Routing and Targeting Instructions GDS Variable" for more information.

If no RTI is present, or if no DAN is present in the RTI, then EP_OPERATIONS_MGMT is the intended target and it proceeds with the execution of the command. If the request is a Request Cancelation CP-MSU that specifies that requests sent to (or received by) EP_CHANGE_MGMT are to be canceled, then the CP-MSU is sent to EP_CHANGE_MGMT over MS protocol boundary BB. See "Protocol Boundary BB--Cancelation From Operations Management" in topic 2.9.1.18 for more information. Any reports (Cancelation CP-MSUs) generated as a result of such a cancelation are sent to EP_OPERATIONS_MGMT over MS protocol boundary CC. See "Request

Cancelation CP-MSU" in topic 1.7.3.1.10 and "Protocol Boundary CC--Reporting Cancelation to Operations Management" in topic 2.9.1.19 for more information.

If EP_OPERATIONS_MGMT is the intended target, but cannot process the request, EP_OPERATIONS_MGMT prepares a CP-MSU containing a Routing/Parsing Report (X'0077') MV, an SNA Condition Report (X'1532') GDS variable, and a Routing and Targeting Instructions (X'154D') GDS variable (if one is present in the request) and sends it to the originator of the request.

After performing the requested action, the target application program creates one or more of the following major vectors, depending on the request that was sent and the reporting requirements established by that request:

- Activation Acceptance (X'0066') MV
- Activation (X'0067') MV
- Operate Report (X'0070') MV
- Deactivation Acceptance (X'0071') MV
- Deactivation (X'0072') MV
- Set Clock Report (X'0075') MV
- Cancelation (X'0076') MV

These reports give two indications. First, the target indicates whether the request is accepted for processing. If it is not accepted, an SNA Condition Report (X'1532') GDS variable (SNACR) is returned (within the same CP-MSU as the acceptance report) with an appropriate sense data. Secondly, if the target does accept the request, an additional report is sent indicating the success/failure of the attempt. If the original request was a Request Activation (X'8066') MV, this additional report also indicates which of four activation stages has been reached. The issuer of the Request Activation MV may expect one of these additional reports for each stage that it is capable of reaching. Activation stage reporting allows the reporting entity to indicate that various levels of initialization have been completed.

If an error occurred during routing from EP_OPERATIONS_MGMT to a second-level application program (or between second-level application programs), then the Routing/Parsing Report (X'0077') MV is sent. This MV is also sent by second-level application programs that, during parsing of the received request, discovered that either:

- No MV was present in the request
- More than one MV was present in the request

The Routing/Parsing Report is sent in the latter two cases since the second-level application program cannot determine the intended request, and therefore, which reply (Activation Acceptance, Operate Report, Deactivation Acceptance, Set Clock Report, or Cancelation) to return.

After creating the appropriate major vector, the application program passes that major vector to CPMS, along with an RTI and SNACR if necessary. CPMS envelopes the structures in an CP-MSU, which it then sends to the appropriate destination.

The request and reporting flows are of the types described in "Request/Reply Flows for Non-Bulk Data" in topic 1.2.2.3 and "Unsolicited Flows" in topic 1.2.2.1 for APPN End Nodes. Refer to Figure 7-11 and Figure 7-12 for examples of operations management flows.

MDS Message Types and Application-Level Flows: The Flags (X'90') subvector of the MDS header contains various flags related to the *transport* of data between SNA/MS-registered application programs. It indicates if the MDS message type is an MDS request, an MDS reply, or an MDS error message. *These are all indications from the MDS perspective*, not the second-level application program perspective.

When the first operations management acceptance report (or reply) is received, a unit of work has been completed as far as MDS is concerned, since an MDS request/reply flow has completed. But, since possibly other acceptance reports and one or more completion reports need to be sent to complete the second-level application program unit of work, the same AUOWC may be used in the MDS header for those acceptance and completion reports. Reuse of an AUOWC by MS application programs is acceptable to MDS as long as any previous MDS request/reply transactions associated with the AUOWC are complete. Implementations may also use a private correlator (transported in the Correlator Control subvector) to relate second-level

application program units of work.

The Routing and Targeting Instructions (X'154D') GDS variable contains the necessary information related to correlation from the second-level application program perspective (as well as routing and targeting instructions information). See "The Routing and Targeting Instructions GDS Variable" for more information.

The Application Correlator Control (X'01') subfield of the Correlator Control (X'0D') subvector is the format used to relate requests, replies, and reports as being part of the same second-level application program unit of work. More specifically, 3 bit settings in the Application Correlator Control subfield determine three pieces of information concerning the second-level application program unit of work:

Bit Offset	*Bit Names, Settings and Their Meanings*	
0	Correlation First Message Indicator:	
	0	Not the first use of the correlator
	1	First use of the correlator
1	Correlation Last Message Indicator:	
	0	Not the last use of the correlator
	1	Last use of the correlator
2	Correlation Iteration Indicator:	
	0	Not the last use of the correlator for the current iteration, or not part of an iteration
	1	Last use of the correlator for the current iteration of command execution, of which there could be more than one.

The correlation iteration indicator (bit offset 2) above requires further explanation. When an application program is requested to execute a command multiple times (iterations), it may return status reports for each iteration of the command that it executes. So there must be a method for the reports to differentiate the last reports *for each iteration* as well as the last report *for the entire scope of the original request*. Bit offset 2, in various combinations with bit offset 1, provides this capability. For example:

An Activate request is sent to an instance of some second-level application program. The Activate is to be executed three times, based on execution window timing parameters (X'0B' MS common subvector) in the request. The second-level application program is to send status reports as specified in the Reporting Level (X'09') MS common subvector. Assume, for this example, that two status milestones are important to the command issuer: acceptance and success. The target is capable of reporting the first, second, and fourth stages of success (as specified in the Activation Stage (X'18') Reporting Activation subvector):

- Initial connectivity complete, the target is at least capable of communicating
- Target hardware active, target application software not yet active
- Target application function has reached a point where a quiesce of that function is required prior to any disruptive command being executed
- The target resource has been fully activated in compliance with the parameters supplied (and optionally referenced) in the activation request.

Figure 7-9 and Table 7-32, illustrate how, given the scenario above, the MDS header flag bits and the Correlator Control subvector flag bits would be set in the reply and reports.

```
  ___________     ________________  ________________    ___________
 |           |   |                | |                |  |           |
 | Origin    |   | Operations     | | EP_OPERATIONS_ |  | Destination|
 | Application|  | Management     | | MGMT           |  | Application|
 | (OAN)     |   | focal point    | |                |  | (DAN)     |
 |           |   |                | |                |  |           |
 |___________|   |________________| |________________|  |___________|
       °                 °                 °                  °
       ° Activate Request °                 °                  °
 1)    °_________________ÿ°                 °                  °
```

```
        °                  °_________________ÿ°                  °
        °                  °                  °_________________ÿ°
        °                  °                  °                  °
        °                  °                  ° Activate Reply   °
        °                  °                  °_________________°
        °                  °_________________°                  °
 2)     °_________________°                  ° Activate Report  °
        °                  °                  °_________________°
        °                  °_________________°                  °  _|
 3)     °_________________°                  ° Activate Report  °   |
        °                  °                  °_________________°   | First iteration
        °                  °_________________°                  °   |
 4)     °_________________°                  ° Activate Report  °   |
        °                  °                  °_________________°  _|
        °                  °_________________°                  °
 5)     °_________________°                  ° Activate Report  °
        °                  °                  °_________________°
        °                  °_________________°                  °  _
 6)     °_________________°                  ° Activate Report  °   |
        °                  °                  °_________________°   | Second iteration
        °                  °_________________°                  °   |
 7)     °_________________°                  ° Activate Report  °   |
        °                  °                  °_________________°  _|
        °                  °_________________°                  °
 8)     °_________________°                  ° Activate Report  °
        °                  °                  °_________________°
        °                  °_________________°                  °  _
 9)     °_________________°                  ° Activate Report  °   |
        °                  °                  °_________________°   | Third iteration
        °                  °_________________°                  °   |
 10)    °_________________°                  ° Activate Report  °   |
        °                  °                  °_________________°  _|
        °                  °_________________°                  °
 11)    °_________________°                  °                  °
        °                  °                  °                  °
        °                  °                  °                  °
        °                  °                  °                  °
```

Figure 7-9. Example Activation Scenario. This example represents a Request Activation command (specifying three command iterations) being sent from a second-level application program served by the operations management focal point to a second-level application program served by EP_OPERATIONS_MGMT. The accompanying acceptance reply and completion reports (X'0066' and X'0067', respectively) follow as shown. Table 7-32 lists each flow by number, showing the flag settings for both the Flags (X'90') MULTIPLE_DOMAIN_SUPPORT Routing Information subvector and the Application Correlator Control (X'01') Correlator Control subfield.

In Figure 7-9, flows 3, 6, and 9 report that initial connectivity is complete for their respective iterations.

Flows 4, 7, and 10 report that the target hardware, but not the target software, is active for their respective iterations.

And lastly, flows 5, 8, and 11 report that the target has been fully activated for their respective iterations.

Table 7-32. Example MULTIPLE_DOMAIN_SUPPORT Header and Application Correlator Control Bit Settings. Refer to Figure 7-9 for the example activation flows listed in this table. Settings for both the Flags (X'90') MULTIPLE_DOMAIN_SUPPORT Routing Information subvector and the Application Correlator Control (X'01') Correlator Control subfield are shown.

Flow	Flags (X'90') subvector		Application Correlator Control (X'01') subfield		
	First MDS Msg Indicator	Last MDS Msg Indicator	Correlator First Msg Indicator	Correlator Last Msg Indicator	Correlator Iteration Indicator
1	1 (first)	0 (not last)	1 (first)	0 (not last)	0 (not last)
2	0 (not first)	1 (last)	0 (not first)	0 (not last)	0 (not last)
3	1 (first)	1 (last)	0 (not first)	0 (not last)	0 (not last)
4	1 (first)	1 (last)	0 (not first)	0 (not last)	0 (not last)
5	1 (first)	1 (last)	0 (not first)	0 (not last)	1 (last)
6	1 (first)	1 (last)	0 (not first)	0 (not last)	0 (not last)
7	1 (first)	1 (last)	0 (not	0 (not	0 (not

			first)	last)	last)
8	1 (first)	1 (last)	0 (not first)	0 (not last)	1 (last)
9	1 (first)	1 (last)	0 (not first)	0 (not last)	0 (not last)
10	1 (first)	1 (last)	0 (not first)	0 (not last)	0 (not last)
11	1 (first)	1 (last)	0 (not first)	1 (last)	1 (last)

The originator of a request cannot know in advance the outcome of the execution of that request.

For those cases when a subsequent report will be generated for the command request for which a current report is being returned, then the last message indicator bit is set to 0 (not last). This same logic applies where the execution of an interval-driven command is involved, but with respect to the iteration indicator. Likewise, when the current report being returned is indeed the last report of a current interval, then the correlator iteration indicator bit is set to 1 (last). Similarly, when the current report being returned is the last report of the very last or only execution, then the last message indicator is set to 1 (last).

For those cases where it is uncertain whether or not a subsequent report will be sent (such uncertainty arises when the execution of a command request may succeed or fail, and the status-reporting requirements are such that one report status would cause a report to be produced while the other report status would cause the associated report to be suppressed), both the last message indicator and the iteration indicator, appropriate to their context, are set to 0 (not last).

If the status-reporting requirement makes the sending of a report optional, additional uncertainty (from the perspective of the command issuer) arises. This is because the product implementation may choose to produce or not produce reports according to internal design judgments.

Therefore, the general rule is that if it is possible to produce yet another report associated with the same command request, then the last message indicator and the iteration indicator bits are set to 0 (not last). Second-level application program timeouts are considered to be product-specific settings. Implementations should take delayed reporting into consideration when planning iterative operations management commands.

As illustrated, the Correlator Control subvector allows the AUOWC to be used by both MDS and the second-level application programs independently of the other, for distinctly different purposes. Delayed replies and reports can be correlated with their associated requests (possibly across multiple execution iterations) at a second-level application program level, while as far as the MDS Transport is concerned, new units of work are in progress.

It is possible for an unsolicited flow (namely an Operate Report, X'0070') to be unrelated to any previous request (at the transport level *and* at the application program level). An example of this would occur when an application program sends an unsolicited message (unrelated to any previous request) to the network operator.

In this particular case, both the MDS header and the Application Correlator Control subfield flags indicate that the message is the "first (1)" for the unit of work. However, the correlator last-message indicator setting of the Application Correlator Control subfield depends on the context of the message and whether or not corresponding second-level requests, replies, and reports are expected by the issuer. Any such traffic is correlated with the same AUOWC (or private correlator) as the original, unsolicited message, therefore requiring that the "last" flag of the message be set to "not last (0)." If no action is anticipated by the unsolicited message issuer, then the correlator last-message indicator is set to "last." The last-message indicator in the MDS header (in the unsolicited message) is set to "last" regardless of whether or not additional second-level requests, replies, or reports are anticipated. This is consistent with all other unsolicited reports in operations management; they are treated by MDS as both the first and last use of the AUOWC.

The Routing and Targeting Instructions GDS Variable: The Routing and Targeting Instructions (X'154D') GDS variable allows the routing of requests, replies, and error messages beyond the entry point (EP) and focal point (FP) MS application programs. The RTI contains structures that allow specification of both SNA-registered and non-SNA-registered application program names. These programs are referred to as *second-level* application programs.

The RTI also provides an indication of the use/reuse of the AUOWC (or private correlator) by the application programs named in the RTI. See

"MDS Message Types and Application-Level Flows" for a more detailed explanation of this function.

The RTI is composed of two subvectors:

- Name List (X'06')

- Correlator Control (X'0D')

The Correlator Control subvector is discussed in "MDS Message Types and Application-Level Flows."

The following subfields of the Name List subvector are used in operations management:

- Origin Application Name (X'60'): The name of the application program that originated the request, reply, or report. In requests, the application program named in this subfield will be identified by the Destination Application Name (see below) in the RTI of the replies and reports that result from the request being executed.

- Origin Instance Identifier (X'80'): If multiple instances of the application program named in the Origin Application Name exist, then the instance that sent the request, reply, or report may be identified here. In requests, the application program named in this subfield will be the Destination Instance Identifier (see below) in the RTI of the replies and reports that result from the request being sent.

- Destination Application Name (X'50'): The application program that is the target of the request, reply, or report. In requests, the application program named in this subfield will be the Origin Application Name in the RTI of the replies and reports that result from the request being executed.

- Destination Instance Identifier (X'70'): If multiple instances of the application program named in the Destination Application Name exist, then the instance that is the target may be identified here. In requests, The application program named in this subfield will be the Origin Instance Identifier in the RTI of the replies and reports that result from the request being executed.

- Cascaded Resource Name List (X'10'): An entry-point second-level application program involved in operations management requests, replies, and reports may not be able to communicate directly with EP_OPERATIONS_MGMT. Any requests directed to it (and replies and reports it generates) may need to be forwarded along a path of one or more hierarchically related application programs. The Cascaded Resource Name List subfield is used for this purpose.

  For requests, the first name in the list identifies the application program that *can* directly communicate with EP_OPERATIONS_MGMT. The last name in the list identifies the application program that can communicate with the application program that is the actual target of requests being sent. See Figure 7-10 for an example flow.

  For replies and reports, the order is reversed. The first name in the list identifies the application program that can communicate with the application program that is the actual target of requests being sent. The last name in the list identifies the application program that can directly communicate with EP_OPERATIONS_MGMT.

```
                                |
                                |   Incoming MDS-MU
                                |
                                |

                       ____________________
                      |                    |
 MS Application       |  EP_OPERATIONS_MGMT |
 Program              |                    |
                      |________ __________|
                                |
                                |
                                |
                                |

                       ____________________
                      |                    |
 Second-level         |  Cascaded Resource  |     Hierarchical Level 1
 Application          |  Name List, Entry 1 |
 Program              |                    |
                      |________ __________|
                                |
                                |
                                |
                                |

                       ____________________
```

```
                        |                    |
 Second-level           |  Cascaded Resource |     Hierarchical Level 2
 Application            |  Name List, Entry 2 |
 Program                |                    |
                        |____________________|
                                  |
                                  |
                         _________|__________
                        |                    |
                        |                    |


                 _______________      _________________
                |               |    |                 |
 Second-level   | Destination   |    | Destination     |    Hierarchical
 Application    | Appl Program 1|    | Appl Program 1  |    Level 3
 Program        | Instance A    |    | Instance B      |
                |               |    |                 |
                |_______________|    |_________________|
```

Figure 7-10. Second-level Routing Using the Routing and Targeting Instructions GDS variable. This example shows the relationship between EP_OPERATIONS_MGMT and the Cascaded Resource Name List, the Destination Application Name, and the Destination Instance Identifier in a request. The method of transport to (and between) second-level application programs is an implementation choice not defined by SNA/MS. The transport, however, must deliver the entire MDS_MU to the target. Note that this figure illustrates three hierarchical levels of application programs "away" from EP_OPERATIONS_MGMT.

Request/Reply Flows:

The following figure illustrates request and acceptance reply flows for operations management.

```
     Network            Requesting       Destination              Second-level           Second-level
     Operator              CPMS             CPMS                  Application (CRL)      Application (DAN)
        .                   .                .                        .                      .
        .                   .                .                        .                      .
        .     command       .                .                        .                      .
        ____________________ÿ                .                        .                      .
        .                   . request MDS-MU .                        .                      .
        .                   º________________ÿ                        .                      .
        .                   .                .    request MDS-MU      .     request MDS-MU   .
        .                   .                º________________________ÿ_____________________ÿ.
        .                   .                .  accept. reply MDS-MU  .  accept. reply MDS-MU .
        .                   .                _________________________º_______________________.
        .                   . MDS-MU reply   .                        .                      .
        .                   _________________º                       .                      .
        .  acceptance reply .                .                        .                      .
        ___________________º                .                        .                      .
        .                   .                .                        .                      .
        . Reply is the      .                .                        .                      .
        . entire MDS_MU if  .                .                        .                      .
        . a second-level    .                .                        .                      .
        . application pgm is.                .                        .                      .
        . providing network .                .                        .                      .
        . operator function..                .                        .                      .
```

Figure 7-11. The Request/Reply Flow for Non-Bulk Data from an APPN Network Node or End Node. The diagram shows an example of one entry in a Cascaded Resource Name List subfield (CRL), which acts as the "interface" between EP_OPERATIONS_MGMT and the target Destination Application Name (DAN) in the request. If more than one CRL entry is present, a request/reply flow would exist between each entry named. If no CRL is present in the request, the request would flow directly from the destination CPMS to the application program named in the DAN.

Report or Unsolicited Flows:

The following figure illustrates an operations management report or unsolicited message flow. This flow is unsolicited in the MDS context and may be unsolicited in the second-level application program context. The correlator control subvector settings in such a flow indicate how it is related to a prior request, if related at all. If it is not related to a prior request, it is considered to be a message to the network operator and an ancillary report.

```
Network            Destination        Origin            Second-level       Second-level
Operator              CPMS             CPMS             Application (CRL)  Application (OAN)
   .                   .                .                   .                  .
   .                   .                .                   .                  .
   .                   .                .   message or      .   message or     .
   .                   .                .   report MDS-MU   .   report MDS-MU  .
   .                   . message or     ____________________o___________________.
   .                   . report MDS-MU  .                   .                  .
   .    message or     _________________o                   .                  .
   .    report         .                .                   .                  .
   .__________________o                .                   .                  .
   .                   .                .                   .                  .
   .                   .                .                   .                  .
   . This flow is the  .                .                   .                  .
   . entire MDS_MU if  .                .                   .                  .
   . a second-level    .                .                   .                  .
   . application pgm is.                .                   .                  .
   . providing network .                .                   .                  .
   . operator function..                .                   .                  .
   .                   .                .                   .                  .
```

Figure 7-12. The Unsolicited Flow for an APPN End Node. The diagram shows an example of one entry in a Cascaded Resource Name List subfield (CRL), which acts as the "interface" between EP_OPERATIONS_MGMT and the Origin Application Name (OAN) in the message/report. If more than one CRL entry is present, a request/reply flow exists between each entry named. If no CRL is present, then the request is delivered directly from the OAN to the origin CPMS.

2.0 Part 2. Architectural Logic for Management Services

Subtopics:

- 2.8 Chapter 8. Role Requirements of Management Services Components
- 2.9 Chapter 9. General Management Services Function Sets
- 2.10 Chapter 10. Specialized Management Services Function Sets for Entry Points

---

2.8 Chapter 8. Role Requirements of Management Services Components

This chapter introduces subsequent chapters by discussing the conventions used and function sets described in those chapters, and it specifies for the implementer, roles available for a management services component. The reader of this chapter should be familiar with the material in "Implementation Choices" in topic 1.1.4.

This chapter describes the function sets required for the various MS roles, the function sets optionally available for each role, and the dependence of the function sets upon one another.

Subtopics:

- 2.8.1 Role Requirements for Management Services Components

2.8.1 Role Requirements for Management Services Components

The role requirements for management services components are categorized according to the type of node in which the component resides. Refer to "SNA Node Types" in topic 1.1.2.1 for a description of the various node types within SNA. The terminology defined in that introduction is used in the sections that follow.

Subtopics:

- 2.8.1.1 Physical Unit Management Services (PUMS) in a Type 2.0 Node
- 2.8.1.2 Control Point Management Services (CPMS) in an APPN End Node or Network Node

---

2.8.1.1 Physical Unit Management Services (PUMS) in a Type 2.0 Node

Introductory material on roles can be found in "Role Requirements" in topic 1.1.4.3 and Figure 1-11 in topic 1.1.4.3.

```
 _________________________ ________ _________ ____________________
|     EP_CHANGE_MGMT      | EP_RTM | EP_QPI  |    EP_COMMON_      |
|                         |        |         |OPERATIONS_SERVICES |
|_________________________|________|_________|____________________|
|  FILE_SERVICES_SUPPORT  |   RECEIVE_REQUEST_SSCP_PU             |
|                         |                                       |
|_______________________________________________________________________|
|                                 SEND_DATA_SSCP_PU                     |
|                                                                       |
|_______________________________________________________________________|
```

Figure 8-1. Function Sets for the PUMS Type 2.0 Node Role

All implementations of type 2.0 nodes (1) provide the management services described in the base subset of the following:

- "SEND_DATA_SSCP_PU Function Set" in topic 2.9.5
- "EP_ALERT Function Set" in topic 2.10.1

The following function sets are available as options to implementations of type 2.0 nodes. Refer to Figure 8-1 for the relationship of these function sets to one another (a function set requires the presence of a function set below it in the figure).

- "RECEIVE_REQUEST_SSCP_PU Function Set" in topic 2.9.6
- "FILE_SERVICES_SUPPORT Function Set" in topic 2.9.4
- "EP_CHANGE_MGMT Function Set" in topic 2.10.4
- "EP_RTM Function Set" in topic 2.10.2
- "EP_QPI Function Set" in topic 2.10.3
- "EP_COMMON_OPERATIONS_SERVICES Function Set" in topic 2.10.5

---

(1) This role also applies to boundary-function-attached type 2.1 nodes.

| 2.8.1.2 Control Point Management Services (CPMS) in an APPN End Node or Network Node

Introductory material on roles can be found in "Role Requirements" in topic 1.1.4.3 and Figure 1-11 in topic 1.1.4.3.

```
°°°°°°°°°
°       °
° EP_   °
° ALERT °
°       °
°_______°°°°°°°°°_____________    ____________
|               °             |   |           |
|    MS_CAPS    °   EP_XXXX   |   | EP_CHANGE_ |
|               °             |   | MGMT      |
|               °             |   |           |
|               °             |   |           |
|_______________°°°°°°°°°
```

- EP_COMMON_OPERATIONS_SERVICES

Figure 8-2. Function Sets for the CPMS Role for APPN End Nodes or Network Nodes

All implementations of CPMS in an APPN end node or APPN network node provide the management services described in the base subset of the following function sets:

- "MULTIPLE_DOMAIN_SUPPORT Function Set" in topic 2.9.2
- "MS_CAPS Function Set" in topic 2.9.3
- "EP_ALERT Function Set" in topic 2.10.1

The following function sets are available as options to implementations of CPMS in an APPN end node or APPN network node. Refer to Figure 8-2 for the relationship of these function sets to one another.

- "FILE_SERVICES_SUPPORT Function Set" in topic 2.9.4
- "EP_CHANGE_MGMT Function Set" in topic 2.10.4
- "EP_COMMON_OPERATIONS_SERVICES Function Set" in topic 2.10.5
- "EP_OPERATIONS_MGMT Function Set" in topic 2.10.6

2.9 Chapter 9. General Management Services Function Sets

Subtopics:

- 2.9.1 Protocol Boundaries in the Management Services Model
- 2.9.2 MULTIPLE_DOMAIN_SUPPORT Function Set
- 2.9.3 MS_CAPS Function Set
- 2.9.4 FILE_SERVICES_SUPPORT Function Set
- 2.9.5 SEND_DATA_SSCP_PU Function Set
- 2.9.6 RECEIVE_REQUEST_SSCP_PU Function Set

2.9.1 Protocol Boundaries in the Management Services Model

The architectural model for management services presented in Chapters 9-10 decomposes the overall network management capabilities in a node into a number of management services *function sets*. "Conventions Used in Describing Function Sets" in topic 2.9.1.1 summarizes the conventions used when these function sets are presented in Chapters 9-10. "Protocol Boundaries Between MS Function Set Groups" in topic 2.9.1.2 lists all of the lettered protocol boundaries over which data flows between management services function set groups.

Implementations are *not* required to match the architectural model of this book internally. An implementation can diverge from this formal model internally, but not in ways such that technical variations can be detected outside its node.

Subtopics:

- 2.9.1.1 Conventions Used in Describing Function Sets
- 2.9.1.2 Protocol Boundaries Between MS Function Set Groups
- 2.9.1.3 Protocol Boundary A--Send NMVT
- 2.9.1.4 Protocol Boundary B--Held Alert Processing
- 2.9.1.5 Protocol Boundary C--Delayed Alert Processing
- 2.9.1.6 Protocol Boundary D--NMVT Received
- 2.9.1.7 Protocol Boundary E--Send NMVT Response
- 2.9.1.8 Protocol Boundary F--Send MS Bulk Data
- 2.9.1.9 Protocol Boundary G--MS Bulk Data Received
- 2.9.1.10 Protocol Boundary N--MDS Error Message Information
- 2.9.1.11 Protocol Boundary O--Register Application Program Name
- 2.9.1.12 Protocol Boundary O--MDS-MU Received
- 2.9.1.13 Protocol Boundary R--Send MDS-MU

- 2.9.1.14 Protocol Boundary S--Forward CP Status
- 2.9.1.15 Protocol Boundary T--Register MS Category
- 2.9.1.16 Protocol Boundary U--Begin Sending CP Status Information
- 2.9.1.17 Protocol Boundary V--Current Focal Point
- 2.9.1.18 Protocol Boundary BB--Cancelation From Operations Management
- 2.9.1.19 Protocol Boundary CC--Reporting Cancelation to Operations Management

---

2.9.1.1 Conventions Used in Describing Function Sets

The function sets described in detail in Chapters 9-10 are based on the model previously introduced (see "Base and Optional Subsets of the Function Sets" in topic 1.1.4.2). They state the architectural requirements for implementing the subset of CPMS or PUMS that provides the function being described by the function set, and the subset of each of the other nodal components that assist CPMS or PUMS.

Each function set has a corresponding function set group. A diagram of each function set group indicates the nodal components (or subsets of the nodal components) providing the function described by the function set, and the relationship of the subject function set group to other function set groups. Refer to Figure 9-1.

```
                                                        ....................................
                                                        .                                  .
                                                        . function set group interfacing   .
                                                        . with subject function set group  .
                                                        .                                  .
                                                        ....................................
                                                                         "      |
                                                                         |(x)   |(x)
¤¤¤
```

Each function set group outside the subject function set group being described is bounded by a lighter dotted line (.....). The functional composition of these function set groups is not shown.

Each function set falls into one of two categories, general or specialized. A general function set is a management services function set that provides a function used by multiple management services categories, such as transport of management services data. A specialized function set is a management services function set that processes data for a particular management services function. The table shown in Table 9-1 indicates each function set's category.

Table 9-1. Table of General and Specialized Function Sets for the PUMS Role

General Function Sets	Specialized Function Sets
SEND_DATA_SSCP_PU	EP_ALERT
RECEIVE_REQUEST_SSCP_PU	EP_RTM
FILE_SERVICES_SUPPORT	EP_QPI
	EP_CHANGE_MGMT
	EP_COMMON_OPERATIONS_SERVICES

Table 9-2. Table of General and Specialized Function Sets for the CPMS Role (APPN End Node)

General Function Sets	Specialized Function Sets
MULTIPLE_DOMAIN_SUPPORT	EP_ALERT
MS_CAPS	EP_COMMON_OPERATIONS_ SERVICES
FILE_SERVICES_SUPPORT	EP_CHANGE_MGMT
	EP_OPERATIONS_MGMT

2.9.1.2 Protocol Boundaries Between MS Function Set Groups

2.9.1.3 Protocol Boundary A--Send NMVT

Origin	EP_XXXX
Destination	SEND_DATA_SSCP_PU
Data Content	1. A pointer to the completed NMVT 2. For solicited data, the address of the control point that sent the request
Initiates	Process described in "Sending NMVTs" in topic 2.9.5.4.3

2.9.1.4 Protocol Boundary B--Held Alert Processing

Origin	SEND_DATA_SSCP_PU
Destination	EP_ALERT (supporting optional subset 3--Held Alert)
Data Content	1. A pointer to a completely constructed NMVT If the pointer is 0, EP_ALERT is being requested to

	process an existing held Alert or Resolution major vector. If the pointer is nonzero, it contains a pointer to an NMVT to be queued.
Initiates	Process described in "Sending Held Alerts" in topic 2.10.1.7.4

2.9.1.5 Protocol Boundary C--Delayed Alert Processing

Origin	SEND_DATA_SSCP_PU or EP_ALERT
Destination	EP_ALERT (implementing optional subset 2--Delayed Alert) or SEND_DATA_SSCP_PU
Request Data Content	1. A pointer to a completely constructed NMVT
Reply Data Content	1. An indicator whether the NMVT can be sent at this time If the indicator is 0, the calling process can send the NMVT at this time. If the indicator is nonzero, the NMVT cannot be sent at this time and is being processed by this optional subset.
Initiates	Process described in "Processing Delayed Alerts" in topic 2.10.1.6.3

2.9.1.6 Protocol Boundary D--NMVT Received

Origin	RECEIVE_REQUEST_SSCP_PU
Destination	EP_XXXX
Data Content	1. A pointer to the NMVT 2. The address of the control point from which the NMVT was received
Initiates	Passing data across this protocol boundary causes one of the following processes to be started, determined by the key of the major vector in the NMVT referred to by the pointer in item 1. ° "Receiving RTM Requests" in topic 2.10.2.4.4 ° "Receiving OPI requests" in topic 2.10.3.4.3 ° "Receiving Common Operations Services Requests" in topic 2.10.5.4.4

2.9.1.7 Protocol Boundary E--Send NMVT Response

Origin	EP_XXXX
Destination	RECEIVE_REQUEST_SSCP_PU
Data Content	The following items, used to construct and send an NMVT response RU: 1. The address of the control point which is to be sent a +RSP or -RSP 2. Either the 8-digit hexadecimal sense data that is to be sent on a -RSP, or the value X'0000 0000', indicating that a +RSP is to be sent
Initiates	Process described in "Sending NMVT Responses" in topic 2.9.6.4.3

2.9.1.8 Protocol Boundary F--Send MS Bulk Data

Origin	EP_XXXX
Destination	FILE_SERVICES_SUPPORT
Data Content	Information defining the FS request, consisting of these elements: 1. CP-MSU, to be placed in the agent object 2. Agent unit-of-work correlator 3. Server object parameters 4. List of target locations (NETID and LUNAME for each) 5. SNA/DS priority and security level requested
Initiates	"Sending Requests and Data" in topic 2.9.4.4.3

2.9.1.9 Protocol Boundary G--MS Bulk Data Received

Origin	FILE_SERVICES_SUPPORT
Destination	EP_XXXX
Data Content	Information defining the FS report, consisting of these elements: 1. CP-MSU 2. Agent unit-of-work correlator 3. Server object parameters 4. Source location (NETID and LUNAME)
Initiates	Appropriate specialized management services function set process for receiving data

2.9.1.10 Protocol Boundary N--MDS Error Message Information

Origin	MULTIPLE_DOMAIN_SUPPORT
Destination	MS_CAPS
Data Content	Whenever MULTIPLE_DOMAIN_SUPPORT processes an MDS error message, it creates this signal containing the key information from the error message, and passes it to MS_CAPS. MS_CAPS uses this information to determine if focal point to entry point application relationships are impacted. 1. Origin of MDS error message ° NETID ° NAU name ° MS application program name 2. Destination of MDS error message ° NETID ° NAU name ° MS application program name 3. SNA report code (sense data) 4. Reported on location ° NETID ° NAU name ° MS application program name 5. Session over which MDS error message was received: ° Network-qualified NAU name of partner ° Mode name (SNASVCMG | CPSVCMG) 6. Session over which MDS error message was sent: ° Network-qualified NAU name of partner ° Mode name (SNASVCMG | CPSVCMG)
Initiates	Process described in "MS_CAPS Function Set" in topic 2.9.3.

2.9.1.11 Protocol Boundary O--Register Application Program Name

Origin	EP_XXXX | MS_CAPS
Destination	MULTIPLE_DOMAIN_SUPPORT
Data Content	1. Application name 2. Action

	° ADD application name ° DELETE application name
Initiates	Process described in "Processing Register Application Program Name" in topic 2.9.2.4.8

2.9.1.12 Protocol Boundary Q--MDS-MU Received

Origin	MULTIPLE_DOMAIN_SUPPORT
Destination	EP_XXXX | MS_CAPS
Data Content	1. An MDS-MU
Initiates	The "Receiving MDS-MUs" process in the function set group to which the MDS-MU is being passed

2.9.1.13 Protocol Boundary R--Send MDS-MU

Origin	EP_XXXX | MS_CAPS
Destination	MUPTIPLE_DOMAIN_SUPPORT
Data Content	1. An MDS-MU 2. An indication of the desired routing: ° DIRECT ° DEFAULT
Initiates	Process described in "Processing Send MDS-MU" in topic 2.9.2.4.8

2.9.1.14 Protocol Boundary S--Forward CP Status

Origin	MULTIPLE_DOMAIN_SUPPORT
Destination	MS_CAPS
Data Content	1. A list of entries, each of which contains a CP Status Message for a client end node (in a network node), or a CP Status Message for the network node server (in an end node). The CP status message includes the following information: ° NETID.CPNAME of the node ° Status of the node - PENDING-ACTIVE - ACTIVE - INACTIVE ° The CP Capabilities (X'12C1') GDS variable received from the node (present if status = ACTIVE)
Initiates	Process described in "Receiving CP Status Updates" in topic 2.9.3.4.3

2.9.1.15 Protocol Boundary T--Register MS Category

Origin	EP_XXXX
Destination	MS_CAPS
Request Data Content	1. Application program name 2. MS category
Reply Data Content	Return code: ° OK (request accepted)
Initiates	Process described in "Registering Application Programs" in topic 2.9.3.4.3

2.9.1.16 Protocol Boundary U--Begin Sending CP Status Information

Origin	MS_CAPS
Destination	MULTIPLE_DOMAIN_SUPPORT
Data Content	(None) An initial request for information is sent to the MULTIPLE_DOMAIN_SUPPORT component from MS_CAPS, to cause it to begin sending CP status information via protocol boundary S.
Initiates	(none)

2.9.1.17 Protocol Boundary V--Current Focal Point

Origin	MS_CAPS
Destination	EP_XXXX
Data Content	The following information is passed across this protocol boundary in the form of an MS Capabilities major vector, containing a Focal Point Notification (X'E1') subvector and a Focal Point Identification (X'21') subvector: 1. MS category 2. Focal point name (NETID.NAU.APPL) The following information is also passed: 3. Type of routing to be requested for routing MDS-MUs to this focal point (this indication is passed on "Protocol Boundary R--Send MDS-MU" in topic 2.9.1.13): ° DIRECT ° DEFAULT
Initiates	Process in the receiving function set group that maintains current focal point knowledge

2.9.1.18 Protocol Boundary BB--Cancelation From Operations Management

Origin	EP_OPERATIONS_MGMT
Destination	EP_CHANGE_MGMT
Data Content	° A CP-MSU containing a Request Cancelation (X'8076') major vector
Initiates	Process described in "EP_CHANGE_MGMT Optional Subset 7 (Cancel Command Support)" in topic 2.10.4.11.

2.9.1.19 Protocol Boundary CC--Reporting Cancelation to Operations Management

Origin	EP_CHANGE_MGMT
Destination	EP_OPERATIONS_MGMT
Data Content	° A CP-MSU containing a Cancelation (X'0076') major vector
Initiates	Process described in "Sending Operations Management Data" in topic 2.10.6.4.3

2.9.2 MULTIPLE_DOMAIN_SUPPORT Function Set

| ..... |

```
|                .                .
|        __________.    MS_CAPS    .
|      |    ______.  function set  ._______
|      |    |    _ÿ.     group     .___    |
|      |    |    |  .              .   |    |
|      |    |    |  ................   |    |
|      |    |    |      "   "   |       |    |
|      |    |    |      |   |   |       |    |
|      |(O)|(R)|(Q)   |(N)|(T)|(V)     |(U)|(S)
| °°°°|°°°|°°°|°°°°°°|°°°|°°°|°°°°°°°°|°°°|°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°
| °         |      |   |             |                 _______________   °
| °  ___________________________________________   (11)  | session      |  °
| °  |                                         |_______| services     |  °
| °  |                                         |       |______________|  °
| °  |                                         |        ______________   °
| °  |                                         |  (20)  | directory    |  °
| °  |                                         |______ÿ| services     |  °
| °  |                                         |       |______________|  °
| °  |                                         |        ______________   °
| °  |                                         |  (21)  | presentation |  °
| °  |                                   MS¹   |______ÿ| services     |  °
| °  |_________________________________________|       |______________|  °
| °                 "   "   |                                            °
| °                 |   |   |                    MULTIPLE_DOMAIN_SUPPORT  °
| °                 |   |   |                        function set group  °
| °°°°°°°°°°°°°°°°°°|°°°|°°°|°°°°°°°°°°°°°°°°°°°
```

an MDS-MU to MULTIPLE_DOMAIN_SUPPORT for routing to a destination. The signal passed is described in "Protocol Boundary R--Send MDS-MU" in topic 2.9.1.13.

The processing of this signal is described in "Processing Send MDS-MU" in topic 2.9.2.4.8.

- Internal MS protocol boundary U (Begin Sending CP Status Information)

This protocol boundary is used by MS Capabilities to indicate that MULTIPLE_DOMAIN_SUPPORT is to begin sending CP status information via protocol boundary S. In an APPN end node, the CP status information passed concerns the network node server; in an APPN network node, the CP status information passed concerns the client end nodes. in the network node domain. All current status information is passed back immediately; subsequent status changes will then be reported as well.

The signal passed across this protocol boundary is described in "Protocol Boundary U--Begin Sending CP Status Information" in topic 2.9.1.16.

The processing of this signal is described in "Processing Begin Sending CP Status Information" in topic 2.9.2.4.8.

- Internal MS protocol boundary V (Current Focal Point)

The identity of an application program's current focal point is passed across protocol boundary V from the MS_CAPS function set group. The application program must have previously registered itself with MS_CAPS via protocol boundary T. The details of this protocol boundary are described in "Protocol Boundary V--Current Focal Point" in topic 2.9.1.17.

The process by which an application program registers itself with MS_CAPS is described in "Application Program Registration with MS_CAPS" in topic 2.9.3.10.

The MDS_ROUTER maintains awareness of the Alert focal point so that Alerts received from back-level nodes can be addressed and forwarded.

The processing of this signal is described in "Processing Current Focal Point" in topic 2.9.2.4.8.

° Output

- Internal MS protocol boundary N (Copied MDS Error Message)

This protocol boundary is used by MULTIPLE_DOMAIN_SUPPORT to pass a copy of an MDS error message to MS_CAPS. MS_CAPS will receive a copy of every MDS error message received or created by MULTIPLE_DOMAIN_SUPPORT. MS_CAPS examines these error messages to determine if focal point-entry point relationships have been impacted. The signal passed across this protocol boundary is described in "Protocol Boundary N--MDS Error Message Information" in topic 2.9.1.10.

The creation of this signal is described in "MDS Routing Algorithm" in topic 2.9.2.4.8.

- Internal MS protocol boundary Q (MDS-MU Received)

This protocol boundary is used by MULTIPLE_DOMAIN_SUPPORT to pass an MDS-MU to a function set in the local node. The function set chosen is the one that has registered an MS application program name (via protocol boundary O), which matches the destination application program name in the MDS-MU. The signal passed is described in "Protocol Boundary R--Send MDS-MU" in topic 2.9.1.13.

The creation of this signal is described in "Routing MDS-MU Received to Local MS Application Programs" in topic 2.9.2.4.8.

- Internal MS protocol boundary S (Forward CP Status)

This protocol boundary is used to pass CP status information to MS_CAPS after initialization via protocol boundary U. The signal passed is described in "Protocol Boundary S--Forward CP Status" in topic 2.9.1.14.

The creation of this signal is described in "Preparing Forward CP

Status" in topic 2.9.2.4.8.

- Internal MS protocol boundary T (Register MS Category)

In the process of initialization, management services application programs register themselves with the MS_CAPS function set group via protocol boundary T if they want focal point information. The application program passes its name and the category for which it is registering. The details of this protocol boundary are described in "Protocol Boundary T--Register MS Category" in topic 2.9.1.15.

The process by which an application program registers itself with MS_CAPS is described in "Application Program Registration with MS_CAPS" in topic 2.9.3.10.

The MDS_ROUTER maintains awareness of the Alert focal point so that Alerts received from back-level nodes can be addressed and forwarded.

The creation of this signal is described in "Back-level Node Support Preparation of Register MS Category" in topic 2.9.2.4.8.

2.9.2.2 Prerequisite Function Sets

See "Role Requirements for Management Services Components" in topic 2.8.1 for information on the relationships between this function set and other function sets.

2.9.2.3 Overview of Subsets

```
 ____________ _____________ _____________ _____________
| optional   | optional    | optional    | optional    |
| subset 1   | subset 2    | subset 3    | subset 4    |
|            |             |             |             |
| End Node   | Network Node |    High     | Transport   |
| Support    | Support     | Performance | Confirmation |
|            |             |   Option    |   Option    |
|____________|_____________|_____________|_____________|
|                                                      |
|                MULTIPLE_DOMAIN_SUPPORT               |
|                   common base subset                 |
|                                                      |
|______________________________________________________|
```

Figure 9-3. Base and Optional Subsets of MULTIPLE_DOMAIN_SUPPORT Function Set

These subsets are described in the following sections:

"MULTIPLE_DOMAIN_SUPPORT Base Subset" in topic 2.9.2.4

"MULTIPLE_DOMAIN_SUPPORT Optional Subset 1 (End Node Support)" in topic 2.9.2.5

"MULTIPLE_DOMAIN_SUPPORT Optional Subset 2 (Network Node Support)" in topic 2.9.2.6

"MULTIPLE_DOMAIN_SUPPORT Optional Subset 3 (High-Performance Option)" in topic 2.9.2.7

"MULTIPLE_DOMAIN_SUPPORT Optional Subset 4 (Transport Confirmation Option)" in topic 2.9.2.8

Subtopics:

- 2.9.2.3.1 End Node Role Requirements
- 2.9.2.3.2 Network Node Role Requirements

2.9.2.3.1 End Node Role Requirements

APPN end nodes must implement the following:

```
  °   MULTIPLE_DOMAIN_SUPPORT common base subset

| °   End node support (optional subset 1)
```

| 2.9.2.3.2 Network Node Role Requirements

```
| Network nodes must implement the following:

| °   MULTIPLE_DOMAIN_SUPPORT common base subset

| °   Network node support (optional subset 2)
```

2.9.2.4 MULTIPLE_DOMAIN_SUPPORT Base Subset

Subtopics:

- 2.9.2.4.1 Functions Provided
- 2.9.2.4.2 Implementation Requirements
- 2.9.2.4.3 Formats Supported
- 2.9.2.4.4 A Subset of Management Services
- 2.9.2.4.5 Subset of Control Point Session Services
- 2.9.2.4.6 Subset of Control Point Directory Services
- 2.9.2.4.7 Subset of Control Point and LU Presentation Services
- 2.9.2.4.8 MDS Router Processing
- 2.9.2.4.9 Overview of MDS Transaction Programs
- 2.9.2.4.10 MDS_SEND Processing
- 2.9.2.4.11 MDS_RECEIVE Processing
- 2.9.2.4.12 MSU_HANDLER Processing
- 2.9.2.4.13 MDS_SON Processing

2.9.2.4.1 Functions Provided

```
MULTIPLE_DOMAIN_SUPPORT base subset is required in both APPN end nodes and
APPN network nodes.  The functions provide the following support:

  °   MDS_SEND, MDS_RECEIVE, and MDS_SON transaction programs

  °   Support for back-level nodes (MSU_HANDLER transaction program and
      mapping between MDS-MUs and CP-MSUs.)

  °   Sending and receiving MDS-MUs over CPSVCMG sessions (between end nodes
      and their network node servers only) and over SNASVCMG LU-LU sessions
      (to any destination).

  °   Routing MDS-MUs to MS application programs in the local node

  °   Correlating MDS requests and replies based on Agent Unit-of-Work
      Correlator (X'1549') GDS variable

  °   Receiving and processing MDS error messages

  °   Sending MDS error messages for errors detected

  °   Sending and receiving MDS-MUs up to the maximum size (31743 bytes +
      header information).
```

2.9.2.4.2 Implementation Requirements

```
The MULTIPLE_DOMAIN_SUPPORT function set is represented by a model
consisting of the following:

  °   A subset of CP management services and its service transaction
      programs
  °   A subset of CP session services
| °   A subset of CP directory services
  °   A subset of CP and LU presentation services
```

2.9.2.4.3 Formats Supported

MULTIPLE_DOMAIN_SUPPORT supports the following formats:

- MDS-MU (X'1310') GDS variable
- MDS Routing Information (X'1311') GDS variable
- Agent Unit-of-Work Correlator (X'1549') GDS variable
- SNA Condition Report (X'1532') GDS variable

The following format is supported by the MULTIPLE_DOMAIN_SUPPORT function set as an envelope for MS application program data. Other GDS variable variables for MS application program data are also supported implicitly by the MDS function set (i.e., no explicit statement of support is required for their use).

- CP-MSU (X'1212') GDS variable

See "Overview of Multiple-Domain Support" in topic 1.2.3.2.1 for further information.

2.9.2.4.4 A Subset of Management Services

```
°°°
```

```
|     ____________      Register Application Name (O)    ____________
|    |            |____________________________________|            |
|    |            |    MDS-MU Received (Q)             |     MS     |
|    |            |____________________________________ÿ| application |
|    |            |    Send MDS-MU (R)                 |            |
|    |            |____________________________________|            |
|    |            |                                    |____________|
|    |            |    Register Application Name (O)    ____________
|    |            |____________________________________|            |
|    |            |    Send MDS-MU (R)                 |            |
|    |            |____________________________________|            |
|    |            |    MDS-MU Received (Q)             |            |
|    |            |____________________________________ÿ|            |
|    |            |    MDS Error Message Info. (N)     |            |
|    |            |____________________________________ÿ|            |
|    |            |    Register MS Category (T)        |   MS_CAPS  |
|    |            |____________________________________ÿ|            |
|    |            |    Current Focal Point (V)         |            |
|    | MDS_ROUTER |____________________________________|            |
|    |            |    Begin Sending CP Status (U)     |            |
|    |            |____________________________________|            |
|    |            |    Forward CP Status (S)           |            |
|    |            |____________________________________ÿ|            |
|    |            |                                    |____________|
|    |            |                                     ____________
|    |            |    CP_STATUS           (via PB 11)|            |
|    |            |____________________________________|            |
|    |            |                                    |     CP     |
|    |            |    ASSIGN_PCID         (via PB 11)|   Session  | _____
|    |            |____________________________________ÿ|  Services  |   "
|    |            |    ASSIGN_PCID_RSP     (via PB 11)|            |   |
|    |            |____________________________________|            | Optional
|    |            |                                    |____________| subset 2
|    |            |    LOCATE_MESSAGE(RQ)   (via PB 20) ____________  only
|    |            |____________________________________ÿ|     CP     |   |
|    |            |    LOCATE_MESSAGE(RPLY) (via PB 20)|  Directory |   |
|    |            |____________________________________|  Services  |
|    |____________|                                    |____________| _____
```

Figure 9-5. MDS External Protocol Boundaries. The MDS_ROUTER serves the external protocol boundaries between MDS and other MS function sets, as well as the protocol boundaries to other node components required by MDS.

Figure 9-5 illustrates the protocol boundaries between the MDS_ROUTER and those components of the control point external to the MS subset that implements MDS.

The following protocol boundaries are *internal* to the subset of CP management services that implements the MULTIPLE_DOMAIN_SUPPORT function set. These are the signal exchanged between the MDS router and the MDS TPs, illustrated in Figure 9-4.

MDS Internal Signal - MDS Routing Unit

Description	Format for passing MDS-MUs between MDS_ROUTER internal processes and between MDS_ROUTER and MDS transaction programs (MDS_SEND, MDS_RECEIVE, and MSU_HANDLER). **Note:** With the exception of the MDS-MU itself, the fields in this signal are used internally by MULTIPLE_DOMAIN_SUPPORT and are not transmitted out of the node.
Data Content	1. Direction ° INBOUND ° OUTBOUND ° REPLY TO OUTBOUND 2. Type of routing requested ° DIRECT ° DEFAULT 3. Routing status ° INCOMPLETE ° COMPLETE ° ERROR 4. Sense data (Note 2) 5. MDS error message required (Note 2) ° ORIGIN ° DESTINATION ° BOTH ORIGIN AND DESTINATION ° NO (Note 3) 6. Alert required (Note 2) ° YES ° NO 7. Back-level node reply correlation required (Note 4) ° YES ° NO 8. Inbound session information (note 5) 9. Outbound session information (note 5) 10. Origin Location Name (from X'81' subvector) ° NETID ° NAU

```
|              |     °   MS application program                          |
|              | 11. Destination Location Name (from X'82' subvector)    |
|              |     °   NETID                                           |
|              |     °   NAU                                             |
|              |     °   MS application program                          |
|              | 12. Flag settings (from X'90' subvector)                |
|              |     °   MDS Message Type                                |
|              |     °   First MDS Message Indicator                     |
|              |     °   Last MDS Message Indicator                      |
|              | 13. Agent Unit-of-Work Correlator (X'1549') GDS         |
|              |     variable (from MDS-MU)                              |
|              | 14. Complete MDS-MU (CP-MSU if inbound from back-level  |
|              |     node)                                               |
|______________|_________________________________________________________|
```

Notes:

1. All values are null when unknown or not applicable.

2. Defined only when routing status is ERROR.

3. This value is used when no MDS error message is possible, such as for parsing errors on the MDS-MU itself, but an Alert can be created to report the error.

4. Defined only when direction is INBOUND and MULTIPLE_DOMAIN_SUPPORT level of inbound session partner is BACK-LEVEL. A YES value indicates that the inbound message is a reply from a back-level node which must be correlated with an outstanding request.

5. Session information (inbound or outbound) consists of the following fields:

   a. Session partner Network-qualified NAU (CP or LU) name
   b. Session mode name
      - CPSVCMG
      - SNASVCMG
   c. Session CGIDs
      - Contention-winner session
      - Contention-loser session

      The mapping of CGIDs to session type (contention-winner or contention-loser) is significant only when mode name is CPSVCMG (i.e., for a CP-CP session)
   d. MULTIPLE_DOMAIN_SUPPORT level of session partner
      - UNKNOWN
      - BACK-LEVEL (i.e., a back-level node)
      - CURRENT

MDS Internal Signal - SNASVCMG SON (Session-Outage Notification)

```
 _____________ _____________________________________________________
| Origin      | MDS_SON                                             |
|_____________|_____________________________________________________|
| Destination | MDS_ROUTER                                          |
|_____________|_____________________________________________________|
| Data Content | 1.  Session partner Network-qualified LU name      |
|              | 2.  Session mode name                              |
|              |     °   SNASVCMG                                   |
|              | 3.  Session CGID                                   |
|              | 4.  Session type                                   |
|              |     °   Contention-winner                          |
|              |     °   Contention-loser                           |
|              | 5.  Sense data associated with session outage      |
|_____________|_____________________________________________________|
| Initiates   | Process described in "Processing SNASVCMG SON" in   |
|             | topic 2.9.2.4.8                                     |
|_____________|_____________________________________________________|
```

2.9.2.4.5 Subset of Control Point Session Services

The following signals are exchanged with CP session services over external protocol boundary 11, as illustrated in Figure 9-5 in topic 2.9.2.4.4.

CP_STATUS Signal

```
 _____________ _____________________________________________________
| Origin      | CP Session Services                                 |
|_____________|_____________________________________________________|
```

```
| Destination  | MDS_ROUTER                                              |
|______________|_________________________________________________________|
| Data Content | 1.  Network-qualified CP name (of adjacent node)        |
|              | 2.  Session status                                      |
|              |     °   INACTIVE                                        |
|              |     °   ACTIVE                                          |
|              | 3.  Session type                                        |
|              |     °   Contention-winner                               |
|              |     °   Contention-loser                                |
|              | 4.  Session CGID                                        |
|              | 5.  Authorized                                          |
|              |     °   YES                                             |
|              |     °   NO                                              |
|              | 6.  Adjacent node CP_CAPABILITIES GDS variable          |
|              |                                                         |
|              |     Present when                                        |
|              |         Status=ACTIVE                                   |
|              |         Session type=Contention-winner.                 |
|              |                                                         |
|              |     Byte 8 bit 6 of this GDS variable indicates whether |
|              |     or not MS Capabilities exchange is supported by the |
|              |     adjacent node.                                      |
|______________|_________________________________________________________|
| Initiates    | Process described in "Processing CP_STATUS Signal" in   |
|              | topic 2.9.2.4.8.                                        |
|______________|_________________________________________________________|
```

```
| Additional signals are exchanged with CP session services as part of
| "MULTIPLE_DOMAIN_SUPPORT Optional Subset 2 (Network Node Support)" in
| topic 2.9.2.6.
```

| 2.9.2.4.6 Subset of Control Point Directory Services

```
| Interaction with CP directory services is required only as part of
| "MULTIPLE_DOMAIN_SUPPORT Optional Subset 2 (Network Node Support)" in
| topic 2.9.2.6.
```

2.9.2.4.7 Subset of Control Point and LU Presentation Services

```
The MULTIPLE_DOMAIN_SUPPORT TPs interact with CP and LU presentation
services over external protocol boundary 21 with the following LU 6.2
verbs:

1.  DEFINE_MODE

2.  ALLOCATE

3.  DEALLOCATE

4.  SEND_DATA

5.  RECEIVE_AND_WAIT

6.  GET_ATTRIBUTES

7.  CONFIRM

8.  CONFIRMED

Refer to "Overview of MDS Transaction Programs" in topic 2.9.2.4.9 for an
overview of the transaction programs, and illustrations of the verb
sequences.
```

2.9.2.4.8 MDS Router Processing

```
| °°°°
```

```
| °  || CP Status Info.' ||       ||   Session List   ||       | Active Transaction | |     °
| °  ||__________________||       ||__________________||       | List               |_|     °
| °    ________________            ________________            |____________________|       °
| °  ||    Processing    ||       ||  Processing of   ||                                   °
| °  ||   'CP_STATUS'     ||       ||     CPSVCMG      ||         ____________________      °
| °  ||                  ||       ||   Session List   ||       ________________    |       °
| °  ||__________________||       ||__________________||       | CPSVCMG Session    | |     °
| °    ________________            ________________            | List               |_|     °
| °  ||    Processing    ||       ||  Processing of   ||       |____________________|       °
| °  ||  'Send MDS-MU'    ||       ||      Active      ||                                   °
| °  ||                  ||       || Transaction List ||           ____________________      °
| °  ||__________________||       ||__________________||       ________________    |       °
| °    ________________            ________________            | SNASVCMG Session   | |     °
| °  ||    Processing    ||       ||  Processing of   ||       | List               |_|     °
| °  ||     'Current     ||       ||Local Application ||       |____________________|       °
| °  ||   Focal Point'    ||       ||   Routing List   ||                                   °
| °  ||__________________||       ||__________________||                                   °
| °    ________________            ________________                                         °
| °  ||    Processing    ||       ||    MDS Header    ||                                   °
| °  ||'MDS Routing Unit ||       ||     Parsing      ||                                   °
| °  ||    (Reply to     ||       ||                  ||                                   °
| °  ||    Outbound)'    ||       ||__________________||                                   °
| °  ||__________________||       ________________                                         °
| °    ________________            ||  Building MDS   ||                                   °
| °  ||    Processing    ||       ||  Error Messages  ||                                   °
| °  ||'MDS Routing Unit ||       ||    and Alerts    ||                                   °
| °  ||    (Inbound)'    ||       ||__________________||                                   °
| °  ||__________________||       ________________                                         °
| °    ________________            ||  Migration Node  ||                                   °
| °  ||    Processing    ||       ||     Support      ||                                   °
| °  ||  'SNASVCMG SON'   ||       ||                  ||                                   °
| °  ||                  ||       ||__________________||                                   °
| °  ||__________________||       ________________                                         °
| °                                ||                  ||       ||    Directory     ||       °
| °                                || MDS_SEND Manager ||       ||     Services     ||       °
| °                                ||                  ||       ||     Support      ||       °
| °                                ||__________________||       ||__________________||       °
| °   M D S _ R O U T E R                                      Optional subset 2 only       °
| °                                                                                         °
| °°°°
```

Table 9-4. MDS_ROUTER High Level Processes

Process Name	Page
Processing Register Application Program Name	2.9.2.4|8
Processing Send MDS-MU	2.9.2.4|8
Processing Current Focal Point	2.9.2.4|8
Processing Begin Sending CP Status Information	2.9.2.4|8
Processing MDS Routing Unit (Inbound) from TPs	2.9.2.4|8
Processing MDS Routing Unit (REPLY TO OUTBOUND)	2.9.2.4|8
Processing SNASVCMG SON	2.9.2.4|8
Processing CP_STATUS Signal	2.9.2.4|8

These processes are described in the following sections.

Processing Register Application Program Name

Process Name	Processing Register Application Program Name
Description	This process performs the work necessary when an MS application program invokes protocol boundary O, Register Application Program Name.
Input	"Protocol Boundary O--Register Application Program Name" in topic 2.9.1.11
Output	The output from this process is one of the following: ° Local application routing table is updated. A new entry may be added, or an existing entry may be updated or deleted. ° If the MS application program name is MS_CAPS, Back-level Node Support is invoked.

Figure 9-7 illustrates the flow of this process.

```
                    (Register Application Program Name)
                                     |
                                     |

                            _________________
                           ||  Processing of  ||
                           ||Local Application ||
                           ||   Routing List  ||
                           ||_________________||
                                     |
                                     |
 _________________          _________________
||  Back-Level Node ||______|  Is Application  |
||     Support      ||  yes |   Program Name   |
||                  ||      |     MS_CAPS?     |
||__________________||      |__________________|
          |                          |no
          |                          |
          |__________________________ÿ|
                                     |

                                  (exit)
```

Figure 9-7. Processing Register Application Program Name

This processing is described in detail in the following steps.

1. "Processing of Local Application Program Routing List" is invoked.

2. If the MS application program name in the signal is MS_CAPS, "Back-level Node Support" is invoked.

3. This process ends.

Processing Send MDS-MU

Process Name	Processing Send MDS-MU
Description	This process performs the work necessary when an MS application program invokes protocol boundary R, Send MDS-MU.
Input	"Protocol Boundary R--Send MDS-MU" in topic 2.9.1.13
Output	One of the following: ° An MDS Routing Unit (OUTBOUND) is built from the Send MDS-MU signal and passed to "MDS Routing Algorithm" for routing to the intended destination. ° If "MDS Header Parsing" finds format errors in the MDS-MU, an MDS error message will be built and passed to MDS Routing Algorithm for routing back to the originating MS application program. An Alert will also be generated.

Figure 9-8 illustrates the flow of this process.

```
                              (Send MDS-MU)
                                    |
                                    |

                          _________________
                         |  convert 'Send  |
                         |  MDS-MU' to MDS |
                         |   Routing Unit  |
                         |_________________|
                                    |

 _________________        _________________
||  Building MDS  ||      ||   MDS Header  ||
||  Error Messages ||_____||    Parsing     ||
||                ||error ||                ||
||________________||      ||________________||
          |                        |
          |                        |
          |________________________ÿ|
                                   |

                          _________________
                         ||  MDS Routing  ||
                         ||   Algorithm   ||
                         ||               ||
                         ||_______________||
```

Figure 9-8. Processing Send MDS-MU

This processing is described in detail in the following steps.

1. An MDS Routing Unit (see "MDS Internal Signal - MDS Routing Unit" in topic 2.9.2.4.4) is prepared from the Send MDS-MU signal

   - Direction is set to OUTBOUND.
   - Routing status is set to INCOMPLETE
   - Type of routing requested is copied (DIRECT or DEFAULT)
   - The MDS-MU is copied.
   - All other values are set to null.

2. "MDS Header Parsing" is invoked to validate the syntax of the MDS header information.

3. If Routing Status is ERROR:

   a. "Building MDS Error Messages and Alerts" is invoked to build an MDS error message from the MDS-MU in the MDS Routing Unit and to send an Alert.

   b. The MDS Routing Unit direction is changed to INBOUND.

4. The MDS Routing Unit is passed to "MDS Routing Algorithm" for routing.

5. This process ends.

| *Processing Current Focal Point*

Process Name	Processing Current Focal Point

Description	This process performs the work necessary when MS_CAPS invokes protocol boundary V, Current Focal Point.
Input	"Protocol Boundary V--Current Focal Point" in topic 2.9.1.17
Output	The awareness of the Alert focal point kept by Back-Level Node Support will be updated.

Figure 9-9 illustrates the flow of this process.

```
                              (Current Focal Point)
                                       |
                                       |
                                 _________________
                              ||  Back-Level Node ||
                              ||     Support      ||
                              ||                  ||
                              ||__________________||
```

Figure 9-9. Processing Send Current Focal Point

This processing is described in detail in the following steps.

1. The Current Focal Point signal is passed to "Back-level Node Support."

2. This process ends.

Processing Begin Sending CP Status Information

Process Name	Processing Begin Sending CP Status Information
Description	This process performs the work necessary when MS_CAPS invokes protocol boundary U.
Input	"Protocol Boundary U--Begin Sending CP Status Information" in topic 2.9.1.16
Output	"Protocol Boundary S--Forward CP Status" in topic 2.9.1.14 will be invoked to pass information about all current CPSVCMG sessions for which CP Capabilities has been received. Subsequent updates will also be sent.

Figure 9-10 illustrates the flow of this process.

```
                              (Current Focal Point)
                                       |
                                       |
                                 _________________
                              ||  Processing of   ||
                              ||     CPSVCMG      ||
                              ||   Session List   ||
                              ||__________________||
```

Figure 9-10. Processing Begin Sending CP Status Information

This processing is described in detail in the following steps.

1. "Processing of CPSVCMG Session List" is invoked to send information on all current entries in the list to MS_CAPS.

Processing CP_STATUS Signal

Process Name	Processing CP_STATUS Signal
Description	This process performs the work necessary when CP session services sends MDS_ROUTER the CP_STATUS signal, indicating that a CPSVCMG session has changed status.
Input	"CP_STATUS Signal" in topic 2.9.2.4.5
Output	The signal is passed to "Processing of CPSVCMG Session

	List."

Figure 9-11 illustrates the flow of this process.

```
                                (CP_STATUS)
                                     |
                                     |

 _________________        _________________
|| Processing of   ||      || Processing of  ||
||     Active      ||_____||    CPSVCMG      ||
|| Transaction List || error ||  Session List  ||
||_________________||      ||_________________||
          |                         |
          |                         |
          |_________________________ÿ|
                                    |

                                 (exit)
```

Figure 9-11. Processing CP_STATUS Signal

This processing is described in detail in the following steps.

1. The CP_STATUS signal is passed to "Processing of CPSVCMG Session List."

2. If the invoked process returns an error indication (meaning that connectivity has been lost to the CP identified in the CP_STATUS), "Processing of Active Transaction List" is invoked. That process will determine if any outstanding transactions are impacted, and will build an MDS error message for each.

3. This process ends.

Processing MDS Routing Unit (Inbound) from TPs

Process Name	Processing MDS Routing Unit (INBOUND) from TPs
Description	This process performs the required steps when an MDS Routing Unit (INBOUND) is received from either the MDS_RECEIVE or MSU_HANDLER transaction program.
Input	"MDS Internal Signal - MDS Routing Unit" in topic 2.9.2.4.4
Output	° The MDS Routing Unit is passed to MDS Routing Algorithm for routing to the intended destination. ° If "MDS Header Parsing" finds format errors in the MDS-MU, "Building MDS Error Messages and Alerts" will be invoked to build an MDS error message and an Alert. If an MDS error message can be successfully built (for some format errors it cannot be) it will be passed to MDS Routing Algorithm for routing back to the originating MS application program. ° If the MDS Routing Unit (INBOUND) was received from a back-level node, "Back-level Node Support" is invoked to perform CP-MSU to MDS-MU mapping. If that process is successful, an MDS-MU is returned, which is passed to MDS Routing Algorithm. If that process detects an error, Building MDS Error Messages and Alerts is invoked. If Back-level Node Support returns a null MDS-MU, no error reporting or routing is performed.

Figure 9-12 illustrates the flow of this process.

```
|                 (MDS Routing Unit(Inbound))
|                            |
|                            |
|
|
|                  _________________        _________________
|                 |    Is MDS Level    |      ||    Back-level    ||
|                 |    BACK-LEVEL?     |______||   Node Support   ||
|                 |                    | yes  ||                  ||
|                 |____________________|      ||__________________||
|                           | no                        |
|
|
|                  _________________        _________________
|                ||    MDS Header    ||      | Is MDS-MU pointer  |
```

```
|                   ||     Parsing      ||       |       null?        |_____
|                   ||                  ||       |                    | yes |
|                   ||__________________||       |____________________|     |
|                            |                          | no              |
|                            |__________________________|                 |
|                            |                                            |
|                                                                         |
|                   ____________________      ____________________        |
|                   | Is Routing Status  |       ||   Building MDS   ||   |
|                   |       ERROR?       |_____ÿ||  Error Messages  ||   |
|                   |                    | yes   ||    and Alerts   ||   |
|                   |____________________|       ||________________||   |
|                            | no                       |                 |
|                            |                                            |
|                            |                   ____________________     |
|                            |                   | Is MDS-MU pointer  |   |
|                            |_________________|       null?        |   |
|                            |                 no |                    |   |
|                                                |____________________|   |
|                   ____________________                  | yes          |
|                   ||   MDS Routing    ||                |              |
|                   ||    Algorithm     ||                |              |
|                   ||                  ||                |              |
|                   ||__________________||                |              |
|                            |                                            |
|                            |____________________________________________|
|                            |
|
|                         (exit)
```

Figure 9-12. Processing MDS Routing Unit (Inbound)

This processing is described in detail in the following steps.

1. If the MDS support level in the MDS Routing Unit is BACK-LEVEL:

   a. "Back-level Node Support" is invoked to map the received CP-MSU to an MDS-MU.

   b. If the invoked process returned a null MDS-MU pointer (meaning that the inbound data was discarded and no further action is required), this process exits.

2. If the MDS support level in the MDS Routing Unit is not BACK-LEVEL:

   "MDS Header Parsing" is invoked to validate the syntax of the MDS header information.

3. If the Routing Status in the MDS Routing Unit is ERROR:

   a. "Building MDS Error Messages and Alerts" is invoked to convert the MDS-MU in the MDS Routing Unit to an MDS error message and to send an Alert, if required.

   b. If the invoked process returned a null MDS-MU pointer (meaning that the data was discarded and no further action is required), this process exits.

   c. The MDS Routing Unit direction is changed to OUTBOUND.

4. The MDS Routing Unit is passed to "MDS Routing Algorithm" for routing.

5. This process ends.

Processing SNASVCMG SON

Process Name	Processing SNASVCMG SON
Description	This process performs the work necessary when MDS_SON reports the loss of an SNASVCMG session via the SNASVCMG SON signal.
Input	"MDS Internal Signal - SNASVCMG SON (Session-Outage Notification)" in topic 2.9.2.4.4
Output	The following outputs occur: ° The SNASVCMG Session List is updated by "Processing of SNASVCMG Session List."

```
|              | °   If both the contention-winner and contention-loser  |
|              |     sessions are inactive, the Active Transaction List  |
|              |     is checked for impacts.                             |
|              |                                                         |
|              |     If the outage does impact currently active          |
|              |     transactions, a "null message" will be created and  |
|              |     passed to "MDS Routing Algorithm," an action which  |
|              |     will cause session re-activation if possible.  If   |
|              |     this session reactivation attempt is unsuccessful,  |
|              |     the MDS_SEND TP will report the error back to the   |
|              |     MDS_ROUTER, causing the appropriate error           |
|              |     notification processing to take place.              |
|              |                                                         |
|              |     If the outage does not impact currently active      |
|              |     transactions, an MDS error message Information      |
|              |     signal (described in "Protocol Boundary N--MDS      |
|              |     Error Message Information" in topic 2.9.1.10) is    |
|              |     created with the UNBIND sense data and is passed to |
|              |     MS_CAPS, which will determine if recovery is        |
|              |     appropriate at the MS application program level     |
|              |     (e.g., switching to a backup focal point).          |
|______________|_________________________________________________________|
```

The SNASVCMG sessions may be nonpersistent; however, when the MDS_ROUTER has outstanding transactions routed to or through a node, at least one of the (contention-winner, contention-loser) pair of sessions must remain active. If the MDS_ROUTER allowed both sessions to become inactive while transactions were outstanding, there might be no notification that actual connectivity to the partner has been lost.

SON for the SNASVCMG sessions does *not* trigger error reporting for the outstanding transactions. The only event that triggers error reporting for all transactions with a particular session partner is the notification from the MDS_SEND TP that it cannot communicate with the partner (after all reasonable retries have been exhausted).

When both sessions with a partner become inactive and there are outstanding transactions with that partner, the MDS_ROUTER will attempt to reactivate one of the sessions. This is accomplished by starting the MDS_SEND TP with a null message to send the partner. If that action succeeds, one of the (contention-winner, contention-loser) session pair will have become active, so that the MDS_ROUTER can be assured that connectivity still exists. This is illustrated in Figure 9-52 in topic 2.9.2.4.9.

A null message in this context is an MDS Routing Unit (OUTBOUND) signal without an MDS-MU.

If the null send fails (i.e., MDS_SEND is unable to allocate a conversation), the TP will notify the MDS_ROUTER (via an MDS Routing Unit (REPLY TO OUTBOUND) containing sense data) that it cannot communicate with the partner. This notification will drive the error reporting of the MDS_ROUTER ("Processing MDS Routing Unit (REPLY TO OUTBOUND)").

If the null send succeeds (i.e., MDS_SEND is able to allocate and deallocate a conversation), the MDS Routing Unit (REPLY TO OUTBOUND) will cause the SNASVCMG Session List to be updated.

When both sessions with a partner become inactive and there are *no* outstanding transactions with that partner, the MDS_ROUTER will not attempt to reactivate the sessions. However, the loss of sessions between a SOC node and a focal point due to a connectivity failure may require that MS_CAPS initiate some recovery action, such as switching to a backup focal point. The normal deactivation of SNASVCMG sessions, which may be done by the LU to conserve system resources when the sessions are idle, does not cause MS_CAPS to initiate recovery actions. The MDS_ROUTER reports the loss of the last SNASVCMG session to MS_CAPS by creating an MDS error message information signal (described in "Protocol Boundary N--MDS Error Message Information" in topic 2.9.1.10) containing the UNBIND sense data and passes the information to MS_CAPS.

This process is illustrated in Figure 9-13.

```
              (SNASVCMG SON)
                    |
                    |

            _________________
         ||  Processing of   ||
 ________||     SNASVCMG     ||
|        ||   Session List   ||
|        ||_________________||
|one session       |     last session to
|still active      | partner is inactive
|
|           _________________
|        ||  Processing of   ||
|        ||      Active      ||_________________
|        || Transaction List || no transactions |
|        ||_________________|| impacted        |
|     transactions |                            |
|     impacted     |                            |
```

```
 |
 |                                                          °°°°°°°°°°°°°°°°°°°°°
 |        _________________        _____________________     °                   °
 |       | create MDS Routing |   |  create MDS Error   |    °                   °
 |       |  Unit (Outbound)   |   | Msg Info W/UNBIND   |_____ÿ°      MS_CAPS      °
 |       |  with null MDS-MU  |   |     sense data      | PB N °                   °
 |       |____________________|   |_____________________|     °°°°°°°°°°°°°°°°°°°°°
 |                 |                         |
 |                 |                         |
 |                                          |
 |        _________________                  |
 |       ||   MDS Routing    ||              |
 |       ||    Algorithm     ||              |
 |       ||                  ||              |
 |       ||__________________||              |
 |                 |                         |
 |_________________ÿ|_________________________|
                   |

                 (exit)
```

Figure 9-13. Processing SNASVCMG SON

When this process is started:

1. "Processing of SNASVCMG Session List" is invoked.

2. If the invoked process returns an error indication, "Processing of Active Transaction List" is invoked to see if any active transactions are impacted. If an error indication is returned:

   a. An MDS Routing Unit (OUTBOUND) is prepared with the following information:

      - Direction OUTBOUND
      - Session partner name (from SNASVCMG SON)
      - Session mode name (SNASVCMG)
      - Routing Status COMPLETE
      - No MDS-MU

      This is a "null message," which will drive session recovery attempts by MDS_SEND.

   b. The MDS Routing Unit (OUTBOUND) is passed to "MDS Routing Algorithm."

3. If no active transactions are impacted, an MDS error message information signal ("Protocol Boundary N--MDS Error Message Information" in topic 2.9.1.10) is created to notify MS_CAPS that the SNASVCMG sessions to the session partner have become inactive. The UNBIND sense data is included so that MS_CAPS can determine why the last session became inactive. It may have become inactive because of a network connectivity failure or because the LU simply shut it down normally to conserve system and network resources. In the case of a connectivity failure between a SOC node and a focal point, MS_CAPS will initiate appropriate application program-level recovery actions (such as switching to a backup focal point).

   The MDS error message information signal is built with the following information:

   - The origin is MDS_ROUTER in this node.
   - The destination is MS_CAPS in this node.
   - The SNA report code is copied from the sense data in the SNASVCMG SON signal.
   - The Reported on location is MS_CAPS in the session partner to which connectivity has been lost.
   - Session over which MDS error message was received is set to SNASVCMG mode with the session partner name.
   - Session over which MDS error message was sent is null.

4. The signal is passed to MS_CAPS.

5. This process ends.

Processing MDS Routing Unit (REPLY TO OUTBOUND)

Process Name	Processing MDS Routing Unit (REPLY TO OUTBOUND)
Description	This process performs the work necessary when an MDS Routing Unit (REPLY TO OUTBOUND) is received from MDS_SEND.

Input	"MDS Internal Signal - MDS Routing Unit" in topic 2.9.2.4.4
Output	° If session mode name is SNASVCMG, "Processing of SNASVCMG Session List" is invoked, so that the session status entry can be updated. ° If the routing status is ERROR, "Processing of Active Transaction List" is invoked, so that all active transactions impacted by the session-level detected by MDS_SEND can be processed (MDS error messages generated).

The MDS_SEND TP is invoked when "MDS Routing Algorithm" passes it an MDS Routing Unit (OUTBOUND) signal. When the TP has completed its processing of that signal, it passes back an MDS Routing Unit (REPLY TO OUTBOUND) signal to the MDS_ROUTER. That signal causes this process to be started.

There are two pieces of information needed from the MDS Routing Unit (REPLY TO OUTBOUND):

° MDS support level of session partner
° Sense data for send failures.

This process is illustrated in Figure 9-14.

```
       (MDS Routing Unit(REPLY TO OUTBOUND))

              _________________        _________________
             |    Is mode name    |     ||  Processing of   ||
             |     SNASVCMG?      |_____ÿ||     SNASVCMG    ||
             |                    | yes  ||   Session List  ||
             |____________________|      ||_________________||
                      | no                          |
                      |_____________________________|

|             _________________        ____
|            |     Is Routing     |     | EN |
|            |   Status ERROR?    |____ÿ| or |_____________________________________
|            |                    | yes | NN? |  EN                                 |
|            |____________________|     |____|                                      |
|                      | no                | NN                                    |
|                      |                                                          |
|   ___                |        _________________                                 |
|    "                 |       | Sense X'08A8000A'  |                              |
|     |                |       |  with nonverified  |_____________________________ÿ|
|     |                |       |     flag OFF?      |  no                          |
|     |                |       |____________________|                              |
|     |                |                 | yes                                     |
|     |                |                                                          |
|     |                |        _________________        _________________        |
|     |                |       ||    Directory    ||       ||  Building MDS    ||  |
|     |                |       ||     Services    ||_______ÿ|| Error Messages   ||  |
|     |                |       ||     Support     || ERROR  ||   and Alerts     ||  |
| (NN only)            |       ||_________________||       ||_________________||  |
|     |                |                 | COMPLETE                  |            |
|     |                |                                                          |
|     |                |        _________________        _________________        |
|     |                |       ||                 ||       ||   MDS Routing    ||  |
|     |                |       || MDS_SEND Manager ||       ||    Algorithm    ||  |
|     |                |       ||                 ||       ||                 ||  |
|     |                |       ||_________________||       ||_________________||  |
|                     |_________________|                           |            |
|   ___                |_____________________________________________|            |
|                      |                                                          |
|                      |                        _________________                 |
|                      |                       ||  Processing of   ||             |
|                      |_______________________||     Active      ||_____________|
|                      |                       || Transaction List ||
|                      |                       ||_________________||
|
|   ___        _________________        _________________
|    "        |Is Routing Status   |     | Update MDSRU with  |
|     |       |INCOMPLETE with     |_____ÿ|partner=destination |
|     |       |sense X'08A0 0007'? | yes  |   mode=SNASVCMG    |
|     |       |____________________|      |  status=COMPLETE   |
|     |                | no              |____________________|
|     |                |
|     |                |                        _________________
|     |                |                       ||  Processing of   ||
|     |                |                       ||     SNASVCMG    ||
| (NN only)            |                       ||   Session List  ||
|     |                |                       ||_________________||
|     |                |
|     |                |                        _________________
|     |                |                       ||                 ||
|     |                |                       || MDS_SEND Manager ||
|     |                |                       ||                 ||
|     |                |                       ||_________________||
|                     |_________________________|
|   ___
|                     (exit)
```

Figure 9-14. Processing MDS Routing Unit (REPLY TO OUTBOUND)

Following are the steps taken when this process is started:

1. If Session Mode name is SNASVCMG, "Processing of SNASVCMG Session List" is invoked so that session information can be recorded.

2. If the Routing Status is ERROR (meaning that the MDS_SEND TP was unable to successfully send the MDS-MU to the session partner):

| a. If this is a network node, the sense data in the MDS Routing Unit (REPLY TO OUTBOUND) is examined. If the sense data is X'08A8 000A' (unrecoverable session failure), *and* if the routing verification indicator (byte 3 bit 2) of the Flags (X'90') subvector in the MDS Routing Information (X'1311') GDS variable is 0 (MDS-MU was routed based on nonverified directory information), the nonverified information from the APPN directory was invalid, and the MDS_SEND TP was unable to allocate a conversation with the network node server CP name that came from the directory. An attempt will be made to reroute the MDS-MU by requesting that directory services perform a verified search.

| 1) The direction in the MDS Routing Unit is changed to OUTBOUND.

| 2) "Directory Services Processing" in topic 2.9.2.6.2 is invoked. The ERROR Status will cause that process to force a verified search by directory services.

| 3) If Directory Services Processing returns the MDS Routing Unit with a status of COMPLETE (indicating that the verified search located the destination end node):

| a) "MDS_SEND Manager" is invoked to send the MDS-MU again.

| b) This process exits.

| 4) If Directory Services Processing returns the MDS Routing Unit with a status of ERROR (indicating that the verified search did not locate the destination end node):

| a) "Building MDS Error Messages and Alerts" is invoked to build an MDS error message with the sense data returned by Directory Services Processing.

| b) "MDS Routing Algorithm" is invoked to route the MDS error message.

| c) This process exits.

b. "Processing of Active Transaction List" is invoked. That process will search the Active Transaction List for any outstanding transactions that are impacted by the session-level error detected by MDS_SEND. For each impacted transaction (including the MDS Routing Unit (REPLY TO OUTBOUND) which triggered this error processing), an MDS error message will be built and routed.

| 3. If Routing Status is INCOMPLETE *and* the sense data contains the value X'08A8 0007', then this MDS Routing Unit was returned from MDS_SEND because the destination of the MDS-MU was an end node, but the network node server through which the MDS-MU was to be routed was a back-level node (unable to perform MDS routing). This MDS-MU will be rerouted *directly* to the destination end node.

| The MDS Routing Unit is completed as follows:

| a. Session Partner Name is set to the destination location name in MDS Routing Information (X'1311') GDS variable.

| b. Session Mode Name is set to SNASVCMG.

| c. The Routing Verification Indicator in the Flags (X'90') subvector of the MDS Routing Information (X'1311') GDS variable is set to 1.

| d. Routing Status is set to COMPLETE.

| e. "Processing of SNASVCMG Session List" is invoked to complete the session information.

| f. The MDS Routing Unit is passed to "MDS_SEND Manager."

4. This process ends.

MDS_ROUTER Specialized Processes:

Table 9-5. MDS_ROUTER Specialized Processes

Process Name	Page
MDS Routing Algorithm	2.9.2.4|8
MDS Header Parsing	2.9.2.4|8
Building MDS Error Messages and Alerts	2.9.2.4|8
Processing of Active Transaction List	2.9.2.4|8
Processing of Local Application Program Routing List	2.9.2.4|8
Processing of SNASVCMG Session List	2.9.2.4|8
Processing of CPSVCMG Session List	2.9.2.4|8
Back-level Node Support	2.9.2.4|8
MDS_SEND Manager	2.9.2.4|8
Directory Services Processing (Optional subset 2)	2.9.2.6|2

Note: Unless otherwise noted, all processes are part of the common base subset.

MDS Routing Algorithm

Process Name	MDS Routing Algorithm
Description	This process performs the work necessary to determine what the next step will be in the transport of an MDS-MU toward its destination.
Input	The input to this process is an MDS Routing Unit (see "MDS Internal Signal - MDS Routing Unit" in topic 2.9.2.4.4 for a description of the fields in that signal).
Output	The output from this process is one of the following: ° MDS Routing Unit passed to "Routing MDS-MU Received to Local MS Application Programs" for delivery to a local MS application program. ° MDS Routing Unit passed to MDS_SEND for delivery to a remote node. ° The MDS-MU converted to an MDS error message. In that case, this process causes itself to be re-invoked, to process the new MDS Routing Unit. ° "Protocol Boundary N--MDS Error Message Information" in topic 2.9.1.10 is created and passed to MS_CAPS for each MDS error message processed.

Figure 9-15 illustrates the logic of this process.

Routing Algorithm Process

```
                          °°°°°°°°°°°°°°°°°°°°°°        °°°°°°°°°°°°°°°°°°°°°°
                          °  Invoking Process  °        °   MS Application   °
                          °°°°°°°°°°°°°°°°°°°°°°        °°°°°°°°°°°°°°°°°°°°°°
                                    |(MDS Routing Unit)           "
                                                          |(MDS-MU Received)
                                                          |
      ___                 ___________________
     | 2 |_______________| MDS Error Message? |            |
     |___|           yes |____________________|            |
                 ___                 | no                    |
               | 1 |_______________ÿ                        |
               |___|
                           ___________________
                          ||  Processing of   ||       ||  Processing of   ||
            ______________||      Active      ||       ||Local Application ||
```

```
              |            ERROR || Transaction List ||       ||   Routing List   ||
              |                  ||___________________||       ||___________________||
              |                           | otherwise                   "
              |                                                     |
              |                   ___________________                  |
              |                  | Is Destination NAU |_________________|
              |                  |  the local node?   | yes
              |                  |____________________|
              |                           | no
              |
              |                   ___________________
              |                  | Is routing status  |
              |                  |COMPLETE from Active|____________
              |                  | Transaction List?  | yes        |
              |                  |____________________|            |
              |                           | no                     |
              |                                                 |
  |           |                   ___________________            |
  |           |                  |  Direct Routing    |____________|______________
  |           |                  |   requested by     | yes        |              |
  |           |                  | application program|            |              |
  |           |                  |____________________|            |              |
  |           |                           |                        |              |
  |           |                            no                  |              |
  |           |                   ___________________            |              |
  |           |                  ||  Processing of   ||           |              |
  |           |_________________||     CPSVCMG      ||___________ÿ|              |
  |           |            error ||   Session List   || complete   |
  |           |                  ||___________________||           |     ___________________
  |           |                  incomplete|                       |    | Update MDS Routing |
  |           |                                              ___|___ÿ| Unit session info, |
  |           |                   ___________________         |   |    |   Routing Status   |
  |           |                  |Is this an NN or EN?|________|   |    |      COMPLETE      |
  |           |                  |____________________| EN        |    |____________________|
  |           |                           | NN                     |              |
  |           |                                                 |
  |           |                   ___________________   (1)      |     ___________________
  |           |                  ||  Processing of   ||           |    ||  Processing of   ||
  |           |                  ||    SNASVCMG      ||___________ÿ|    ||     SNASVCMG     ||
  |           |                  ||   Session List   || complete   |    ||   Session List   ||
  |           |                  ||___________________||           |    ||___________________||
  |                                     | incomplete                |              |
  |  ________________                                              |
  | ||   Building MDS   ||        ___________________   (1)      |     ___________________
  | ||  Error Messages  ||       ||    Directory      ||           |___ÿ||                   ||
  | ||    and Alerts    ||______||     Services      ||_________________ÿ|| MDS_SEND Manager ||
  | ||__________________|| error ||     Support       || complete        ||                   ||
  |           |                  ||___________________||                ||___________________||
  |                                                                                |
  |          ___                 Note:  (1) Network Node only
  |         | 2 |                                                                (exit)
  |         |___|
```

Figure 9-15. MDS Routing Algorithm (Part 1 of 2)

The process is described in more detail below:

1. If the MDS Routing Unit contains an MDS error message, the special processing for MDS error message is performed (see "Special Handling for MDS Error Messages")

2. "Processing of Active Transaction List" is invoked. That process will supply reverse routing information for MDS replies.

3. If the Destination Location Name is the local node, the MDS Routing Unit is passed to "Routing MDS-MU Received to Local MS Application Programs."

4. If Routing Status in the MDS Routing Unit is COMPLETE from Processing of Active Transaction List, then:

   a. The Routing Verification Indicator in the Flags (X'90') subvector of the MDS Routing Information (X'1311') GDS variable is set to 1.

   b. The MDS Routing Unit is passed to "MDS_SEND Manager."

   c. This process ends.

5. if Routing Status in the MDS Routing Unit is ERROR, then:

   a. "Building MDS Error Messages and Alerts" is invoked.

   b. This process is re-invoked with the MDS Routing Unit which now contains an MDS error message ("Routing Algorithm Process").

6. If Type of Routing Requested is DIRECT, the MDS Routing Unit is completed as follows:

a. Session Partner Name is set to the destination location name in MDS Routing Information (X'1311') GDS variable.

b. Session Mode Name is set to SNASVCMG.

c. The Routing Verification Indicator in the Flags (X'90') subvector of the MDS Routing Information (X'1311') GDS variable is set to 1.

d. Routing Status is set to COMPLETE.

e. "Processing of SNASVCMG Session List" is invoked to complete the session information.

f. The MDS Routing Unit is passed to "MDS_SEND Manager."

g. This process ends.

7. "Processing of CPSVCMG Session List" is invoked to determine if the MDS-MU should be routed over a CPSVCMG session.

8. If Routing Status in the MDS Routing Unit is COMPLETE from "Processing of CPSVCMG Session List," then:

a. The Routing Verification Indicator in the Flags (X'90') subvector of the MDS Routing Information (X'1311') GDS variable is set to 1.

b. The MDS Routing Unit is passed to "MDS_SEND Manager."

c. This process terminates.

9. if Routing Status in the MDS Routing Unit is ERROR, then:

a. "Building MDS Error Messages and Alerts" is invoked.

b. This process is re-invoked with the MDS Routing Unit which now contains an MDS error message ("Routing Algorithm Process").

10. If Routing Status is INCOMPLETE, and if this is an end node, the MDS Routing Unit is completed as follows:

a. Session Partner Name is set to the destination location name in MDS Routing Information (X'1311') GDS variable.

b. Session Mode Name is set to SNASVCMG.

c. The Routing Verification Indicator in the Flags (X'90') subvector of the MDS Routing Information (X'1311') GDS variable is set to 1.

d. Routing Status is set to COMPLETE.

e. "Processing of SNASVCMG Session List" is invoked to complete the session information.

f. The MDS Routing Unit is passed to "MDS_SEND Manager."

g. This process ends.

Note

For an end node, this is the final step of the routing algorithm. The steps that follow are for network nodes only.

11. "Processing of SNASVCMG Session List" is invoked to determine if an SNASVCMG session is already active to the destination.

12. If Routing Status in the MDS Routing Unit is COMPLETE from Processing of SNASVCMG Session List, then:

a. The Routing Verification Indicator in the Flags (X'90') subvector of the MDS Routing Information (X'1311') GDS variable is set to 1.

b. The MDS Routing Unit is passed to "MDS_SEND Manager."

```
|     c.  This process terminates.

| 13. "Directory Services Processing" in topic 2.9.2.6.2 is invoked.  (The
|     Routing Verification Indicator in the Flags (X'90') subvector of the
|     MDS Routing Information (X'1311') GDS variable is set by the invoked
|     process.)

| 14. If Routing Status in the MDS Routing Unit is COMPLETE, the MDS Routing
|     Unit is passed to "MDS_SEND Manager."

  15. If Routing Status in the MDS Routing Unit is ERROR then:

      a.  "Building MDS Error Messages and Alerts" is invoked.

      b.  This process is reinvoked with the MDS Routing Unit which now
          contains an MDS error message ("Routing Algorithm Process").

  16. This process terminates.
```

Special Handling for MDS Error Messages

```
|                                        ___
|                                       | 2 |
|                                       |___|
|                                         |
|                                 (MDS Error Message
|                                in MDS Routing Unit)
|                                         |
|
| _________________ (1)     _________________ (1)
| | Build MDS Routing |     |   Is Direction  |
| | Unit (OUTBOUND),  |     | INBOUND, w/SNACR |
| |status ERROR, sense |_____|  sense data of  |_____
| | data X'08A8 0009', |  yes | X'08A8 0009', with | no  |
| | MDS-MU copied from |     |  copy of original |     |
| | MDS Error Message  |     |     MDS-MU?       |     |
| |___________________|     |_________________|     |
|          |                                          |
|                                                    |
| _________________       _________________          |
| ||   Directory    ||     ||  Building MDS  ||      |
| ||    Services    ||_____ÿ|| Error Messages ||      |
| ||    Support     || ERROR ||   and Alerts   ||      |
| ||_______________||     ||_______________||      |
|          | COMPLETE                |               |
|          |                         |_____________|
|
| _________________       _________________
| ||               ||     |   Is origin     |
| || MDS_SEND Manager ||  ____|  application    |
| ||               ||  | no |   program of    |
| ||_______________||  |    | MDS Error Message |
|          |           |    |   = MDS_ROUTER  |
|          |           |    | and destination &ne.  |
|          |           |    |  local MS_CAPS? |
|          |           |    |_________________|
|          |           |            | yes
|          |           |
|          |           |    _________________
|          |           |    |     Build       |
|          |           |    | MDS Error Message |
|          |           |    |   Information   |
|          |           |    |_________________|
|          |           |            |
|          |           |
|          |           |    _________________        °°°°°°°°°°°°°°°°°°°°°
|          |           |    |                |        °                   °
|          |           |    | Send to MS_CAPS |_____ÿ°      MS_CAPS       °
|          |           |    |                | PB N   °                   °
|          |           |    |_________________|        °°°°°°°°°°°°°°°°°°°°°
|          |           |            |
|          |           |____________ÿ|
|          |
|                                   ___
|         (exit)                   | 1 |
|                                   |___|
| Note:  (1) Network Node only
```

Figure 9-16. MDS Routing Algorithm (Part 2 of 2)

Figure 9-16 illustrates the special processing for MDS error messages, which is described in the following steps:

```
| 1.  (Network nodes only)
```

| If the following are all true:

| ° Direction of the MDS Routing Unit is INBOUND (i.e., received over
| a session)

| ° Sense Data (X'7D') subvector of the SNA Condition Report (X'1532')
| GDS variable contains the value X'08A8 0009' (Destination not
| supported by reporting network node)

| ° The MDS error message contains a copy of the original MDS-MU

| Then the original MDS-MU was routed based on nonverified directory
| services information, and the reporting network node has indicated
| that the information is no longer correct. This network node must
| force a verified search by directory services so that the MDS-MU can
| be rerouted. This routing routing error case is described in "Routing
| Error: Incorrect Cached Directory Information" in topic 2.9.2.6.2.

| The following is done:

| a. An MDS Routing Unit is built:

| ° Direction: OUTBOUND
| ° Routing Status: ERROR
| ° Sense data: X'08A8 0009'
| ° MDS-MU: taken from MDS error message

| b. "Directory Services Processing" in topic 2.9.2.6.2 is invoked.
| The ERROR Status and Sense data of X'08A8 0009' will cause that
| process to force a verified search by directory services.

| c. If Directory Services Processing returns the MDS Routing Unit with
| a status of COMPLETE (implying that the verified search located
| the destination end node):

| 1) "MDS_SEND Manager" is invoked to send the MDS-MU again.

| 2) This process exits.

| d. If Directory Services Processing returns the MDS Routing Unit with
| a status of ERROR (meaning that the verified search did not locate
| the destination end node):

| "Building MDS Error Messages and Alerts" is invoked to build an
| MDS error message with the sense data returned by Directory
| Services Processing.

| 2. An MDS Error Message Information signal (see "Protocol Boundary N--MDS
| Error Message Information" in topic 2.9.1.10) is prepared from the MDS
| error message.

3. If the following are true:

° The Application Program ID (X'03') subfield of the Origin Location
Name (X'81') subvector contains the value X'23F0F1F0' (MDS_ROUTER)

° The destination of the MDS-MU is *not* MS_CAPS (X'23F0F1F1') in this
node.

an MDS error message information signal (see "Protocol Boundary N--MDS
Error Message Information" in topic 2.9.1.10) is prepared from the MDS
error message. If the MDS error message originated from an MS
application program and not from an MDS_ROUTER, it is reporting an
error at the application program level, not a routing or connectivity
problem. MS_CAPS is not concerned with such errors, so those MDS
error messages are not passed to it.

The MDS error message information signal is built:

° The origin is copied from the Origin Location Name (X'81')
subvector in the MDS Routing Information (X'1311') GDS variable.
° The destination is copied from the Destination Location Name
(X'82') subvector in the MDS Routing Information (X'1311') GDS
variable.
° The SNA report code is copied from the Sense Data (X'7D')
subvector in the SNA Condition Report (X'1532') GDS variable.
° The Reported on location is copied from the Reported On Location
Name (X'09') subvector and the Reported on Agent (X'0B') subvector
in the SNA Condition Report (X'1532') GDS variable.
° Session over which MDS error message was received is copied from
the Inbound session information in the MDS-MU (these value may be

null if not applicable).

- Session over which MDS error message was sent is copied from the Outbound session information in the MDS-MU (these value may be null if not applicable).

4. The signal is passed to MS_CAPS.

5. The MDS error message is passed back to step 2 for routing.

Processing of Local Application Program Routing List

Process Name	Processing of Local Application Routing List
Description	This process builds and maintains the application routing table and performs local application routing for inbound MDS Routing Units.
Input	The input to this process is one of the following: ° "Protocol Boundary O--Register Application Program Name" in topic 2.9.1.11 ° "MDS Internal Signal - MDS Routing Unit" in topic 2.9.2.4.4
Output	The output from this process is one of the following: ° Local Application Routing List is updated. - A new entry may be added to the list. - An existing entry may be updated. - An existing entry may be deleted. ° MDS Routing Unit is converted to Protocol Boundary O--MDS-MU Received for delivery to a local MS application program.

This process builds and maintains the application program routing table. Application program names are added or deleted from the table via the Register Application Program Name signal (protocol boundary O) from MS application programs. Processing that signal is described in "Processing Register Application Program Name."

MS application programs issue "Protocol Boundary O--Register Application Program Name" in topic 2.9.1.11) so that MULTIPLE_DOMAIN_SUPPORT is able to map an application program name to the internal process actually implementing the application program. The information kept in the Local Application Program Routing List is described in Table 9-6.

Table 9-6. Contents of Local Application Program Routing List

Field Name	Description of Contents
MS application program name	This is the application program name that appears in an MDS Routing Information (X'1311') GDS variable.

This process may also be invoked with an MDS Routing Unit (INBOUND) signal from "MDS Routing Algorithm."

The function of this process is illustrated in Figure 9-17.

Figure 9-17. Processing of Local Application Routing List (Part 1 of 3)

1. If the input signal is Register Application Program Name, the processing on page 2.9.2.4.8 is done.

2. If the input signal is an MDS Routing Unit, the processing on page 2.9.2.4.8 is done.

Local Application Program Routing List Processing for Register Application Program Name

```
                                  ___
                                 | 1 |
                                 |___|
                                   |
                        (Register Application Name)
                                   |
                                   |

                          ___________________
                         |    search Appl.   |
          ________________|  Routing List for |
         |        duplicate |  duplicate name  |
         |                 |___________________|
         |                          | No duplicate

 ___________________       ___________________
|       update      |     |   if ACTION=ADD   |
|   existing entry  |     |  create new entry |
|                   |     |                   |
|___________________|     |___________________|
         |                          |

     (exit error)              (exit normal)
```

Figure 9-18. Processing of Local Application Routing List (Part 2 of 3)

When protocol boundary O is invoked, the following processing takes place. This process is illustrated in Figure 9-18.

1. The Application Program Routing List is searched for an entry with a matching MS application program name.

2. If a matching entry is *not* found, the following is done:

   a. If ACTION = ADD, a new entry is inserted into the list with the following information:

      - MS application program name copied from the Register Application Program Name signal

      - Internal process ID copied from the Register Application Program Name signal

   b. If ACTION = DELETE, no action is taken.

3. If a matching entry is found, the following is done:

   a. If ACTION = ADD, the entry is updated with the internal process ID copied from the Register Application Program Name signal.

```
 ___ Implementation Note ____________________________________
|                                                            |
| The architecture model does not discover MS application    |
| program failures except as interprocess communication      |
| failures.  If an MS application program process fails and is |
| reinitialized, the MDS_ROUTER will accept reregistration of |
| the application program name.                              |
|                                                            |
| Implementations may manage this interaction between the    |
| MDS_ROUTER and the MS application programs differently.    |
|                                                            |
|____________________________________________________________|
```

   b. If ACTION = DELETE, the entry is deleted from the list.

Routing MDS-MU Received to Local MS Application Programs

```
                                   ___
                                  | 2 |
                                  |___|
                                    |
                          (MDS Routing Unit (Inbound))
                                    |

                            _________________
                           |   search Appl.    |
              _____________|  Routing List for |
             |        error |   matching name   |
             |              |___________________|
             |                      | match

 _________________          _________________
|Update MDS Routing |       |   convert MDS     |
| Unit w/sense data |       |  Routing Unit to  |
| X'08A8 0003', set |       | 'MDS-MU Received' |
|  Status to ERROR  |       |___________________|
|___________________|                |
          |
          |                                          °°°°°°°°°°°°°°°°°°°°°°
          |                 _________________        °                    °
          |                |      send         |      °   MS Application   °
          |                | 'MDS-MU Received' |______ÿ°                  °
          |                | to MS application |      °                    °
          |                |___________________|      °°°°°°°°°°°°°°°°°°°°°°
          |                         |
          |
          |                 _________________
          |                |   Was send to     |
          |                |   application     |_______
          |                |   successful?     | yes   |
          |                |___________________|       |
          |                         | no                |
          |                                           |
          |                 ___________________       |
          |                |For permanent error |      |
          |                | delete application |      |
          |                |  entry from list   |      |
          |                |____________________|      |
          |                         |                 |
          |                                           |
          |                 _________________         |
 _________________          |Prepare MDS Routing |     |
||   Building MDS   ||      | Unit w/sense data  |     |
||  Error Messages  ||______| X'08A8 000C' or    |     |
||                  ||      | X'08A8 000F', set  |     |
||__________________||      |  Status to ERROR   |     |
          |                |____________________|     |
                                                      |
 _________________                                    |
||   MDS Routing    ||                                |
||    Algorithm     ||_________________________________ÿ|
||                  ||
||__________________||                             (exit)
```

Figure 9-19. Processing of Local Application Routing List (Part 3 of 3)

This process is started by "MDS Routing Algorithm" to route an MDS Routing Unit (INBOUND) (see "MDS Internal Signal - MDS Routing Unit" in topic 2.9.2.4.4) to a local MS application program. This process is illustrated in Figure 9-19.

1. The Local Application Program Routing list is searched for an entry with a matching MS application program name to that in the MS Application program Name (X'03') subfield of the Destination Location Name (X'81') subvector in the MDS Routing Information (X'1311') GDS variable.

2. If a matching entry is *not* found:

   a. The Routing Status flag is set to ERROR.

   b. Sense data is set to X'08A8 0003' (MS application program name not recognized).

   c. "Building MDS Error Messages and Alerts" is invoked.

   d. The MDS error message (in an MDS Routing Unit) is passed to "MDS Routing Algorithm."

3. If a matching entry is found:

   a. The MDS-MU Received signal is prepared by copying the MDS-MU from the MDS Routing Unit (INBOUND).

b. Protocol boundary R to the MS application program is invoked by sending the MDS-MU Received signal to the MS application program

c. If the communication with the MS application program fails:

1) An MDS Routing Unit is prepared from the MDS-MU

2) The Routing Status flag is set to ERROR.

3) Sense data is set to X'08A8 000C' (MS application program failure) or X'08A8 000F' (MS application program failure due to congestion), depending upon the nature of the failure.

Implementation Note

The nature of this interprocess communication and the manner in which failures or congestion is detected are implementation-specific. Sense data X'08A8 000F' (MS application program failure due to congestion) is defined for use by those implementations that can detect that condition. Other implementations are not required to report that value.

4) For X'08A8 000C' (MS application program failure), the Local Application Program Routing List entry is deleted.

5) "Building MDS Error Messages and Alerts" is invoked.

6) The MDS error message (in an MDS Routing Unit) is passed to "MDS Routing Algorithm."

Processing of Active Transaction List

Process Name	Processing of Active Transaction List
Description	This process builds and maintains the active transaction list and creates MDS error messages to notify MS application programs of transaction failures.
Input	The input to this process is one of the following: ° "MDS Internal Signal - MDS Routing Unit" in topic 2.9.2.4.4 ° "MDS Internal Signal - SNASVCMG SON (Session-Outage Notification)" in topic 2.9.2.4.4 ° "Protocol Boundary O--Register Application Program Name" in topic 2.9.1.11 ° "CP_STATUS Signal" in topic 2.9.2.4.5
Output	° Active transaction list is updated. ° MDS Routing Unit is updated with routing information from the list if available. ° For correlation errors, sense data is passed back to the invoking process ("MDS Routing Algorithm") so that an MDS error message can be created. ° For SNASVCMG SON, an error indication is returned if the session outage impacts active transactions. ° For CP_STATUS, MDS error messages are created for all active transactions impacted by the session outage. ° For replies inbound from back-level nodes, a copy of the active transaction list entry matching the back-level node is returned. This makes request/reply correlation possible for "Back-level Node Support." ° For MDS Routing Unit (REPLY TO OUTBOUND) with status ERROR, MDS error messages are created for all active transactions impacted by the session-related error detected by MDS_SEND.

The MDS_ROUTER maintains a list of active transaction control blocks, one for each outstanding unit of work (i.e., MDS request (with reply)). The active transaction control blocks provide reverse routing information for the replies and also make it possible for the router to propagate error notifications to affected application programs when a session-related error is detected.

Each active transaction control block contains the information shown in Table 9-7.

Table 9-7. Contents of Active Transaction List

Field Name	Description of Contents
Correlator	Agent Unit-of-Work Correlator (X'1549') GDS variable from initial request.
Origin	Origin of initial request: ° NETID ° NAU ° MS application program
Destination	Destination of initial request: ° NETID ° NAU ° MS application program
sessions to Origin	sessions (if any) used to reach the origin. ° Session partner name ° Session mode name ° Contention-winner CGID (CPSVCMG mode name only) ° Contention-loser CGID (CPSVCMG mode name only)
sessions to Destination	sessions (if any) used to reach the destination. ° Session partner name ° Session mode name ° Contention-winner CGID (CPSVCMG mode name only) ° Contention-loser CGID (CPSVCMG mode name only)

The function of this process is illustrated in Figure 9-20.

Figure 9-20. Processing of Active Transaction List (Part 1 of 6)

The connector boxes in the illustration can be found on the following pages:

1. Page 2.9.2.4.8.
2. Page 2.9.2.4.8.
3. Page 2.9.2.4.8.
4. Page 2.9.2.4.8.
5. Page 2.9.2.4.8.

Active Transaction List Processing for MDS Routing Unit

```
                                  | 1 |
                                  |___|
                                    |
                              (MDS Routing Unit)
                                    |

                            _________________
                           |   search Active   |          ___
                           | Transaction List  |_________ÿ| 6 |
                           | for matching UOW  | no match |___|
                           |_________________|
                                    | match

                            _________________
                           |                 |
                 _________| MDS Error Message? |
          |             yes |                 |
          |                 |_________________|
          |                          | no
          |
          |                 _________________        _________________
          |                |                 |       | update MDS Routing |
          |                |   FIRST flag ON?  |_____ÿ| Unit w/sense data  |
          |                |                 | yes   |                   |
          |                |_________________|       |_________________|
          |                          | no                      |

 _________________        _________________        _________________
| update MDS Routing |     |                 |       |   If destination   |
|Unit w/session info |_____|   LAST flag ON?   |      |matches list entry, |
|                   |  yes |                 |       | delete list entry  |
|_________________|       |_________________|       |_________________|
          |                          | no                      |
                                                          |
 _________________        _________________                 |
|                 |       | update MDS Routing |             |
| delete list entry |     |Unit w/session info |             |
|                 |       |                 |             |
|_________________|       |_________________|             |
          |                          |                      |
          |_________________________ÿ|_____________________|

                                (exit)
```

Figure 9-21. Processing of Active Transaction List (Part 2 of 6)

This process is invoked by "MDS Routing Algorithm" and is passed an MDS Routing Unit (either INBOUND or OUTBOUND).

1. The active transaction list is searched for an entry with a Agent Unit-of-Work Correlator (X'1549') GDS variable matching the one in the MDS Routing Unit.

2. If a match is not found, the steps shown on page 2.9.2.4.8 (connector box 6) are performed.

   If a match is found, the following steps are performed.

3. If the MDS-MU is an MDS error message:

   a. The Routing Status in the MDS Routing Unit is set to COMPLETE.

   b. The Session information for the destination is copied from the list entry to the MDS Routing Unit

   c. The active transaction control block is deleted from the list.

   d. This process exits.

4. If the First MDS Message Indicator is 1, there should be no matching entry in the list:

   a. The Routing Status in the MDS Routing Unit is set to ERROR.

   b. The Sense Data field is set to X'08A8 000E' (Correlation Error).

   c. If the Origin Location Name in the MDS-MU matches either the Origin or Destination names stored in the Active Transaction control block, then the error is a state mismatch problem between the application programs. The other application program must receive an MDS error message, as well, to reset the state of the transaction:

      1) The MDS error message Required in the MDS Routing Unit is set to BOTH.

2) Alert Required is set to YES.

3) The active transaction control block is deleted from the list.

d. If the Origin Location Name in the MDS-MU does *not* match either the Origin or Destination names in the control block, then the error is collision of Agent Unit-of-Work Correlator (X'1549') GDS variable values. This should not occur unless an MS application program is re-using Agent Unit-of-Work Correlator (X'1549') GDS variable values assigned by other application programs:

1) The MDS error message Required in the MDS Routing Unit is set to ORIGIN.

2) Alert Required is set to YES.

3) The active transaction control block is *not* deleted from the list, since neither the origin or destination application programs will receive the MDS error message, only the third MS application program, which attempted to use an existing Agent Unit-of-Work Correlator (X'1549') GDS variable.

e. This process exits.

5. If the Last MDS Message Indicator is 1

a. The Routing Status in the MDS Routing Unit is set to COMPLETE.

b. The Session information for the destination is copied from the list entry to the MDS Routing Unit

c. The active transaction control block is deleted from the list.

d. This process exits.

6. Otherwise (First OFF, Last OFF)

a. The Routing Status in the MDS Routing Unit is set to COMPLETE.

b. The Session information for the destination is copied from the list entry to the MDS Routing Unit

c. This process exits.

```
                                   ___
                                  | 6 |
                                  |___|
                                    | (no match on UOW)
                                    |

                             _________________
                            |                 |
                            |    MDS Reply?   |_________________
                            |                 | yes             |
                            |_________________|                 |
                                      | no                        |

                             _________________        _________________
                            |                 |       | update MDS Routing |
                            |  FIRST flag OFF?|______ÿ| Unit w/sense data  |
                            |                 | yes   |                    |
                            |_________________|       |____________________|
                                      | no                        |
                                                                |
 _________________          _________________                    |
|                 |        |                 |                   |
| create list entry |______|  LAST flag OFF?  |                  |
|                 |   yes  |                 |                    |
|_________________|        |_________________|                    |
          |                         | no                          |
          |_________________________ÿ|_____________________________|

                                  (exit)
```

Figure 9-22. Processing of Active Transaction List (Part 3 of 6)

If a match was not found, the following steps are performed.

1. If the MDS-MU is an MDS reply:

a. The Routing Status in the MDS Routing Unit is set to ERROR.

b. The Sense Data field is set to X'08A8 000E' (Correlation Error).

c. The MDS error message Required in the MDS Routing Unit is set to ORIGIN.

d. Alert Required is set to YES.

e. This process exits.

2. If the First MDS Message Indicator is 0 (i.e., this MDS-MU is *not* FIRST, but there is no matching correlator in the list):

a. The Routing Status in the MDS Routing Unit is set to ERROR.

b. The Sense Data field is set to X'08A8 000E' (Correlation Error).

c. The MDS error message Required in the MDS Routing Unit is set to ORIGIN.

d. Alert Required is set to YES.

e. This process exits.

3. If the Last MDS Message Indicator is 0 (i.e., this MDS-MU is FIRST and *not* LAST):

a. An active transaction control block is created and added to the list. The session-related information will be added later when the MDS Routing Unit (REPLY TO OUTBOUND) is processed.

b. This process exits.

4. Otherwise (First ON, Last ON -- no active transaction control block is needed):

a. This process exits.

Active Transaction List Processing for SNASVCMG SON

Figure 9-23. Processing of Active Transaction List (Part 4 of 6)

This process is invoked by "Processing SNASVCMG SON" and is passed an SNASVCMG SON signal. Figure 9-23 illustrates the processing.

1. The Active transaction list is searched for a match on the session partner name and session mode name.

2. If a match is found (just one match is all that is needed), an error indication is passed to the invoking process ("Processing SNASVCMG SON").

3. If no match is found in the Active Transaction list, no further action is required.

4. This process ends.

Active Transaction List Processing for Inbound Back-level Message

```
|                                 ___
|                                | 3 |
|                                |___|
|                                  |
|                       (MDS Routing Unit(INBOUND)
|                 w/back-level reply correlation required)
|                                  |
|
|                         _________________
|                        |  Search List for  |
|                        | match:  dest. NAU |
|             ___________| to origin session |
|            |   no match |  partner in MDS   |
|            |            |   Routing Unit    |
|            |            |___________________|
|            |                     | match
|            |
|            |            _________________
|            |           |   Make copy of    |
|            |           | Active Transaction |
|            |           |    List entry     |
|            |           |___________________|
|            |                     |
|
|     (exit error)              (exit)
```

Figure 9-24. Processing of Active Transaction List (Part 5 of 6)

This process is illustrated in Figure 9-24, and is described in detail below:

1. The Active Transaction List is searched for a match of the Network-qualified CP or LU name of the session partner in the MDS Routing Unit (INBOUND) to the Destination NETID and NAU fields of the list entries.

2. If a match is found, a copy of the Active Transaction List entry is passed back to the invoking process ("Back-level Node Support").

3. If a match is not found, an error indication is passed back to the invoking process.

Active Transaction List Processing for CP_STATUS and MDS_SEND Errors

```
          ___                          ___
         | 4 |                        | 5 |
         |___|                        |___|
           |                            |
 (MDS Routing Unit                  (CP_STATUS)
  (REPLY TO OUTBOUND))                  |

 _________________        _________________
|   Is sense data   |_____ÿ|   Search Active   |
|   X'08A8 000A'?   | yes  |Transaction list for|_____________________________
|                   |      |  match on session  | at end of list              |
|___________________|   ___ÿ|    information    |                             |
          | no          |  |___________________|                             |
          |             |             | for each match                        |
          |             |                                                     |
          |             |   _________________                                 |
          |             |  |   Does UOW match  |_________________             |
          |             |  | input MDS Routing | no or input was |            |
          |             |  |       Unit?       | CP_STATUS       |            |
          |             |  |___________________|                 |            |
          |             |             |yes                        |            |
                        |                                                     |
 _________________      |   _________________        _________________        |
||  Building MDS   ||    |  ||  Building MDS   ||      | Build MDS Routing |  |
||  Error Messages ||    |  ||  Error Messages ||______|Unit w/status ERROR,|  |
||    and Alerts   ||    |  ||    and Alerts   ||      | sense X'08A9 0001' |  |
||_________________||    |  ||_________________||      |    or X'08A9 0002' |  |
          |             |             |                 |____________________|  |
                        |                                                     |
 _________________      |   _________________                                 |
||  MDS Routing    ||    |  | Update MDS Routing |        _________________    |
||   Algorithm     ||    |  |   Unit w/status    |       |   Has input MDS   |__|
||                 ||    |  | COMPLETE, outbound |       | Routing Unit been |
||_________________||    |  |session information |       |     processed?    |___
          |             |  |___________________|        |___________________|yes|
          |             |             |                           | no        |
          |             |                                                     |
          |             |   _________________        _________________        |
          |             |  |   Delete Active   |      ||   Building MDS   ||  |
          |             |  |  Transaction List |      ||  Error Messages  ||  |
```

```
|            |    |       entry        |       ||    and Alerts    ||   |
|            |    |____________________|       ||__________________||   |
|            |             |                         |              |
|            |                                                      |
|            |    _________________          _________________      |
|            |    ||  MDS Routing   ||       ||  MDS Routing   ||   |
|            |____||   Algorithm    ||       ||   Algorithm    ||   |
|                 ||                ||       ||                ||   |
|                 ||________________||       ||________________||   |
|                                                     |             |
|_____________________________________________________ÿ|___________|

                                                     (exit)
```

Figure 9-25. Processing of Active Transaction List (Part 6 of 6)

This processing is invoked to search the Active Transaction List for transactions impacted by the error notification, and to generate MDS error messages for each of those transactions.

CP_STATUS is passed to this process only when connectivity to the adjacent CP has been lost, so any transaction with matching CPSVCMG session information will get an MDS error message

MDS Routing Unit (REPLY TO OUTBOUND) with routing status ERROR indicates that the MDS_SEND TP was unable to send the MDS-MU to the session partner. If the sense data in the MDS Routing Unit (REPLY TO OUTBOUND) indicates session allocation failure (X'08A8 000A'), the Active transaction list will be searched for all matches on the session information, and these transactions will all receive MDS error messages. Other errors detected by the MDS_SEND TP, such as remote TP failure (X'08A8 000B') are isolated to a particular transaction and do not cause the Active Transaction List to be searched for all matches.

The steps in the processing are listed below and illustrated in Figure 9-25.

1. If the input is an MDS Routing Unit (REPLY TO OUTBOUND), is the sense data X'08A8 000A' (Session allocation failure)?

   The test of sense code is required because other failure causes should not trigger the sending of MDS error messages for other transactions.

   If this is *not* true:

   a. The MDS Routing Unit is updated:

      - The MDS error message Required indicator in the MDS Routing Unit is set to ORIGIN (i.e., sent an MDS error message to the origin of the MDS-MU).
      - The Alert Required indicator in the MDS-MU is set to NO.

   b. "Building MDS Error Messages and Alerts" is invoked.

   c. "MDS Routing Algorithm" is invoked to route the MDS error message to its destination (in this case, the original sender).

   d. This process exits.

2. If the input is a CP_STATUS signal or an MDS Routing Unit (REPLY TO OUTBOUND) with sense code X'08A8 000A', then connectivity has been lost and application programs with outstanding transactions must be notified. The MDS error messages sent to these MS application programs will have sense data values of either:

   - X'08A9 0001' -- failure of CPSVCMG session.

   - X'08A9 0002' -- failure of SNASVCMG session.

   Note: This processing does *not* apply to an MDS Routing Unit (REPLY TO OUTBOUND) with sense data X'08A8 000B' (unrecoverable TP failure in remote node). That error only applies to a single MDS-MU - it is not the equivalent of session-outage notification. An unrecoverable TP failure in a remote node may be a transient error and subsequent communication may succeed. The MS application program receiving an MDS error message with that sense data will persue application program-level recovery, such as switching to a backup focal point, but other application program-level traffic should not be impacted by this error.

   a. The Active Transaction List is searched for a match on the session information in the input signal.

For CP_STATUS, this session information is:

- CP name from the signal
- Mode name CPSVCMG

For MDS Routing Unit (REPLY TO OUTBOUND), this session information is:

- Outbound session Network-qualified CP or LU name
- Outbound Session mode name

The search is for every transaction which uses the impacted session to reach either the origin or destination (relative to the initial request of the transaction).

b. For each transaction found:

1) Does the Agent Unit-of-Work Correlator (X'1549') GDS variable match the MDS Routing Unit (REPLY TO OUTBOUND) which was input to this process (this check is not done if the input was CP_STATUS)? If YES, then the MDS Routing Unit is updated:

- MDS error message required ORIGIN
- Alert required NO

2) Otherwise, an MDS Routing Unit is built (this will be the input to Building MDS Error Messages and Alerts)

- Direction OUTBOUND
- Status ERROR
- MDS error message required:
  - MDS error message required ORIGIN, if the *destination* in the list entry is the one impacted by the error.
  - MDS error message required DESTINATION if the *origin* in the list entry is the one impacted by the error.
- Alert required NO
- Origin location name (origin from the list entry)
- Destination location name (destination from the list entry)

3) "Building MDS Error Messages and Alerts" is invoked.

4) "MDS Routing Algorithm" is invoked to route the MDS error message to its destination (in this case, the MS application program to which, presumably, connectivity still exists).

5) The search through the list continues.

c. When all the entries in the Active Transaction List have been processed, this process must be sure that the input MDS Routing Unit (REPLY TO OUTBOUND) (if that was the input) has been processed. If not:

1) The MDS Routing Unit is updated:

- MDS error message required ORIGIN
- Alert required NO

2) "Building MDS Error Messages and Alerts" is invoked.

3) "MDS Routing Algorithm" is invoked to route the MDS error message to its destination (in this case, the MS application program to which, presumably, connectivity still exists).

d. This process exits.

Processing of CPSVCMG Session List

Process Name	Processing of CPSVCMG Session List
Description	This process builds and maintains the CPSVCMG Session List and sends "Protocol Boundary S--Forward CP Status" in topic 2.9.1.14 to MS_CAPS. It also takes part in the MDS Routing Algorithm processing by supplying CPSVCMG session information for routing between network node servers and end nodes.

Input	The input to this process is one of the following: ° "Protocol Boundary U--Begin Sending CP Status Information" in topic 2.9.1.16 ° "CP_STATUS Signal" in topic 2.9.2.4.5 ° "MDS Internal Signal - MDS Routing Unit" in topic 2.9.2.4.4
Output	° CPSVCMG Session list is updated. ° Forward CP Status is prepared and passed to MS_CAPS. ° MDS Routing Unit is updated with CPSVCMG session information, if applicable.

This list contains the status of all CPSVCMG sessions used by MDS. In a network node, the list has an entry for each client APPN end node in the network node domain. In an APPN end node, the list has a single entry for the CPSVCMG sessions to the network node server. This information is maintained for routing MDS-MUs between APPN end nodes and network node servers.

The information kept is described in Table 9-8.

Table 9-8. Contents of CPSVCMG Session List

Field Name	**Description of Contents**
Control Point Name	Network-qualified CP name of session partner
MDS Level	Level of MULTIPLE_DOMAIN_SUPPORT support in adjacent node: ° Unknown ° Back-level ° Current
Contention-winne CGID	Conversation Group ID for the contention-winner session. A null value indicates that the session is not active.
Contention-loser CGID	Conversation Group ID for the contention-loser session. A null value indicates that the session is not active.
CP Capabilities	A copy of the CP Capabilities GDS variable received from the adjacent CP.

The function of this process is illustrated in Figure 9-26.

Figure 9-26. Processing of CPSVCMG Session List (Part 1 of 5)

The three connector boxes referenced in the illustration are:

1. Page 2.9.2.4.8
2. Page 2.9.2.4.8
3. Page 2.9.2.4.8

CPSVCMG Session List Processing of CP_STATUS Signal

```
                |___|
                  |
          (CP_STATUS signal)
                  |

        _________________
       |  Search list for  |_________________
       |  matching CP name | no match        |
       |___________________|                 |
                 | match                     |

        _________________        _________________
       |    Update entry   |      |    Create entry   |
       |                   |      |                   |
       |___________________|      |___________________|
                 |                         |
                 |_________________________|

        _________________        _________________
       |  CP_CAPABILITIES  |_____ÿ| Connectivity lost? |____
       |     received?     | no   |____________________| no |
       |___________________|                 | yes          |
                 | yes                       |              |
                                                          |
                  ___                     _________________   |
                 | 4 |_______________|   Delete entry    |   |
                 |___|               |  set RC to error  |   |
                                     |___________________|   |
                                                             |

                                                          (exit)
```

Figure 9-27. Processing of CPSVCMG Session List (Part 2 of 5)

Whenever a CPSVCMG session becomes active or inactive, CP session services prepares the CP_STATUS signal described in "CP_STATUS Signal" in topic 2.9.2.4.5 and passes it to the MDS_ROUTER. In a network node, session services passes CP_STATUS signals to MS only for CPSVCMG sessions to client end nodes. In an end node, the only CP_STATUS signals are for the CPSVCMG sessions to the network node server.

When the MDS_ROUTER receives the signal, it performs the following steps. This processing is illustrated in Figure 9-27.

1. The CPSVCMG session status list is searched for an entry with a CP name matching the one in the CP_STATUS signal.

2. If a match is found, the entry is updated with the new information:

   a. If the signal indicates that a session has become active, the corresponding CGID in the entry is updated.

   b. If the signal contains a CP Capabilities GDS variable received from the CP, the CP Capabilities GDS variable from the signal is copied into the entry.

   c. If the signal indicates that a session has become inactive, the corresponding CGID in the entry is set to null. If this causes both CGIDs to be null:

      1) A Forward CP Status signal is prepared for MS_CAPS. See "Preparing Forward CP Status" for a description of this process.

      2) The entry in the list is deleted.

      3) An error indication is passed back to the invoking process ("Processing CP_STATUS Signal").

3. If a match is not found, a new entry is inserted into the list:

   a. The CP name is copied to the entry.

   b. If the signal indicates that a session has become active, the corresponding CGID in the entry is added.

   c. The CP Capabilities GDS variable from the signal is copied into the entry.

      Byte 8, bit 6 of the CP Capabilities is examined so that the MDS support level field in the entry can be set. If the bit is 1, the CP supports the current level of MULTIPLE_DOMAIN_SUPPORT, so the field is set to CURRENT. If the bit is 0, the CP does not support the current level of MULTIPLE_DOMAIN_SUPPORT, so the field

is set to BACK-LEVEL.

4. If the signal contains a CP Capabilities GDS variable received from the CP, a Forward CP Status signal is prepared for MS_CAPS. See "Preparing Forward CP Status" for a description of this process.

Preparing Forward CP Status

```
        ___                         ___
       | 2 |                       | 4 |
       |___|                       |___|
         |Begin Sending CP           |
         |Status Info (PB U)         | CP_STATUS

 ___________________        ___________________
| Remember that PB U |      |   Has PB U been   |_____________
|  has been invoked  |      |previous invoked?  | no          |
|____________________|      |___________________|             |
          |                          | yes                     |
                                                            |
 ___________________        ___________________               |
|  Prepare 'Forward  |      |  Prepare 'Forward  |             |
| CP Status' for all |      |   CP Status' for   |             |
|  entries in list   |      | indicated CP name  |             |
|____________________|      |____________________|             |
          |                          |                         |
          |__________________________ÿ|                        |
°°°°°°°°°°°°°°°°°°°°°°                                         |
°                    °      ___________________               |
°      MS_CAPS       °_____| Send 'Forward CP  |              |
°                    °      | Status' to MS_CAPS |             |
°°°°°°°°°°°°°°°°°°°°°°      |___________________|              |
                                   |                          |
                                   |_________________________|

                                  (exit)
```

Figure 9-28. Processing of CPSVCMG Session List (Part 3 of 5)

This process is started when information concerning a CPSVCMG session must be passed to MS_CAPS. This occurs either when "CPSVCMG Session List Processing of CP_STATUS Signal" is invoked or when MS_CAPS invokes Protocol Boundary U--Begin Sending CP Status Information.

This process is illustrated in Figure 9-28.

1. If this process was invoked by "Processing CP_STATUS Signal," and MS_CAPS has previously invoked protocol boundary U, Forward CP Status (described in "Protocol Boundary S--Forward CP Status" in topic 2.9.1.14) is prepared as follows:

   a. NETID and CP name are copied from the CP_STATUS signal.

   b. The new status (ACTIVE or INACTIVE) is copied from the CP_STATUS signal. This process is not invoked for a PENDING-ACTIVE CP status message.

   c. If the new status is ACTIVE, the CP Capabilities GDS variable is copied from the CP_STATUS signal.

   d. Protocol boundary S to MS_CAPS is invoked.

2. If this process was invoked by protocol boundary U ("Begin Sending CP Status Information"), Forward CP Status (described in "Protocol Boundary S--Forward CP Status" in topic 2.9.1.14) is prepared as follows:

   a. The fact that protocol boundary U has been invoked is remembered.

   b. The CPSVCMG session list is processed sequentially. For each entry that contains a CP Capabilities GDS variable, an entry in Forward CP Status is created:

      1) NETID and CP name are copied.

      2) Status (ACTIVE or INACTIVE) is copied.

      3) The CP Capabilities GDS variable is copied.

c. When all the entries in the CPSVCMG session list have been processed, protocol boundary S to MS_CAPS is invoked.

CPSVCMG Session List Processing for MDS Routing Unit:

This process is invoked as part of the MDS routing algorithm. It supplies CPSVCMG session information when appropriate for an MDS Routing Unit (OUTBOUND). In an end node, the only entry in the CPSVCMG session list is that of the network node server, and this value will always be supplied.
| In a network node, this process searches for a match on the name of one of
| the client end nodes.

```
                                  ___
                                 | 3 |
                                 |___|
                                   |
                                   |

                        _________________
                       | Is there an active |
                       |     session to     |______
                       |     NN server?     | no   |
                       |____________________|      |
                                 | yes             |
                                                  |
                        _________________         |
                 _______|  Is NN back-level? |     |
                |   no  |____________________|     |
                |                 | yes            |
                |                                  |
                |       _________________          |
                |      |Does MDS-MU contain |_____ÿ|
                |      |     an Alert?      | no   |
                |      |____________________|      |
                |                 | yes            |
                |_________________ÿ|               |
                                                  |
                        _________________         |
                       | update MDS Routing |     |
                       |Unit w/session info.|     |
                       |   for NN server?   |     |
                       |____________________|     |
                                 |                |
                                 |________________|

                                (exit)
| End Node Only
```

Figure 9-29. Processing of CPSVCMG Session List (Part 4 of 5)

If this node is an end node, the following is done (illustrated in Figure 9-29):

1. Is there an active CPSVCMG session to a network node server?

   a. If there is, is the network node server back-level?

   b. If the network node server is back-level, does the MDS-MU contain a CP-MSU GDS variable with an Alert major vector? That is the only data which a back-level network node server will accept from an end node. If the MDS-MU does not contain an Alert, the Routing Status in the MDS Routing Unit remains INCOMPLETE.

   c. If the network node server is not back-level, or if it is, but the MDS-MU contains an Alert, the MDS Routing Unit (OUTBOUND) is completed with information from the CPSVCMG Session List entry for the network node server:

      1) Routing status is set to COMPLETE.

      2) Session partner Network-qualified CP name is copied.

      3) Session mode name is set to CPSVCMG

      4) Session CGIDs are copied.

2. If there is not an active CPSVCMG session to a network node server, the Routing Status in the MDS Routing Unit remains INCOMPLETE.

3. This process exits.

```
|                                    ___
|                                   | 3 |
|                                   |___|
| Network Nodes Only                  |
|
|                            ___________________
|                           |  Search list for  |_________________
|                           |  matching CP name | no match        |
|                           |___________________|                 |
|                                    | match                       |
|
|                            ___________________        ___________________
|                           |    Is MDS Level   |      |     Was MDS-MU    |
|            ________________|    BACK-LEVEL?    |      |   received over   |____
|           |             yes |                   |      | SNASVCMG session? | no |
|           |                 |___________________|      |___________________|    |
|           |                          | no                     | yes            |
|                                     |                                        |
| ___________________                 |                 ___________________    |
| | Update MDS Routing |              |                 | Update MDS Routing |   |
| | Unit w/sense data  |              |                 | Unit w/sense data  |___ÿ|
| |    X'08A8 0005'    |              |                 |    X'08A8 0009'    |    |
| |____________________|              |                 |____________________|    |
|           |                          |
|           |                                                                    |
|           |                            ___________________    ___________________
|           |                           |     Was MDS-MU    |  |  Search list for  |
|           |                           |   received over   |_____ÿ| match on session |
|           |                           |  CPSVCMG session? | yes |    partner name   |
|           |                           |___________________|  |___________________|
|           |                                    | no                 |match   |no
|           |                                    |                    |        |match
|           |                                    |____________________|        |
|           |                                    |
|           |                            ___________________    ___________________
|           |                           | update MDS Routing |  | Update MDS Routing |
|           |                           |Unit w/session info.|  | Unit w/sense data  |
|           |                           | for destination EN |  |    X'08A8 0004'    |
|           |                           |___________________|   |____________________|
|           |                                    |                        |
|           |                                    |
|           |___________________________ÿ|_____________________________________|
|                                        |
|
|                                      (exit)
```

| Figure 9-30. Processing of CPSVCMG Session List (Part 5 of 5)

| If this node is a network node, the following is done (illustrated in
| Figure 9-30):

| 1. The CPSVCMG Session List is searched for a match on the destination
| location name in MDS Routing Information (X'1311') GDS variable of the
| MDS-MU.

| 2. If a match is *not* found:

| a. If the session mode name in the MDS Routing Unit is SNASVCMG
| (i.e., MDS-MU was received over an SNASVCMG session from another
| network node):

| 1) The MDS Routing Unit is updated with sense data X'08A8 0009'
| (destination not supported by reporting network node).

| 2) The Routing Status is set to ERROR.

| b. This process exits.

| 3. If a match is found in the CPSVCMG Session List:

| a. If the MDS Level of the session partner is BACK-LEVEL, then the
| end node does not support receipt of any MS requests or replies.

| 1) The MDS Routing Unit is updated with sense data X'08A8 0005'
| (function not supported by end node destination).

| 2) The Routing Status is set to ERROR.

| 3) This process exits.

| b. If the session mode name in the MDS Routing Unit is CPSVCMG (i.e.,
| MDS-MU was received over a CPSVCMG session):

| 1) The CPSVCMG Session List is searched for a match on the
| session partner LU name in the MDS Routing Unit.

```
|         2)  If a match is not found, the session partner is not in a
|             client end node, and must be in an adjacent network node.
|             MULTIPLE_DOMAIN_SUPPORT does not allow the use of CPSVCMG
|             sessions between adjacent network nodes (only between an end
|             node and its network node server).

|             a)  The MDS Routing Unit is updated with sense data X'08A8
|                 0004' (use of CPSVCMG session not permitted).

|             b)  The Routing Status is set to ERROR.

|             c)  This process exits.

|     c.  The MDS Routing Unit (OUTBOUND) is completed with information from
|         the CPSVCMG Session List entry for the destination end node.

|         1)  Routing Status is set to COMPLETE.

|         2)  Session partner Network-qualified LU name is copied.

|         3)  Session mode name is set to CPSVCMG

|         4)  Session CGIDs are copied.

|     d.  This process exits.
```

Processing of SNASVCMG Session List

Process Name	Processing of SNASVCMG Session List
Description	This process builds and maintains the SNASVCMG Session List. It also takes part in the MDS Routing Algorithm processing for network nodes by supplying SNASVCMG session information for routing.
Input	The input to this process is one of the following: ° "MDS Internal Signal - MDS Routing Unit" in topic 2.9.2.4.4 ° "MDS Internal Signal - SNASVCMG SON (Session-Outage Notification)" in topic 2.9.2.4.4
Output	The output from this process is one of the following: ° SNASVCMG Session list is updated. ° MDS Routing Unit is updated with SNASVCMG session information, if applicable. ° If both contention-winner and contention-loser sessions become inactive, an error indication is passed back to invoking process.

This list contains information about each remote node with which an SNASVCMG session is currently active. This information is maintained to optimize the MDS routing algorithm. Each entry in this list contains the information shown in Table 9-9.

Table 9-9. Contents of SNASVCMG Session List

Field Name	**Description of Contents**
NAU Name	Network-qualified LU name of session partner
MDS Level	Level of MULTIPLE_DOMAIN_SUPPORT support in session partner: ° Unknown ° Back-level ° Current
Session CGID	Conversation Group ID (session type, contention-winner or contention-loser, is not known for the SNASVCMG sessions). A null value indicates that the session is not active.
Session CGID	Conversation Group ID (session type, contention-winner or contention-loser, is not known

```
|                 | for the SNASVCMG sessions).  A null value indicates  |
|                 | that the session is not active.                      |
|_________________|______________________________________________________|
```

An entry is added to this table whenever an incoming message is received over an SNASVCMG session not already in the list. An entry is also added when an outgoing message is routed over a session not already in the list.

The MDS level information reflects the status of the trial-and-error procedure followed by the MDS service transaction programs for back-level support. Thus, if the service transaction programs have already detected that the session partner is a back-level node, subsequent communications to the node can be optimized (i.e., avoid the trial-and-error allocation procedure used to detect a back-level node).

This process is illustrated in Figure 9-31.

```
                              (input signal)
                                    |
                                    |

                         ___________________
                        |     select on     |
          _____________ | input signal type |_______________
         | MDS Routing Unit |                | SNASVCMG SON  |
         |                 |___________________|               |
         |                                                     |
         |                                                     |
         |                                                     |

        ___                                                   ___
       | 1 |                                                 | 2 |
       |___|                                                 |___|
```

Figure 9-31. Processing of SNASVCMG Session List (Part 1 of 4)

The connector boxes in the illustration are found on the following pages:

1. Page 2.9.2.4.8
2. Page 2.9.2.4.8

SNASVCMG Session List Processing of MDS Routing Unit

```
                                  ___
                                 | 1 |
                                 |___|
                                   |

                         ___________________
                        |     Select on     |
                        |     Direction     |_________________ÿ| 3 |
                        |                   | Inbound or       |___|
                        |___________________| Outbound Reply
                                 |outbound

                         ___________________
                        |     Select on     |
          _____________ |  Routing Status   |
         |     incomplete  |                 |
         |                 |___________________|
         |                          | complete

 ___________________      ___________________
|  Search list for  |     |  Search list for  |
|match on destination|    |  match on session |_________
|     NAU name      |     |   partner name    | no match |
|___________________|     |___________________|          |
       | no     | match              | match             |
       | match  |                    |                   |
       |        |____________________ÿ|                   |
       |                                              |
       |                    ___________________           |
       |                   | Update MDS Routing |         |
       |                   |Unit w/session info |         |
       |                   |    from entry      |         |
       |                   |___________________|          |
       |                            |                     |
       |____________________________ÿ|____________________|

                              (exit)
```

Figure 9-32. Processing of SNASVCMG Session List (Part 2 of 4)

The processing of MDS Routing Unit is illustrated in Figure 9-32.

The steps in the processing are described below.

1. If Direction is OUTBOUND:

   a. If Routing Status is INCOMPLETE, the SNASVCMG session list is searched for an entry with session partner name matching the destination location name in the MDS Routing Information (X'1311') GDS variable.

      In this case, Processing of SNASVCMG Session List was invoked to see if a session already exists to the destination NAU. If so, that session is used to send the MDS-MU.

   b. If Routing Status is COMPLETE, the SNASVCMG session list is searched for an entry with session partner name matching the session partner name in the MDS Routing Unit (OUTBOUND).

      In this case, the session partner has already been selected (either because MS application program requested DIRECT, or because "Directory Services Processing" in topic 2.9.2.6.2 has been invoked, and Directory Services Processing invoked this process). Processing of SNASVCMG Session List was invoked to see if a session already exists to the chosen session partner.

   c. In either case, if a match is found, the fields in MDS Routing Unit (OUTBOUND) are updated as follows from the list entry:

      - Session CGIDs are copied
      - Session mode name is set to SNASVCMG.
      - MULTIPLE_DOMAIN_SUPPORT support level is copied.
      - Routing Status is set to COMPLETE.

   d. This process exits.

Figure 9-33. Processing of SNASVCMG Session List (Part 3 of 4)

2. If Direction is INBOUND or REPLY TO OUTBOUND:

   a. The SNASVCMG session list is searched for an entry with session partner name matching the session partner LU name in the MDS Routing Unit.

   b. If a match is found:

      1) If the Routing status in the MDS Routing Unit is *not* ERROR, the list entry is updated as follows from the signal:

         - If one of the session CGID fields is null, the session CGID from the signal is copied to it.

         - If the MDS support level in the entry is UNKNOWN, the value from the signal is copied.

      2) If the Routing Status is ERROR, the MDS TP was unable to ALLOCATE a conversation with the session partner on either the

contention-winner or contention-loser sessions. Since neither session is usable, the entry is deleted.

c. If a match is not found:

If the Routing status in the MDS Routing Unit is *not* ERROR, an entry is created as follows from the signal:

- Session partner name is copied.
- Session mode name is set to SNASVCMG.
- Session CGID is copied.
- MDS support level is copied.

d. This process exits.

SNASVCMG Session List Processing of SNASVCMG SON

Figure 9-34. Processing of SNASVCMG Session List (Part 4 of 4)

This process is illustrated in Figure 9-34. The processing is described below:

1. The SNASVCMG Session List is searched for an entry with session partner name matching the session partner name in MDS Internal Signal - SNASVCMG SON (Session-Outage Notification).

2. If an entry is found, it is updated as follows from the signal:

   a. The session CGID field matching the CGID in the signal is set to null.

   b. If both CGID entries are null, the entry is deleted from the list and an error indication is passed back to the invoking process ("Processing SNASVCMG SON").

MDS Header Parsing

Process Name	MDS Header Parsing
Description	This process performs the required checking of MDS

	header syntax.
Input	"MDS Internal Signal - MDS Routing Unit" in topic 2.9.2.4.4
Output	If a format exception is detected, the MDS Routing Unit will be updated: ° Routing Status is set to ERROR. ° Sense Data is set to appropriate value for the error.

This process is illustrated in Figure 9-35.

```
                              (MDS Routing Unit)
                                      |
                                      |

                            ___________________
                           |                   |
                           |    Parse MDS-MU   |
                           |                   |
                           |___________________|
                                      |

                            ___________________   ____________________
                           | Format exception? |_____ÿ| Update MDS Routing |
                           |___________________| yes  |  unit w/sense data |
                                      | no            |Routing Status ERROR|
                                      |               |____________________|
                                      |                         |
                                      |_________________________|

                                   (exit)
```

Figure 9-35. MDS Header Parsing

The following processing is done:

1. The MDS-MU (if present) in the MDS Routing Unit is parsed.

2. If one of the format exceptions described in Table 9-10, is detected, the MDS Routing Unit is updated as follows:

   ° Routing Status is set to ERROR.

   ° Sense data is set to the appropriate value from Table 9-10.

   ° Alert required is set to YES. (Any format exception for the MDS header information is an Alert condition.)

   ° MDS error message required is set to NO (If the MDS header information cannot be parsed, then it is not possible to create a syntactically-valid MDS error message from that header information.)

3. If the MDS header information was parsed successfully, then the values are copied into the appropriate fields in the MDS Routing Unit.

Table 9-10. Required Format Checks for MDS-MUs

Order	Sense Data	Condition
1	X'08AA 1310'	GDS variable received is not an MDS-MU.
2	X'086F 0007'	The length field of an MDS-MU is incompatible with the sum of the lengths of the imbedded GDS variables or an invalid length was found in an imbedded structure (or GDS variable).
3	X'08AA 1311'	MDS Routing Information (X'1311') GDS variable not present.
4	X'08AA 1549'	Agent Unit-of-Work Correlator (X'1549') GDS variable not present.
5	X'086F 0002'	The sum of the subvector lengths in the MDS Routing Information (X'1311') GDS variable is incompatible with the length of the GDS variable.
6	X'086C 8100'	Origin Location Name (X'81') subvector not present.
7	X'086D 8101'	NETID (X'01') subfield of the Origin Location

		Name (X'81') subvector not present.
8	X'0870 8106'	Invalid subfield length in Origin Location Name (X'81') subvector. NETID is not a valid length (1 - 8 characters).
9	X'086D 8102'	NAU Name (X'02') subfield of the Origin Location Name (X'81') subvector not present.
10	X'0870 8106'	Invalid subfield length in Origin Location Name (X'81') subvector. NAU is not a valid length (1 - 8 characters).
11	X'086D 8103'	MS Application Program Name (X'03') subfield of the Origin Location Name (X'81') subvector not present.
12	X'0870 8106'	Invalid subfield length in Origin Location Name (X'81') subvector. MS application program name is not a valid length (1 - 8 characters).
13	X'086F 8103'	The sum of the subfield lengths in the Origin Location Name (X'81') subvector is incompatible with the length of the subvector.
14	X'086C 8200'	Destination Location Name (X'82') subvector not present.
15	X'086D 8201'	NETID (X'01') subfield of the Destination Location Name (X'82') subvector not present.
16	X'0870 8206'	Invalid subfield length in Destination Location Name (X'81') subvector. NETID is not a valid length (1 - 8 characters).
17	X'086D 8202'	NAU Name (X'02') subfield of the Destination Location Name (X'82') subvector not present.
18	X'0870 8206'	Invalid subfield length in Destination Location Name (X'81') subvector. NAU name is not a valid length (1 - 8 characters).
19	X'086D 8203'	MS Application Program Name (X'03') subfield of the Destination Location Name (X'82') subvector not present.
20	X'0870 8206'	Invalid subfield length in Destination Location Name (X'81') subvector. MS application program name is not a valid length (1 - 8 characters).
21	X'086F 8203'	The sum of the subfield lengths in the Destination Location Name (X'82') subvector is incompatible with the length of the subvector.
22	X'086C 9000'	Flags (X'90') subvector not present.
23	X'0870 9003'	Flags (X'90') subvector contains an invalid flag setting.
24	X'0870 9002'	Flags (X'90') subvector contains invalid setting for MDS message type.

Notes:

1. The **Order** column indicates the required order of checking for each of the format exceptions. This order is required so that all implementations of MULTIPLE_DOMAIN_SUPPORT will report format exceptions consistently.

2. The sense data is sent in the Sense Data (X'7D') subvector of the SNA Condition Report (X'1532') GDS variable. An Alert is also required when any of these errors is detected.

Building MDS Error Messages and Alerts

Process Name	Building MDS Error Message and Alerts
Description	This process builds MDS error messages and, when appropriate, invokes EP_ALERT to build and send an Alert.
Input	"MDS Internal Signal - MDS Routing Unit" in topic 2.9.2.4.4
Output	° An MDS error message is created, replacing the MDS-MU in the input MDS Routing Unit. ° EP_ALERT is invoked to build and send an Alert, if required.

```
|_____________|_____________________________________________________|

                                (MDS Routing Unit)
                                        |

                               _____________________
                              | Build MDS Routing  |
                              |    Information     |
                              |   (X'1311') GDS    |
                              |____________________|
                                        |

                               _____________________
                              |Build SNA Condition |
                              |Report (X'1532) GDS |
                              | w/sense data from  |
                              |  MDS Routing Unit  |
                              |____________________|
                                        |

|                              ____________________        ____________________
|                              |   Is sense data    |       |  Copy MDS-MU into  |
|                              |  X'08A8 0009' and  |______ÿ| MDS Error Message  |
|                              |Routing Verification| yes   |after SNA Condition |
|                              |   Indicator OFF?   |       |      Report.       |
|                              |____________________|       |____________________|
|                                        | no                        |
|                                        |___________________________|
|
                                                     °°°°°°°°°°°°°°°°°°°°°°
                               ____________________        °                    °
                              | Alert Required?    |______ÿ°                    °
                              |____________________| yes   °      EP_ALERT      °
                                        |                  °                    °
                                        |                  °°°°°°°°°°°°°°°°°°°°°°

                                       (exit)
```

Figure 9-36. Building MDS Error Messages and Alerts

This process is illustrated in Figure 9-36. Following is the order of processing:

1. The MDS Routing Information (X'1311') GDS variable is constructed with the values shown in Table 9-11.

   The Destination Location name is selected as follows:

   - If MDS error message required is set to ORIGIN, The Destination Location Name is taken from the *Origin* Location Name in the MDS Routing Unit.

   - If MDS error message required is set to DESTINATION, The Destination Location Name is taken from the *Destination* Location Name in the MDS Routing Unit.

   - If the sense data is X'08A8 0009' (destination not supported by reporting network node) *and* if the Routing Verification Indicator of the Flags (X'90') subvector of the MDS Routing Information (X'1311') GDS variable is 0, indicating that routing was based on nonverified directory information, the Destination Location Name NETID and NAU Name values are taken from the Inbound Session Information in the MDS Routing Unit. The MS application program name value is set to X'23F0F1F0' (MDS_ROUTER).

     This is done because the MDS error message in this case must be processed by the MDS_ROUTER in the network node that *routed* the MDS-MU. The undeliverable MDS-MU is included in the MDS error message in this case, also.

2. If the Agent Unit-of-Work Correlator (X'1549') GDS variable field of the MDS Routing Unit is not null, it is used in the MDS error message

3. If the Agent Unit-of-Work Correlator (X'1549') GDS variable field is null, a new correlator must be created. This Agent Unit-of-Work Correlator (X'1549') GDS variable should contain:

   - Reported on NETID - the NETID of this node
   - Reported on node ID - the NAU name of this node
   - Requester agent - X'23F0F1F0' (MDS_ROUTER)

4. The SNA Condition Report (X'1532') GDS variable is built with the following values:

   a. Sense Data (X'7D') subvector is taken from the Sense Data field of the MDS Routing Unit.

   b. Reported On Location Name and Reported On Agent are selected as

follows:

- If MDS error message required is set to ORIGIN, they are taken from the *Destination* Location Name in the MDS Routing Unit.

- If MDS error message required is set to DESTINATION, they are taken from the *Origin* Location Name in the MDS Routing Unit.

5. If the sense data is X'08A8 0009' (destination not supported by reporting network node) *and* if the Routing Verification Indicator of the Flags (X'90') subvector of the MDS Routing Information (X'1311') GDS variable is 0, indicating that routing was based on nonverified directory information, then the original MDS-MU is copied into the MDS Error Message after the SNA Condition Report (X'1532') GDS variable.

   This is done so that the previous MDS_ROUTER can request a verified directory search to update the directory services cache and perhaps reroute the MDS-MU if a new location is found for the destination end node.

6. If the Alert Required indicator in the MDS Routing Unit is YES, EP_ALERT is invoked to build "Management Services Alert CPMS002" in topic A.5.6.1. This Alert reports MULTIPLE_DOMAIN_SUPPORT format and protocol exceptions.

7. This process exits.

Table 9-11. Contents of an MDS Error Message created by MDS_ROUTER

Structure Name	GDS/SV/SF	Description of Contents
MDS-MU	X'1310'	
MDS Routing Information	X'1311'	
Origin Location Name	X'81'	
NETID	X'01'	NETID of reporting node
NAU Name	X'02'	NAU name of reporting node
Application ID	X'03'	X'23F0F1F0' (MDS_ROUTER)
Destination Location Name	X'82'	Destination for the MDS error message (see note)
NETID	X'01'	NETID of destination node
NAU Name	X'02'	NAU name of destination node
Application ID	X'03'	Destination MS application program
Flags	X'90'	
MDS Message Type	__	MDS error message (X'02')
First MDS Message Indicator	__	1 _ first message for UOW
Last MDS Message Indicator	__	1 _ last message for UOW
Routing verification indicator	__	Value set when MDS error message is routed
Agent Unit of Work	X'1549'	Correlator of failed transaction
SNA Condition Report	X'1532'	
SNA Report Code	X'7D'	Sense data indicating nature of error (see note)
Reported on Dest Prefix	X'08'	(delimiter)

Reported on Location Name	X'09'	Node being reported upon
Reported on NETID	X'01'	NETID of node reported on
Reported on Node ID	X'02'	NAU name of node reported on
Reported on Dest Suffix	X'0B'	(delimiter)
Reported on Agent	X'04'	MS application name reported upon
MDS-MU	X'1310'	Returned MDS-MU. (see note)

Note:

For SNA Report code X'08A80009', the destination MS application program name is X'23F0F1F0' (MDS_ROUTER), and the returned MDS-MU is present. Otherwise, the returned MDS-MU is *never* present.

This extended format of an MDS error message is exchanged only between MDS_ROUTERs, and is never received by MS application programs.

Sense Data Reported by MDS in MDS Error Messages:

The sense data which may appear in MDS error messages built by this process are listed in:

- Format exceptions: Table 9-10
- Routing exceptions: Table 9-12
- Transaction exceptions: Table 9-13

Table 9-12. MDS Routing Exceptions

Sense Data	Condition
X'08A8 0001'	Destination NAU name unknown. Directory services could not locate the requested destination name.
X'08A8 0002'	Directory services unavailable. No routing possible.
X'08A8 0003'	MS application program name not recognized.
X'08A8 0004'	Use of CPSVCMG session not permitted. The reporting network node has received an MDS-MU over a CPSVCMG session from another network node. These sessions are used for MDS-MUs only between a network node and its client end nodes.
X'08A8 0005'	Function not supported by end node destination. The end node destination does not support receipt of MDS-MUs. (Reported by network node server).
X'08A8 0006'	Function not supported by network node destination. The network node destination does not support receipt of MS messages other than MS Capabilities major vector and Alert major vector. (Reported by network node performing routing.)
X'08A8 0007'	Function not supported by network node server. The network node server of the end node destination does not support routing of MDS-MUs. (Reported by network node performing routing.)
X'08A8 0008'	Function not supported by end node. The reporting end node has received an MDS-MU with a destination other than itself.
X'08A8 0009'	Destination not supported by reporting network node. A network node has received an MDS-MU from another network node that cannot be routed. The destination is not the reporting network node itself nor is it one of the client end nodes. (Note 2)
X'08A8 000A'	Irrecoverable session failure. The MDS_SEND TP in the reporting node was unable to send the message because of an allocation error. Retries have been exhausted.
X'08A8 000B'	Irrecoverable TP failure in remote node. The MDS_SEND TP in the reporting node was unable to send the message because of a TP failure in a remote node. Retries have been exhausted.

X'08A8 000C'	MS application program failure. The MDS_ROUTER for the destination NAU is unable to communicate with the destination MS application program.
X'08A8 000F'	MS application program congestion. The MDS_ROUTER for the destination NAU is unable to communicate with the destination MS application program because of local congestion (implementation buffer space for queuing additional MDS-MUs has been exhausted).
X'08A8 000D'	Irrecoverable TP failure in reporting node. The MDS_ROUTER for the reporting node was unable to send the message because of a local TP failure.
X'08A8 000E'	Correlation error. An MDS-MU has been received that is not the first for a unit of work (First MDS message indicator in the MDS Routing Information is OFF), but the unit of work correlator is unknown (does not match any active MDS transaction).

Notes:

1. The sense data is sent in the Sense Data (X'7D') subvector of the SNA Condition Report (X'1532') GDS variable. This GDS variable is included in an MDS error message.

2. For sense data X'08A80009', if the MDS-MU was routed based on nonverified directory information (as indicated by the Routing verification indicator in the Flags (X'90') subvector of the MDS Routing Information (X'1311') GDS variable), the MDS-MU will be returned in the MDS error message after the SNA Condition Report (X'1532') GDS variable.

Table 9-13. MDS Transaction Exceptions

Sense Data	Condition
X'08A9 0001'	Failure caused by outage of CPSVCMG session.
X'08A9 0002'	Failure caused by outage of SNASVCMG session. All retries have been exhausted.
X'08A9 0004'	Unit of work canceled by reporting MDS_ROUTER. The unit of work has been canceled by a garbage collection timeout in the reporting MDS_ROUTER.
X'08A9 0005'	MDS Router internal failure. The unit of work has been canceled because of an internal failure in the reporting MDS Router.
X'08A9 0007'	MS Application router reinitialization. The unit of work has been canceled by the reporting MDS Router because of a re-initialization of the application-level router.

Notes:

1. The sense data is sent in the Sense Data (X'7D') subvector of the SNA Condition Report (X'1532') GDS variable. This GDS variable is included in an MDS error message.

MDS_SEND Manager

Process Name	MDS_SEND Manager
Description	This process creates instances of the MDS_SEND TP when needed to process outbound MDS Routing Units.
Input	"MDS Internal Signal - MDS Routing Unit" in topic 2.9.2.4.4
Output	° Instance of MDS_SEND TP started if necessary ° MDS Routing Unit passed to MDS_SEND for routing on a session. ° If unable to start a TP or to pass an MDS Routing Unit to a TP, MDS Routing Unit is updated with routing status is ERROR, and "Processing MDS Routing Unit (REPLY TO OUTBOUND)" is invoked to handle error processing.

An implementation of MULTIPLE_DOMAIN_SUPPORT must be designed to guarantee FIFO delivery of MDS-MUs. This requirement is met in the architecture model in the following manner:

- This process of the MDS_ROUTER starts a copy of the MDS_SEND TP when it has an MDS-MU to be sent on a session. However, only a single copy of the TP is started for any given session partner at a given time. Creating multiple instances of the TP for a given session opens the window for non-FIFO delivery, since there is no guarantee in most LU 6.2 implementations that TPs waiting for a single resource (a session) are dispatched in exactly the same order as they were created.

- When this process receives additional MDS-MUs that must be routed by a currently active instance of MDS_SEND, it places them in a queue which is served by the TP. As the TP successfully sends each MDS-MU, it removes it from the queue.

- By ensuring FIFO processing on the send side, no special considerations are needed on the receive side, except that the MDS_ROUTER process the received messages in the order in which they were received.

This process is illustrated in Figure 9-37.

```
                              (MDS Routing Unit)
                                      |

                            _________________
                           | Is there an active |
                 _________|copy of MDS_SEND for|
                |      yes |   the indicated    |
                |          | outbound session?  |
                |          |____________________|
                |  ___                | no
                | | 1 |___________ÿ|
                | |___|
                |                                    °°°°°°°°°°°°°°°°°°°°°°°
                |           _________________        °                     °
                |          |    Request that    |    °
                |          |MDS_SEND be started |_____ÿ°CP Resources Manager°
                |          |                    |    °                     °
                |          |____________________|    °°°°°°°°°°°°°°°°°°°°°°°
                |                     |
                |
                |           _________________        ___
                |          |   TP started OK?   |_____ÿ| 3 |
                |          |____________________| no   |___|
                |                     |yes
                |
                |           _________________
                |          |     Add new TP     |
                |          |  instance to list  |
                |          |   of active TPs    |
                |          |____________________|
                |                     |
                |_________________ÿ|

                                                    °°°°°°°°°°°°°°°°°°°°°°°
                            _________________       °                     °
                           |  Send MDS Routing  |   °
                           |  Unit to MDS_SEND  |_____ÿ°    MDS_SEND TP     °
                           |                    |    °                     °
                           |____________________|    °°°°°°°°°°°°°°°°°°°°°°°
                                     |

                                      ___
                                     | 2 |
                                     |___|
```

Figure 9-37. MDS_SEND Manager (Part 1 of 2)

The processing is described in the steps below:

1. The list of active instances of MDS_SEND is searched for a match with the outbound session information in the MDS Routing Unit (OUTBOUND)

2. If a match is not found, a new copy of MDS_SEND must be started by the CP resources manager. The manner in which a TP is started is implementation-specific.

3. If the TP was started OK, its identity and associated outbound session information is added to the list of active TPs.

4. The MDS Routing Unit (OUTBOUND) is sent to the TP.

```
                                      ___
                                     | 2 |
                                     |___|
```

```
                                  |

                            _________________
                           |   Was send to TP  |_____
                           |    successful?    | yes |
                           |___________________|     |
                                     | no            |
                                                   |
                            _________________       |
                           | Delete TP instance |    |
                           |     from list      |    |
                           |____________________|    |
                                     |               |
                                                   |
                            _________________       |
                           |  Was TP instance  |     |
                     ___   | taken from list or |    |
                    | 1 |_______|                    |
                    |___|  list |   just started    |     |
                           |____________________|    |
                     ___                 | just started   |
                    | 3 |_________________ÿ|               |
                    |___|                              |
                            _________________       |
                           | Update MDS Routing |    |
                           |Unit w/status ERROR,|    |
                           |sense X'08A8 000D', |    |
                           |     direction      |    |
                           | REPLY TO OUTBOUND  |    |
                           |____________________|    |
                                     |               |
                                                   |
                            _________________       |
                           ||    Processing    ||    |
                           ||'MDS Routing Unit ||    |
                           ||    (Reply to     ||    |
                           ||    Outbound)'    ||    |
                           ||__________________||    |
                                     |               |
                                     |_______________|

                                  (exit)
```

Figure 9-38. MDS_SEND Manager (Part 2 of 2)

5. If the send was successful, this process exits.

6. If the send to the TP was not successful:

   a. The TP instance is deleted from the list.

   b. If the TP information had been taken from the list rather initially, rather than having been just started, processing resumes at step 1. Since there is no signal to the MDS_ROUTER when a TP instance ends, the presumption is that the list entry was old.

   c. If the TP was just started:

      1) The MDS Routing Unit is updated:

         - Direction is set to REPLY TO OUTBOUND
         - Routing status is set to ERROR
         - Sense data is set to X'08A8 000D'

      2) "Processing MDS Routing Unit (REPLY TO OUTBOUND)" is invoked.

7. This process exits.

| *Back-Level Node Support*

Process Name	Back-Level Node Support
Description	This process maps CP-MSUs inbound from back-level nodes to MDS-MUs, which will then be processed normally by MULTIPLE_DOMAIN_SUPPORT. In the outbound direction, the only mapping required is the extraction of the CP-MSU from the MDS-MU. This simple mapping is preformed by the MDS_SEND TP. In order to perform the required mapping of inbound Alerts to the current Alert focal point, Back-level Node Support must register itself (i.e., register MDS_ROUTER) with MS_CAPS for the Alert category and

	then remember the focal point information supplied by MS_CAPS
Input	The input to this process is one of the following: ° "Protocol Boundary O--Register Application Program Name" in topic 2.9.1.11 (for MS_CAPS only) ° "Protocol Boundary V--Current Focal Point" in topic 2.9.1.17 ° "MDS Internal Signal - MDS Routing Unit" in topic 2.9.2.4.4
Output	The output from this process is one of the following: ° "Protocol Boundary T--Register MS Category" in topic 2.9.1.15 is prepared and forwarded to MS_CAPS ° Current focal point for Alerts is remembered. ° Inbound CP-MSU is mapped to MDS-MU.

This process is illustrated in Figure 9-39.

Figure 9-39. Back-Level Node Support (Part 1 of 4)

The three connector boxes in the illustration are found on the following pages:

1. Page 2.9.2.4.8
2. Page 2.9.2.4.8
3. Page 2.9.2.4.8

Back-level Node Support Preparation of Register MS Category

Figure 9-40. Back-Level Node Support (Part 2 of 4)

This process is illustrated in Figure 9-40. The processing is described in the following steps:

1. Protocol Boundary T--Register MS Category is prepared as described in "Application Program Registration with MS_CAPS" in topic 2.9.3.10 for category EP_ALERT.

2. The signal is passed to MS_CAPS.

3. This process exits.

Back-level Node Support Processing of Current Focal Point

```
                                  ___
                                 | 2 |
                                 |___|
                                   |
                          (Current Focal Point)
                                   |
                                   |

                        _________________
                       |   Save name of   |
                       | Alert Focal Point |
                       |                  |
                       |__________________|
                                   |

                                 (exit)
```

Figure 9-41. Back-Level Node Support (Part 3 of 4)

This process is illustrated in Figure 9-41. The processing is described in the following steps:

1. Protocol Boundary V--Current Focal Point is received and the remembered focal point name for Alerts is updated.

2. This process exits.

Back-level Node Support Processing of Inbound CP-MSUs

```
                                  ___
                                 | 3 |
                                 |___|
                                   |
                        (MDS Routing Unit (Inbound))
                                   |

                        _________________      _________________
                       | Is received GDS  |    | Update MDS routing |
                       |     a CP-MSU?    |____ÿ| unit w/sense data  |_____
                       |                  | no |   X'080C 0012',    |     |
                       |__________________|    |Routing Status ERROR|     |
                                 | yes         |____________________|     |
                                                                          |
                        _________________      _____________________      |
                       | Select on major  |    | Update MDS routing |     |
                _______|    vector key    |____ÿ| unit w/sense data  |____ÿ|
               |  Alert |                  | other |   X'080C 0005',    |     |
               |        |__________________|    |Routing Status ERROR|     |
               |                 |             |____________________|     |
               |                 | MS-Capabilities                        |
                                                                          |
     _________________  _________________      _____________________      |
    | Is there a current |  | Select on MS-Caps |    | Update MDS routing |     |
    |    focal point?    |  |       type        |____ÿ| unit w/sense data  |____ÿ|
    |                    |  |                   | other |   X'080C 0006',    |     |
    |____________________|  |___________________|    |Routing Status ERROR|     |
          |    |            FP      |        |      |____________________|     |
          |    |            Authorization    |   FP Authorization Reply        |
          |    | yes        Request |        |____________________             |
            |___________                                                    |
     _________________  |  _________________      _____________________      |
    | Update MDS routing |  | Build MDS Routing |   ___|      Revoke?       |     |
    | unit w/sense data  |  |Info. w/dest. local |  |no |___________ _______|     |
    |   X'080C 0000',    |  |node MS_CAPS, origin|  |                yes        |
    |Routing Status ERROR|  |  session partner  |  |        _____________________ |
    |____________________|  |      MS_CAPS      |  |       | Build MDS Routing  |  |
 ___________|               |___________________|  |       |Info. w/dest. local |  |
|          _______________|      |                        |node MS_CAPS, origin|_ÿ|
|          |                     |        _________________ |  session partner  |  |
|          |                _____|   ||  Processing of   ||  |      MS_CAPS      |  |
|          |               |         ||      Active      ||  |___________________|  |
|          |               |         || Transaction List ||                       |
|          |               |         ||_________________||                       |
|          |               |       entry found |    | error                       |
|          |               |            ______|    |________________             |
|                         |                                                 |
|     _________________  |  _________________      _____________________      |
|    | Build MDS Routing |  | Build MDS Routing |    | Update MDS routing |     |
|    |     Info. with    |  | Info. from Active |    | unit w/sense data  |     |
|    |  current FP name  |  |  Transaction List |    |   X'080C 000E',    |     |
|    |___________________|  |       entry       |    |Routing Status ERROR|     |
|          |               |___________________|    |____________________|     |
|          |               |          |                        |              |
```

```
| |                                    |                                          |
| |____________________________________ÿ|__________________________________________|
|
|                                      (exit)
```

Figure 9-42. Back-level Node Support (Part 4 of 4)

An MDS Routing Unit (INBOUND) from a session partner with MDS Support Level of BACK-LEVEL must be examined so that the appropriate MDS Routing Information (X'1311') GDS variable can be built to route the message to its destination.

Back-level nodes support only one GDS variable, CP-MSU, and two major vectors:

- MS Capabilities (X'80F0') major vector
- Alert (X'0000') major vector

This process is illustrated in Figure 9-42. The processing done is described below:

1. The MDS-MU is built:

   a. If major vector key is X'0000' (Alert), the MDS Routing Information (X'1311') GDS variable is built as follows:

      - Origin identification (X'81') subvector

        - NETID (X'01') subfield

          NETID portion of Network-qualified LU name of session partner.

        - NAU Name (X'02') subfield

          LU portion of Network-qualified LU name of session partner.

        - MS Application Program Name (X'03') subfield

          X'23F0F3F0' (EP_ALERT)

      - Destination Location Name (X'82') subvector

        - NETID (X'01') subfield

          NETID portion of Alert focal point name

        - NAU Name (X'02') subfield

          NAU portion of Alert focal point name

        - MS Application Program Name (X'03') subfield

          MS application program name of Alert focal point

      - Agent Unit-of-Work Correlator

        A unique correlator must be built. The node name and agent name in the correlator need not reflect the identification of the back-level node.

      - MS Application Program GDS variable is the CP-MSU

   b. If major vector key is X'0000' (Alert), but there is no current focal point, the following is done:

      1) Routing Status in the MDS Routing Unit is set to ERROR.

      2) CP-MSU is discarded (leaving a null MDS-MU in MDS Routing Unit)

         Since there is no Alert focal point, the inbound Alert cannot be processed any further. The Alert was logged locally by the originating CP, so it is not lost.

```
|     c.  If major vector key is MS Capabilities (X'80F0') and the first
|         subvector is Focal Point Authorization Request (X'61') subvector,
|         the MDS Routing Information (X'1311') GDS variable is built as
|         follows:

|         °   Origin Location Name (X'81') subvector

|             -   NETID (X'01') subfield

|                 NETID portion of Network-qualified LU name of session
|                 partner.

|             -   NAU Name (X'02') subfield

|                 LU portion of Network-qualified LU name of session
|                 partner.

|             -   MS Application Program Name (X'03') subfield

|                 X'23F0F1F1' (MS_CAPS).

|         °   Destination Location Name (X'82') subvector

|             -   NETID (X'01') subfield

|                 NETID portion of Network-qualified LU name of LU in this
|                 node.

|             -   NAU Name (X'02') subfield

|                 LU portion of Network-qualified LU name of LU in this
|                 node.

|             -   MS Application Program Name (X'03') subfield

|                 X'23F0F1F1' (MS_CAPS).

|         °   Agent Unit-of-Work Correlator

|             A unique correlator must be built.  The node name and agent
|             name in the correlator need not reflect the identification of
|             the back-level node.

|         °   MS Application Program GDS variable is the CP-MSU

|     d.  If major vector key is MS Capabilities (X'80F0') and the first
|         subvector is Focal Point Authorization Reply (X'62') subvector,
|         the following is done:

|         1)  If the X'62' subvector contains the Authorization Revoked
|             (X'30') subfield, then this MS Capabilities major vector was
|             sent unsolicited by the back-level entry point to revoke its
|             current focal point (this node).  An MDS-MU is built as
|             follows:

|             °   Origin identification (X'81') subvector

|                 -   NETID (X'01') subfield

|                     NETID portion of Network-qualified LU name of session
|                     partner.

|                 -   NAU Name (X'02') subfield

|                     LU portion of Network-qualified LU name of session
|                     partner.

|                 -   MS Application Program Name (X'03') subfield

|                     X'23F0F1F1' (MS_CAPS).

|             °   Destination Location Name (X'82') subvector

|                 -   NETID (X'01') subfield
```

```
|                    NETID portion of Network-qualified LU name of LU in
|                    this node.

|                -   NAU Name (X'02') subfield

|                    LU portion of Network-qualified LU name of LU in this
|                    node.

|                -   MS Application Program Name (X'03') subfield

|                    X'23F0F1F1' (MS_CAPS).

|            °   Agent Unit-of-Work Correlator

|                A unique correlator must be built.  The node name and
|                agent name in the correlator need not reflect the
|                identification of the back-level node.

|            °   MS Application Program GDS variable is the CP-MSU
|                containing the MS Capabilities major vector.

|        2)  If the Focal Point Authorization Reply (X'62') subvector does
|            not contain the Authorization Revoked (X'30') subfield, then
|            this MS Capabilities major vector was sent as a reply to a
|            Focal Point Authorization Request (X'61') subvector request.
|            The following is done:

|            a)  Back-level node reply correlation required is set to YES

|            b)  "Processing of Active Transaction List" is invoked, so
|                that the Active Transaction List entry for the request can
|                be found.

|            c)  If the Active Transaction List entry is found, the MDS-MU
|                is constructed as follows:

|                i.  MDS Routing Information (X'1311') GDS variable fields:

|                    °   Origin Location Name (X'81') subvector

|                        -   NETID (X'01') subfield

|                            NETID portion of Destination field in Active
|                            Transaction List entry.

|                        -   NAU Name (X'02') subfield

|                            NAU portion of Destination field in Active
|                            Transaction List entry.

|                        -   MS Application Program Name (X'03') subfield

|                            MS application program portion of Destination
|                            field in Active Transaction List entry.

|                    °   Destination Location Name (X'82') subvector

|                        -   NETID (X'01') subfield

|                            NETID portion of Origin field in Active
|                            Transaction List entry.

|                        -   NAU Name (X'02') subfield

|                            NAU portion of Origin field in Active
|                            Transaction List entry.

|                        -   MS Application Program Name (X'03') subfield

|                            MS application program portion of Origin field
|                            in Active Transaction List entry.

|                ii. Agent Unit-of-Work Correlator (X'1549') GDS variable
|                    copied from Correlator field of Active Transaction
|                    Entry.
```

```
|                iii. MS Application Program GDS variable is the CP-MSU

|            d)  If "Processing of Active Transaction List" returns an
|                error indication (i.e., no corresponding Active
|                Transaction List entry found):

|                °   Routing Status in the MDS Routing Unit is set to
|                    ERROR.

|                °   Sense Data is set to X'08A8 000E' (correlation error).

|                °   MDS error message required is set to NO (the
|                    back-level node would not understand an MDS error
|                    message, and there is no corresponding error
|                    notification).

|                °   Alert required is set to YES.

|     e.  If the received MS application program GDS variable is none of the
|         above, the following is done:

|         °   Routing Status in the MDS Routing Unit is set to ERROR.

|         °   Sense Data is set to one of:

|             -   X'080C 0012' (MS application program GDS variable not
|                 supported)

|             -   X'080C 0005' (major vector key not supported)

|             -   X'080C 0006' (command subvector not supported)

|         °   MDS error message required is set to NO (the back-level node
|             would not understand an MDS error message, and there is no
|             corresponding error notification).

|         °   Alert required is set to YES.

| 2.  This process exits.
```

Error Processing by MDS_ROUTER:

Figure 9-43. Sources of MDS_ROUTER Error Notification

The MDS Router receives the following types of error notification, illustrated in Figure 9-43. See the referenced sections for a description of the processing done in each case.

- MDS Error Message (from local MS application program)

  "Processing Send MDS-MU."

- MDS Error Message (from remote MS application program or MDS_ROUTER)

  "Processing MDS Routing Unit (Inbound) from TPs."

- Session outage notification for SNASVCMG sessions

  "Processing SNASVCMG SON."

- Session allocation error

  "Processing MDS Routing Unit (REPLY TO OUTBOUND)."

- Session outage notification for CPSVCMG sessions

  "Processing CP_STATUS Signal."

- Directory Services Locate errors (control point directory services)

  "Directory Services Processing" in topic 2.9.2.6.2.

- Runtime errors (failure of interprocess communication)

  "Routing MDS-MU Received to Local MS Application Programs."

2.9.2.4.9 Overview of MDS Transaction Programs

```
 _____________________________________________________________________
| |                                                                   |
| |                          M D S _ R O U T E R                      |
| |                                                                   |
| |___________________________________________________________________|
|        "                   "                   |                    "
|        |                   |                   |                    |
| SNASVCMG|        MDS Routing |        MDS Routing |         MDS Routing |
|   SON  |      Unit (inbound) |    Unit (outbound) |       Unit (inbound) |
|        |                   |                   |                    |
|        |                   |                                       |
|        |             _____________       _____________      _____________
| _____________     _____________   |   _____________   |   _____________  |
| |   MDS_SON   |   | MDS_RECEIVE  | |   |   MDS_SEND   | |   | MSU_HANDLER  | |
| |             |   |  X'23F0F0F1' |__|   |              |__|   |  X'23F0F0F0' |__|
| |_____________|   |______________|      |______________|      |______________|
|        "                  "                   |                    "
|        |      RECEIVE_AND_|           SEND_DATA |         RECEIVE_AND_|
|   PIP  |      WAIT (after |            (after   |         WAIT (after |
|   Data |        Attach)   |           ALLOCATE) |           Attach)   |
|        |                  |                                        |
| _____________________________________________________________________
| |                                                                   |
| |                       CP or LU presentation services              |
| |                                                                   |
| |___________________________________________________________________|
|      o o               o o                   o o                  o o
|      o o               o o  CP-CP (CPSVCMG)  o o                  o o
|      o o               o o         and       o o                  o o
|      o o               o o LU-LU (SNASVCMG)  o o                  o o
|      o o               o o     sessions      o o                  o o
|      o o               o o                   o o                  o o
```

Figure 9-44. Mulitiple-Domain Support Transaction Programs. Arrows illustrate the direction of data flow, not control information.

Figure 9-44 illustrates the MDS service transaction programs and their relationship to the CP and LU, the sessions, and the MDS router. Information about these transaction programs is summarized in Table 9-14.

Table 9-14. Multiple Domain Support LU 6.2 Transaction Programs

| Name (hex) | Descriptive | Function |

	name	
X'23F0F0F0'	MSU_HANDLER	Performs application routing by MS major vector key. Implemented by back-level nodes to send and receive CP-MSUs. MDS_SEND provides back-level compatibility so that APPN end nodes can send alerts to back-level network nodes. Since these back-level network nodes will not attempt to send CP-MSUs to end nodes, an end node does not need to provide the MSU_HANDLER TP itself, however.
	MDS_SEND	MULTIPLE_DOMAIN_SUPPORT program for sending MDS-MUs. This TP is never the target of an Attach, so no registered name is required.
X'23F0F0F1'	MDS_RECEIVE	MULTIPLE_DOMAIN_SUPPORT transaction program for receiving MDS-MUs.
	MDS_SON	MULTIPLE_DOMAIN_SUPPORT transaction program for processing session-outage notification (SON). This TP is never the target of an Attach, so no registered name is required.

LU 6.2 Verb Sequences:

The following diagrams illustrate the sequences of LU 6.2 verbs that make up the interactions of the MDS_SEND MDS_RECEIVE, and MSU_HANDLER service transaction programs.

These illustration are based on the following implementation assumptions. Other implementation designs are possible.

1. MDS_SEND issues SEND_DATA for a buffer that contains the entire MDS-MU.

   An equally valid implementation choice would be to issue SEND_DATA multiple times, sending a portion of the MDS-MU each time. This might be necessary because of implementation restrictions on send buffer size, for example. The MDS_RECEIVE TP in another implementation will not be aware of the method used by the sending TP.

2. MDS_RECEIVE issues RECEIVE_AND_WAIT FILL(LL) one or more times as needed to receive the MDS-MU. An MDS-MU longer than the receive buffer would force the TP to issue RECEIVE_AND_WAIT multiple times (until DATA_COMPLETE were indicated by the LU). The application program logical record is reassembled by MDS_RECEIVE as the parts of the record are received.

   An equally valid implementation choice would be to issue RECEIVE_AND_WAIT one time with a buffer length of 2, just to get the total length of the MDS-MU, then to allocate a buffer of exactly the right size so that RECEIVE_AND_WAIT can be issued a single time for the rest of the MDS-MU.

These illustrations do not show all the LU 6.2 verb parameters required in an actual implementation, nor do they necessarily show the exact syntax required. More information is given in the descriptions of the TPs.

Single MDS-MU on a Conversation:

```
 ____________________________________________________________________
|                                                                    |
|        MDS_SEND                                MDS_RECEIVE         |
|        __________________________________________________________|__
| 1      Allocate                                                    |
|          .                                                         |
|          .                                                         |
|          RC=OK                                                     |
| 2      Send_data                                                   |
|          RC=OK                                                     |
| 3      Deallocate(confirm) _______                                 |
| 4        .                       |___________ÿ Receive_and_Wait   |
|          .                                     RC=OK               |
|          .                                     What=DATA   (MDS-MU) |
|          .                                        . . .            |
| 5        .                                   Receive_and_Wait      |
|          .                                     RC=OK               |
```

```
|          .                                    What=Confirm_dealloca|e
| 6        .                          ___________ Confirmed          |
|          .                     |                RC=OK             |
| 7        .                     |            Deallocate(local)     |
|          .                     |                RC=OK             |
| 8        RC=OK ________________|                                  |
|                                                                   |
|                                                                   |
|___________________________________________________________________|
Figure  9-45. Verb Sequence: Single MDS-MU on a Conversation
```

Figure 9-45 illustrates the simplest interaction between MDS_SEND and MDS_RECEIVE, the sending of a single MDS-MU.

In this case, the sending implementation has chosen never to send more than one MDS-MU per conversation. This implementation is permitted by the architecture, but it does not provide an optimal use of system resources, except in very low traffic situations.

1. A conversation is allocated

2. A single MDS-MU is sent

3. DEALLOCATE (CONFIRM) is requested

4. The MDS-MU is received

   MDS_RECEIVE uses FILL(LL) processing to receive the MDS-MU and application program logical record. Multiple receives may be necessary to receive the entire logical record (dependent upon implementation buffer size).

5. The request for CONFIRM_DEALLOCATE is received

6. The MDS-MU is CONFIRMED

7. MDS_RECEIVE DEALLOCATES the conversation locally

8. The conversation is DEALLOCATED

Multiple MDS-MUs on a Conversation:

```
 ___________________________________________________________________
|                                                                   |
|        MDS_SEND                                MDS_RECEIVE        |
|        __________________________________________________________|__
| 1      Allocate                                                   |
|          .                                                        |
|          .                                                        |
|          RC=OK                                                    |
| 2      Send_data (MDS-MU)                                         |
|          RC=OK                                                    |
| 3      Confirm _________________                                  |
| 4        .                     |_______________ÿ Receive_and_Wait |
|          .                                        RC=OK           |
|          .                                        What=DATA   (MDS-MU) |
|          .                                           . . .        |
| 5        .                                      Receive_and_Wait  |
|          .                                        RC=OK           |
|          .                                        What=Confirm    |
| 6        .                      _________________ Confirmed      |
|          .                     |                  RC=OK           |
| 7        .                     |                Receive_and_Wait  |
| 8        RC=OK _________________|                 .              |
| 9      Send_data (MDS-MU)                          .              |
|          RC=OK                                     .              |
| 10     Confirm _________________                   .              |
|          .                     |                   .              |
| 11       .                     |_______________ÿ   RC=OK          |
|          .                                        What=DATA   (MDS-MU) |
| 12       .                                      Receive_and_Wait  |
|          .                                        RC=OK           |
|          .                                        What=Confirm    |
| 13       .                      _________________ Confirmed      |
|          .                     |                  RC=OK           |
| 14                             |                Receive_and_Wait  |
| 15       RC=OK _________________|                 .              |
| 16     Deallocate(flush) ______                    .              |
|          RC=OK                 |                   .              |
| 17                             |_______________ÿ   RC=Deallocate_normal |
| 18                                              Deallocate(local)  |
|                                                    RC=OK          |
|                                                                   |
|___________________________________________________________________|
Figure  9-46. Verb Sequence: Multiple MDS-MUs on a Single Conversation
```

Figure 9-46 illustrates the sending of multiple MDS-MUs on a single conversation.

1. A conversation is ALLOCATED

2. The first MDS-MU is sent

3. CONFIRM is requested

4. The MDS-MU is received

   MDS_RECEIVE uses FILL(LL) to receive the MDS-MU. Multiple receives may be necessary to receive the entire logical record (dependent upon implementation buffers).

5. The request for CONFIRM is received

6. The CONFIRMED is sent

7. MDS_RECEIVE goes into receive state

8. The CONFIRMED is received

9. The second MDS-MU is sent

10. CONFIRM is requested

11. The MDS-MU is received

12. The request for CONFIRM is received

13. The CONFIRMED is sent

14. MDS_RECEIVE goes into receive state

15. The CONFIRMED is received

16. DEALLOCATE (FLUSH) is issued, since there are no additional MDS-MUs to send.

17. The DEALLOCATE_NORMAL indication is received.

18. MDS_RECEIVE DEALLOCATES the conversation locally.

MDS_RECEIVE TP Failure:

```
 ____________________________________________________________________
|                                                                    |
|        MDS_SEND                                MDS_RECEIVE         |
|        __________________________________________________________|__
| 1      Allocate                                                    |
|          .                                                         |
|          .                                                         |
|          RC=OK                                                     |
| 2      Send_data                                                   |
|          RC=OK                                                     |
| 3      Confirm _________________                                   |
| 4        .                     |______________ÿ Receive_and_Wait  |
|          .                                        RC=OK            |
|          .                                        What=DATA   (MDS-MU) |
|          .                                           . . .         |
| 5        .                                      Receive_and_Wait   |
|          .                                        RC=OK            |
|          .                                        What=Confirm     |
| 6        .                      _________________ Confirmed       |
|          .                    |                   RC=OK            |
| 7        .                    |                 Receive_and_Wait   |
| 8        RC=OK _________________|                  .                |
| 9      Send_data                                   .                |
|          RC=OK                                     .                |
| 10     Confirm _________________                   .                |
| 11       .                     |_______________ÿ   RC=OK          |
|          .                                        What=DATA (part of MD|-MU)
|          .                                                         |
| 12       .                                       *** Error ***     |
|          .                                       (program failure) |
```

```
|          .                                                        |
| 13       .                        _________________ Deallocate(Abend_Prog) |
|          .                      |                  RC=OK                 |
| 14       RC=Dealloc_ _________|                                        |
|             Abend_Prog                                                 |
| 15     Deallocate(local)                                               |
|          RC=OK                                                         |
|                                                                        |
|                                                                        |
|________________________________________________________________________|
```

Figure 9-47. Verb Sequence: MDS_RECEIVE TP Failure (Part 1 of 2)

Figure 9-47 illustrates an error case. The first MDS-MU was successfully received and confirmed. The second MDS-MU was not successfully received because of a program failure that caused MDS_RECEIVE to issue DEALLOCATE(ABEND). MDS_SEND attempts to resend the MDS-MU. If the second attempt had also failed, MDS_SEND would reported a send failure to the MDS_ROUTER.

1. A conversation is ALLOCATED

2. The first MDS-MU is sent

3. The MDS-MU is received

4. CONFIRM is requested

5. The request for CONFIRM is received

6. The CONFIRMED is sent

7. MDS_RECEIVE goes into receive state

8. The CONFIRMED is received

9. The second MDS-MU is sent

10. CONFIRM is issued.

11. The first buffer of the MDS-MU is received.

12. MDS_RECEIVE experiences a program error (e.g., unable to allocate storage)

13. MDS_RECEIVE issues DEALLOCATE (ABEND_PROG).

14. MDS_SEND receives the error indication as the return code from DEALLOCATE

15. MDS_SEND issues DEALLOCATE (LOCAL) to prepare for another allocation attempt.

```
 ___________________________________________________________________
|                                                                   |
|        MDS_SEND                                MDS_RECEIVE        |
|        __________________________________________________________|__
| 16     Allocate                                                   |
|          .                                                        |
|          .                                                        |
|          RC=OK                                                    |
| 17     Send_data                                                  |
|          RC=OK                                                    |
| 18     Confirm _________________                                  |
| 19       .                    |_____________ÿ Receive_and_Wait   |
|          .                                      RC=OK             |
|          .                                      What=DATA   (MDS-MU) |
|          .                                         . . .          |
| 20       .                                    Receive_and_Wait    |
|          .                                      RC=OK             |
|          .                                      What=Confirm      |
| 21       .                    _________________ Confirmed        |
|          .                    |                 RC=OK             |
| 22       .                    |                Receive_and_Wait   |
|          .                    |                  .               |
| 23       RC=OK _______________|                 .                |
| 24     Deallocate(flush) ______                  .               |
|          RC=OK                |                  .               |
| 25                            |_______________ÿ  RC=Deallocate_normal |
| 26                                             Deallocate(local)  |
|                                                   RC=OK           |
|                                                                   |
|                                                                   |
|___________________________________________________________________|
```

Figure 9-48. Verb Sequence: MDS_RECEIVE TP Failure (Part 2 of 2)

16. MDS_SEND attempts to retry by allocating another conversation

17. The second MDS-MU is sent again

18. CONFIRM is issued

19. The entire MDS-MU is received this time.

20. The request for CONFIRM is received

21. The CONFIRMED is sent

22. MDS_SEND issues DEALLOCATE (FLUSH).

23. MDS_RECEIVE receives the DEALLOCATE_NORMAL return code.

24. MDS_RECEIVE DEALLOCATES locally

Session Outage:

```
 ___________________________________________________________________
|                                                                   |
|       MDS_SEND                                MDS_RECEIVE         |
|       __________________________________________________________|__
| 1     Allocate                                                    |
|         .                                                         |
|         .                                                         |
|         RC=OK                                                     |
| 2     Send_data                                                   |
|         RC=OK                                                     |
| 3     Confirm _______________                                     |
| 4       .                    |_____________ÿ Receive_and_Wait    |
|         .                                      RC=OK              |
|         .                                      What=DATA    (MDS-MU)|
|         .                                         . . .           |
| 5       .                                    Receive_and_Wait     |
|         .                                      RC=OK              |
|         .                                      What=Confirm       |
| 6       .                     _______________ Confirmed          |
|         .                    |                 RC=OK              |
| 7       .                    |               Receive_and_Wait     |
| 8       RC=OK _______________|                 .                 |
| 9     Send_data                                 .                  |
|         RC=OK                                   .                  |
| 10    Confirm _______________ÿ ° ° °            .                  |
|         .                                       .                  |
| 11      .                  *** Error ***        .                  |
|         .                 (session outage)      .                  |
|         .                        | |            .                  |
| 12      .                        | |__________ÿ RC=Resource_       |
|         .                        |                Failure_retry   |
| 13      RC=Resource_ ___________|                                 |
|           Failure_retry                                           |
| 14    Deallocate(local)                                           |
|         RC=OK                                                     |
|                                                                   |
|                                                                   |
|___________________________________________________________________|
```

Figure 9-49. Verb Sequence: Session Outage (Part 1 of 2)

Figure 9-49 illustrates another error case. The first MDS-MU was successfully received and confirmed. The second MDS-MU was not successfully received because of a session outage. MDS_SEND attempts to re-allocate the conversation and resend the MDS-MU. If the second attempt had also failed, MDS_SEND would reported a send failure to the MDS_ROUTER.

1. A conversation is ALLOCATED

2. The first MDS-MU is sent

3. CONFIRM is requested

4. The MDS-MU is received

5. The request for CONFIRM is received

6. The CONFIRMED is sent

7. MDS_RECEIVE goes into receive state

8. The CONFIRMED is received

9. The second MDS-MU is sent

10. CONFIRM is requested

11. Session outage occurs. The MDS-MU is lost.

12. MDS_RECEIVE sees the failure as an error RC from RECEIVE_AND_WAIT.

13. MDS_SEND sees the failure as an error RC from DEALLOCATE(CONFIRM).

14. MDS_SEND issues DEALLOCATE (LOCAL) to prepare for another allocation attempt.

```
 ____________________________________________________________________
|                                                                    |
|        MDS_SEND                                MDS_RECEIVE         |
|        __________________________________________________________|__
| 15     Allocate                                                    |
|          .                                                         |
|          .                                                         |
|          RC=OK                                                     |
| 16     Send_data                                                   |
|          RC=OK                                                     |
| 17     Confirm _________________                                   |
| 18       .                     |______________ÿ Receive_and_Wait  |
|          .                                       RC=OK             |
|          .                                       What=DATA   (MDS-MU) |
|          .                                          . . .          |
| 19       .                                     Receive_and_Wait    |
|          .                                       RC=OK             |
|          .                                       What=Confirm      |
| 20       .                      _________________ Confirmed       |
|          .                     |                 RC=OK             |
| 21       .                     |                Receive_and_Wait   |
|          .                     |                  .               |
| 22       RC=OK _________________|                 .                |
| 23     Deallocate(flush) ______                    .               |
|          RC=OK                 |                  .               |
| 24                             |______________ÿ   RC=Deallocate_normal |
| 25                                              Deallocate(local)  |
|                                                    RC=OK           |
|                                                                    |
|                                                                    |
|____________________________________________________________________|
```

Figure 9-50. Verb Sequence: Session Outage (Part 2 of 2)

15. MDS_SEND attempts to retry by allocating another conversation

16. The second MDS-MU is sent again

17. CONFIRM is requested

18. The MDS-MU is received

19. The request for CONFIRM is received

20. The CONFIRMED is sent

21. MDS_RECEIVE goes into receive state

22. MDS_SEND receives the CONFIRMED

23. MDS_SEND issues DEALLOCATE (FLUSH) since it has no more to send.

24. MDS_RECEIVE receives the DEALLOCATE_NORMAL return code.

25. MDS_RECEIVE DEALLOCATES locally

Initial Contact with Back-level Node on SNASVCMG Session:

```
|                                                                  |
|       MDS_SEND                                MSU_HANDLER        |
|       ________________________________________________________|__
| 1     Allocate TPN(MDS_RECEIVE)                                  |
|         .                                                        |
|         .                                                        |
|         RC=OK                                                    |
| 2     Send_data                                                  |
|         RC=OK                                                    |
| 3     Deallocate(confirm) ____                                   |
| 4       .                    |_______________ÿ  *** Error ***  |
|         .                     _________________ (TP name not recognized|
| 5       RC=Alloc_Error. _____|                                  |
|             TPN_Not_Recognized                                   |
| 6     Deallocate(local)                                          |
|         RC=OK                                                    |
| 7     Allocate TPN(MSU_HANDLER)                                  |
|         .                                                        |
|         .                                                        |
|         RC=OK                                                    |
| 8     Send_data (CP-MSU only)                                    |
|         RC=OK                                                    |
| 9     Deallocate(confirm) ____                                   |
| 10      .                    |_______________ÿ Receive_and_Wait |
|         .                                       RC=OK            |
|         .                                       What=DATA    (CP-MSU)|
| 11      .                                     Receive_and_Wait   |
|         .                                       RC=OK            |
|         .                                       What=(Confirm_dealloc|te)
| 12      .                     _________________ Confirmed        |
|         .                    |                  RC=OK            |
| 13      .                    |                Deallocate(local)  |
|         .                    |                  RC=OK            |
| 14      RC=OK _______________|                                  |
|                                                                  |
|                                                                  |
|__________________________________________________________________|
```

Figure 9-51. Verb Sequence: Initial Contact With Back-level Node

Figure 9-51 illustrates the technique used by MDS_SEND to recognize and communicate with a back-level node, one that implements MSU_HANDLER only. This sequence is applicable only to the SNASVCMG mode. For CPSVCMG sessions, a back-level node is recognized by the contents of CP Capabilities. The allocation for MDS_RECEIVE fails because the back-level node does not recognize the TP name MDS_SEND then attempts to ALLOCATE to MSU_HANDLER, and resends only the application program data portion of the MDS-MU (i.e., the MDS-MU GDS variable less the MDS Routing Information (X'1311') GDS variable and the Agent Unit-of-Work Correlator (X'1549') GDS variable). If the application program data GDS variable is received by MSU_HANDLER, it will respond with confirmation. MDS_SEND then reports to MDS_ROUTER that the destination node is back-level.

1. A conversation is ALLOCATED to MDS_RECEIVE

2. A single MDS-MU is sent

3. DEALLOCATE (CONFIRM) is requested

4. The remote LU does not recognize the TP name MDS_RECEIVE and returns an Attach failure.

5. MDS_SEND is informed that the TPN was not recognized.

6. MDS_SEND issues DEALLOCATE (LOCAL) to prepare for another allocation attempt.

7. A conversation is ALLOCATED to MSU_HANDLER

8. The application program GDS variable from the MDS-MU is sent (no MDS Routing Information (X'1311') GDS variable or Agent Unit-of-Work Correlator (X'1549') GDS variable)

9. DEALLOCATE(CONFIRM) is requested

10. The application program logical record (CP-MSU) is received

11. The request for CONFIRM_DEALLOCATE is received

12. The CP-MSU is CONFIRMED

13. MDS_RECEIVE DEALLOCATES locally

14. The conversation is DEALLOCATED

Processing of a Null Message from MDS Router:

```
 ___________________________________________________________________
|                                                                   |
|        MDS_SEND                                MDS_RECEIVE        |
|        _________________________________________________________|__
| 1      Allocate (When_session_allocated)                          |
|          .                                                        |
|          .                                                        |
|          RC=OK                                                    |
| 2      Deallocate(flush) _________                                |
|          RC=OK                  |                                 |
| 3                               |___________ÿ Receive_and_Wait    |
|                                                  RC=Deallocate_normal |
| 4                                            Deallocate(local)    |
|                                                 RC=OK             |
|                                                                   |
|                                                                   |
|___________________________________________________________________|
```

Figure 9-52. Verb Sequence: Null Message From MDS Router

Figure 9-52 illustrates how MDS_SEND reactivates a session on behalf of the MDS_ROUTER. See "Processing SNASVCMG SON" in topic 2.9.2.4.8 for a description of the circumstances under which this is done. The MDS_ROUTER passes MDS_SEND a null message, indicating that the TP is to ALLOCATE a conversation with the session partner (to bring up a session), but not to send any data. If the allocation is successful, the conversation is DEALLOCATED immediately without sending.

1. A conversation is ALLOCATED successfully

2. DEALLOCATE(FLUSH) is issued

3. MDS_RECEIVE issues RECEIVE_AND_WAIT, which gets a return code indicating that the conversation has been DEALLOCATED normally.

4. MDS_RECEIVE DEALLOCATES locally.

Allocation Failure:

```
 ___________________________________________________________________
|                                                                   |
|        MDS_SEND                                MDS_RECEIVE        |
|        _________________________________________________________|__
| 1      Allocate (when_session_allocated)                          |
|          .                                                        |
|          .                                                        |
|          RC=allocation_failure_retry                              |
| 2      Deallocate(local)                                          |
|          RC=OK                                                    |
| 3      Allocate (when_session_allocated)                          |
|          .                                                        |
|          .                                                        |
|          RC=allocation_failure_retry                              |
| 4      Deallocate(local)                                          |
|          RC=OK                                                    |
| 5      Allocate (when_session_allocated)                          |
|          .                                                        |
|          .                                                        |
|          RC=allocation_failure_retry                              |
| 6      Deallocate(local)                                          |
|          RC=OK                                                    |
| 7      (notify MDS router--                                       |
|         allocation failure)                                       |
|                                                                   |
|                                                                   |
|___________________________________________________________________|
```

Figure 9-53. Verb Sequence: MDS_SEND Unable To Allocate Conversation

Figure 9-53 illustrates the retries that MDS_SEND will attempt to ALLOCATE a conversation on an SNASVCMG session.

1. An attempt is made to ALLOCATE a conversation on either a contention-winner or contention-loser session. The return code indicates that the ALLOCATE failed, but can be retried.

2. MDS_SEND issues DEALLOCATE (LOCAL) to prepare for another allocation attempt.

3. An attempt is made to ALLOCATE a conversation again. The ALLOCATE fails.

4. MDS_SEND issues DEALLOCATE (LOCAL) to prepare for another allocation attempt.

5. An attempt is made to ALLOCATE a conversation a third and final time. If this ALLOCATE had succeeded, MDS_SEND would have gone on to send whatever data it had.

6. MDS_SEND issues DEALLOCATE (LOCAL) to go to reset state.

7. The MDS_ROUTER is notified of the allocation failure.

2.9.2.4.10 MDS_SEND Processing

Process Name	MDS_SEND TP
Description	This transaction program sends MDS-MUs over the sessions indicated by the MDS_ROUTER
Input	"MDS Internal Signal - MDS Routing Unit" in topic 2.9.2.4.4
Output	° MDS-MU sent over session indicated in MDS Routing Unit. ° MDS Routing Unit updated with sense data if status is ERROR. ° MDS Routing Unit updated with MULTIPLE_DOMAIN_SUPPORT support level detected in session partner. ° MDS Routing Unit updated with session status information.

This transaction program sends outbound messages as directed by the MDS_ROUTER. It uses a trial-and-error technique to identify MDS support levels at remote destinations.

The processing of this TP is illustrated in Figure 9-54 and succeeding charts. The description of the processing in each chart follows it.

MDS_SEND ALLOCATE Processing

```
     (MDS Routing Unit)
             |_____________________| 1 |
                                   |___|
    _____________________
   |For BACK-LEVEL,      |
   |  TPN=MSU_HANDLER,   |
   |otherwise,           |
   |  TPN=MDS_RECEIVE    |
   |_____________________|
             |_________________________________________
                                                       |
    _____________________                              |
   |   issue ALLOCATE    |                             |
   |_____________________|                             |
             |                                         |
                                                       |
    _____________________                              |
 ___| Return code = OK?   |                             |
|yes|_____________________|                             |
|            |no                                       |
|                                                      |
|   _____________________                              |
|  |       issue         |                             |
|  | DEALLOCATE(LOCAL)   |                             |
|  |_____________________|                             |
|            |                                         |
|                                                      |
|   _____________________  ...FAILURE_RETRY            |
|  |What is return code?|_________________            |
|  |_____________________|                |            |
|            |other                       |            |
|            |                                         |
|            |                  _____________________  |
|            |                 |  ALLOCATE retries   |____|
|            |                 |     exhausted?      | no
|            |                 |_____________________|
|            |                           |yes
|            |                                         °°°°°°°°°°°°°°°°°°°°°°°
|            |                  _____________________  °                     °
|            |_________________ÿ|       issue         |  °     MDS_ROUTER     °
|                               | DEALLOCATE(LOCAL)   |  °                     °
|____________                   |_____________________|  °°°°°°°°°°°°°°°°°°°°°°°
             |                           |                        "
                                                                 |
```

```
 _________________        _____________________        _________________
|      issue      |       | Update MDS Routing |       |  Send MDS Routing  |
|  GET_ATTRIBUTES |    __ÿ|Unit w/status ERROR,|______ÿ| Unit to MDS_ROUTER |
|                 |   |   | sense X'08A8 000A' |       |                    |
|_________________|   |   |____________________|       |____________________|
          |             |                                          |
                       |                                          |
 _________________     |   _____________________                  |
| Update MDS Routing |   |   | Take next MDS      |                 |
|Unit w/session info.|   |___| Routing Unit from  |_________________|
|                 |       | input queue        |
|_________________|       |____________________|
          |                         |queue empty

           ___                   (exit)
          | 2 |
          |___|
```

Figure 9-54. MDS_SEND Processing (Part 1 of 4)

The ALLOCATE processing is invoked when the MDS_ROUTER starts an instance of MDS_SEND and passes an MDS Routing Unit (OUTBOUND) to it.

1. For Outbound session MDS level BACK-LEVEL, the TPN used in ALLOCATE is MSU_HANDLER; otherwise (CURRENT or UNKNOWN), the TPN used is MDS_RECEIVE.

2. ALLOCATE for a contention-winner session is issued: (1)

   For SNASVCMG mode name

```
ALLOCATE LU_NAME(<session-partner>)
         MODE_NAME(SNASVCMG)
         TPN(<tpn>)
         TYPE(BASIC_CONVERSATION)
         RETURN_CONTROL(WHEN_SESSION_ALLOCATED)
         SYNC_LEVEL(CONFIRM)
```

   For CPSVCMG mode name

```
ALLOCATE LU_NAME(<session-partner>)
         CGID(<conwinner CGID>)
         TPN(<tpn>)
         TYPE(BASIC_CONVERSATION)
         RETURN_CONTROL(WHEN_CONV_GROUP_ALLOCATED)
         SYNC_LEVEL(CONFIRM)
```

3. The ALLOCATE return code is examined to determine the next step:

   a. OK

      1) GET_ATTRIBUTES is issued.

      2) The MDS Routing Unit is updated with the session CGID information.

      3) Processing continues at connector box 2, labeled successful ALLOCATION.

   b. ALLOCATION_ERROR.ALLOCATION_FAILURE_RETRY

      1) The LU is indicating that the contention-winner session could not be activated, but that an allocation retry may be successful.

      2) DEALLOCATE TYPE(LOCAL) is issued to reset the state of the TP.

      3) If the session mode name is SNASVCMG, the same ALLOCATE verb is issued up to two more times (continuing at step 2). CPSVCMG sessions are not retried.

   c. Any other return code:

      1) DEALLOCATE TYPE(LOCAL) is issued to reset the state of the TP.

      2) No further attempt is made to ALLOCATE Processing continues at step 6.

4. If ALLOCATE for the contention-winner CPSVCMG session is not successful, ALLOCATE will be issued for the contention-loser session.

```
ALLOCATE LU_NAME(<session-partner>)
         CGID(<conloser CGID>)
         TPN(<tpn>)
         TYPE(BASIC_CONVERSATION)
         RETURN_CONTROL(WHEN_CONV_GROUP_ALLOCATED)
         SYNC_LEVEL(CONFIRM)
```

5. If the allocation return code is OK:

   a. GET_ATTRIBUTES is issued.

   b. The MDS Routing Unit is updated with the session CGID information.

   c. Processing continues at connector box 2, labeled "Successful ALLOCATION."

6. If the ALLOCATE for either CPSVCMG or SNASVCMG ended any other return code:

   a. DEALLOCATE TYPE(LOCAL) is issued to reset the state of the TP.

   b. The MDS Routing Unit is updated:

      - Routing Status is set to ERROR.
      - Sense data is set to X'08A8 000A' (unrecoverable session failure)
      - Direction is set to REPLY TO OUTBOUND
      - MDS error message required is set to ORIGIN

   c. The MDS Routing Unit is sent to MDS_ROUTER.

   d. Any additional MDS Routing Units in the TP input queue are also updated in the same manner and sent back to MDS_ROUTER.

   e. The transaction program ends.

Send Processing

```
                        |___|
                                                  °°°°°°°°°°°°°°°°°°°°°°
                                 ____________________                     °                    °
                                |  Send MDS Routing  |                    °                    °
                                | Unit to MDS_ROUTER |_________ÿ°     MDS_ROUTER     °
                                |____________________|                    °                    °
                                          |                               °°°°°°°°°°°°°°°°°°°°°°

                                 ____________________
                                |  Additional MDS    |        ___
                                | Routing Units in   |____ÿ| 2 |
                                |   input queue?     | yes |___|
                                |____________________|
                                          |no

                                 ____________________
                                |       issue        |
                                | DEALLOCATE(FLUSH)  |
                                |____________________|
                                          |

                                        (exit)
```

Figure 9-55. MDS_SEND Processing (Part 2 of 4)

1. If the MDS-MU in the MDS Routing Unit is null:

   a. If additional MDS Routing Units are enqueued to this TP:

      1) The MDS Routing Unit with the null MDS-MU is discarded.
      2) Processing continues at the beginning of this process.

   b. If additional MDS Routing Units are *not* enqueued to this TP:

      1) The MDS Routing Unit is updated with direction REPLY TO OUTBOUND.
      2) The MDS Routing Unit is sent to the MDS_ROUTER
      3) DEALLOCATE(FLUSH) is issued.
      4) This process ends.

2. If the MDS level of the session partner is BACK-LEVEL, processing continues at connector box 4 (step 1).

3. If the MDS-MU is not null,

   SEND_DATA DATA(<MDS-MU>) is issued.

   If the MDS level of the session partner is BACK-LEVEL, only the MS application program data GDS variable (CP-MSU) is sent.

4. CONFIRM is issued.

5. If the primary RC is *not* OK, processing continues at connector box 3.

6. If the primary RC is OK, the message was received and CONFIRMED by target TP in the session partner.

7. The MDS Routing Unit is updated:

   ° Direction REPLY TO OUTBOUND

   ° Session information

   ° Routing status COMPLETE

8. The MDS Routing Unit is returned to the MDS_ROUTER.

9. If additional MDS Routing Units are in the TP input queue, and last send was successful, processing continues at beginning of this process.

10. DEALLOCATE (FLUSH) is issued and this process exits.

```
                    ___
                   | 3 | (Primary RC was not OK)
                   |___|
                     |

             ____________________
            |       issue        |
```

```
          | DEALLOCATE(LOCAL)  |
          |____________________|
                    |
 ___________________|
|
|
|    ______________________________
|    |when RC=RESOURCE_FAILURE_RETRY|                                    ___
|___ÿ|     or ALLOCATION_ERROR.     |___________________________________ÿ| 1 |
|    |       TP_NOT_AVAILABLE_RETRY |                                    |___|
|    |______________________________|
|
|    ______________________________        ____________________
|    |  when RC=ALLOCATION_ERROR.   |       | Update MDS Routing |          ___
|___ÿ|           TPN_NOT_RECOGNIZED |______ÿ|  Unit w/MDS level  |______ÿ| 1 |
|    |    and MDS level UNKNOWN     |       |     BACK-LEVEL     |       |___|
|    |______________________________|       |____________________|
|
|    ______________________________        ____________________
|    |                              |       |    Retry limit     |          ___
|___ÿ| when RC=DEALLOCATE_ABEND_PROG|______ÿ|exceeded? (Dealloc_ |______ÿ| 1 |
|    |                              |       |   abend 2 times)   |  no   |___|
|    |______________________________|       |____________________|
|                                                   | yes
|
|                                            ____________________
|    ______________________________         |Update MDS Routing  |
|___ÿ|          otherwise           |______ÿ|Unit w/status ERROR,|
     |______________________________|   "   |sense X'08A8 000B', |
                                       |    |Dir. REPLY TO OUTB. |
                                       |    |____________________|
                                       |             |
                                       |
                                       |                      °°°°°°°°°°°°°°°°°°°°°°°
                                       |    ____________________    °                   °
                                       |    |  Send MDS Routing  |    °                   °
                                       |    | Unit to MDS_ROUTER |___ÿ°     MDS_ROUTER    °
                                       |    |____________________|    °                   °
                                       |             |                °°°°°°°°°°°°°°°°°°°°°°°
                                       |
                                       |    ____________________
                                       |    |  Additional MDS    |
                                       |___| Routing Units in    |
                                      yes |   input queue?      |
                                           |____________________|
                                                     | no

                                                  (exit)
```

Figure 9-56. MDS_SEND Processing (Part 3 of 4)

1. DEALLOCATE TYPE(LOCAL) is issued to reset the TP state.

2. When the CONFIRM RC = ALLOCATION_ERROR.TPN_NOT_RECOGNIZED or RESOURCE_FAILURE_RETRY, processing continues at connector box 1, "MDS_SEND ALLOCATE Processing"

3. When the CONFIRM RC = ALLOCATION_ERROR.TPN_NOT_RECOGNIZED and the MDS support level is UNKNOWN, allocation will be attempted for a BACK-LEVEL node.

   a. The MDS Routing Unit is updated with MDS level BACK-LEVEL

   b. Processing continues with "MDS_SEND ALLOCATE Processing."

4. When the CONFIRM RC = DEALLOCATE_ABEND_PROG, and the retry limit for this return code has not been exceeded, processing continues at connector box 1, "MDS_SEND ALLOCATE Processing." If the retry limit has been exceeded (MDS_SEND will attempt to reallocate one time after receiving a return code of DEALLOCATE_ABEND_PROG), processing continues at the next step. Refer to "MDS RECEIVE TP Failure" in topic 2.9.2.4.9 for a description of this retry situation.

5. For any other return code, the following steps are taken:

   a. The MDS Routing Unit is updated with these values:

      1) Direction REPLY TO OUTBOUND
      2) Status ERROR
      3) Sense data X'08A8 000B' (TP failure in remote node)
      4) MDS error message required ORIGIN

      Note: Sense data X'08A8 000B' applies only to a single MDS-MU - it is not the equivalent of session-outage notification. An irrecoverable TP failure in a remote node may be a transient error and subsequent communication may succeed. The MS application program receiving an MDS error message with that sense data will persue application program-level recovery, such as switching to a backup focal point, but other application program-level traffic should not be impacted by this error.

b. The MDS Routing Unit is sent to the MDS_ROUTER.

c. If additional MDS Routing Units are in the TP input queue, they are also updated with the same values and sent to the MDS_ROUTER.

```
                        ___
                       | 4 | (MDS level of partner is BACK-LEVEL)
                       |___|
                         |

|              ___________________          ____________________
|             | Destination NAU =  |        | Update MDS Routing |
|             |  session partner   |________ÿ|   Unit w/status    |______
|             |        NAU?        |  no     | INCOMPLETE, sense  |      |
|             |                    |         | data X'08A8 0007'  |      |
|             |____________________|         |____________________|      |
|                       | yes                                           |
|                       |                                               |
|                                                                      |
|              ___________________          ____________________        |
|             | Is Destination NAU |        | Update MDS Routing |      |
|             |    a served EN?    |________ÿ|Unit w/status ERROR,|_____ÿ|
|             |   (mode=CPSVCMG)   |  yes    | sense X'08A8 0005' |      |
|             |____________________|         |____________________|      |
|                       | no                                            |
|                       |                                               |
|                                                                      |
|              ___________________          ____________________        |
|             |Is appl GDS a CP-MSU|        | Update MDS Routing |      |
|             |containing Alert or |________ÿ|Unit w/status ERROR,|_____ÿ|
|             |acceptable MS-Caps? |  no     | sense X'08A8 0006' |      |
|             |____________________|         |____________________|      |
|                       | yes                                           |
|                       |                                               |

                        ___                                          ___
                       | 5 |                                        | 6 |
                       |___|                                        |___|
```

Figure 9-57. MDS_SEND Processing (Part 4 of 4)

1. MDS level of the session partner is BACK-LEVEL (connector box 4)

| 2. The NETID and NAU name of the Destination Location Name is compared to the session partner name. If the names are not the same, the Destination Location Name is assumed to be an end node and the session partner is assumed to be the network node server. Snce back-level network nodes do not support MDS intermediate routing, The MDS_ROUTER will attempt to reroute this MDS-MU over an SNASVCMG session directly to the end node destination.

| If the names are *not* the same:

| a. The MDS Routing Unit is updated:

| ° Routing Status is set to INCOMPLETE. This status, together with the sense data value, will indicate to the MDS_ROUTER that the MDS-MU is to be rerouted to the end node directly.
| ° Sense data is set to X'08A8 0007' (function not supported by network node server)
| ° Direction is set to REPLY TO OUTBOUND
| ° MDS error message required is set to ORIGIN

| b. Processing continues at connector box 6 (step 8).

| 3. If the destination NAU is a client end node (mode name is CPSVCMG), the MDS-MU cannot be sent, because back-level end nodes do not accept any CP-MSUs. An MDS error message will be returned to the origin.

| If the mode name is CPSVCMG:

| a. The MDS Routing Unit is updated:

| ° Routing Status is set to ERROR.
| ° Sense data is set to X'08A8 0005' (function not supported by end node destination)
| ° Direction is set to REPLY TO OUTBOUND
| ° MDS error message required is set to ORIGIN

| b. Processing continues at connector box 6 (step 8).

4. The MS application program GDS variable in the MDS-MU must be examined
| to determine if it is valid for the back-level destination. It must

```
|     be a CP-MSU (X'1212') GDS variable containing one of the following
|     (these are the only valid contents):


|     a.  Alert (X'0000') major vector


|     b.  MS Capabilities (X'80F0') major vector


|         1)  Focal Point Authorization Reply (X'62') subvector


|         2)  Focal Point Authorization Request (X'61') subvector


|             a)  Primary Authorization Request (X'10') Focal Point
|                 Authorization Request subfield is present and contains the
|                 back-level value of X'0000'.


|             b)  Default Authorization Request (X'20') Focal Point
|                 Authorization Request subfield is present and contains the
|                 back-level value of X'0000'.


|     If the MS application program GDS variable is not one of the above, it
|     is not acceptable for a back-level node:


      a.  The MDS Routing Unit is updated:


          °   Routing Status is set to ERROR.
          °   Sense data is set to X'08A8 0006' (function not supported by
              network node destination)
          °   Direction is set to REPLY TO OUTBOUND
          °   MDS error message required is set to ORIGIN


      b.  Processing continues at connector box 6 (step 8).


  5.  If all the back-level compatability tests are met, processing
      continues at connector box 5 (step 3).
```

---

(1) The LU 6.2 verb examples shown illustrate the key options required for this design. They may not show all required parameters.

2.9.2.4.11 MDS_RECEIVE Processing

Process Name	MDS_RECEIVE TP
Description	This transaction program receives MDS-MUs sent by remote instances of the MDS_SEND TP. The MDS-MUs are passed to the MDS_ROUTER.
Input	MDS-MU received from remote node.
Output	° MDS Routing Unit passed to MDS_ROUTER.

The MDS_RECEIVE transaction program is started by an Attach coming into the node from a remote instance of the MDS_SEND transaction program. It is illustrated in Figure 9-58.

```
                        (TP started by incoming Attach)
                                     |

                          _________________
                         |     issue       |
                         | GET_ATTRIBUTES  |
                         |_________________|
                                     |

                          _________________  _____________________
                         | Mode name CPSVCMG |______ÿ| issue DEALLOCATE |___ÿ(exit)
                         |   or SNASVCMG?    | no    |   (ABEND_PROG)   |
                         |_________________|       |_____________________|
                                     | yes
                                     |          ___
                   ________________________ÿ|____________| 1 |________
                  |                                     |___|         |
                  |                          _________________          |
                  |                         |     issue       |          |
                  |                         | RECEIVE_AND_WAIT |          |
                  |                         |    FILL(LL)     |          |
                  |                         |_________________|          |
```

```
           |                          |                          |
           |                                                     |
           |                 ___________________        ___      |
           |                | Is return code OK? |____ÿ| 2 |     |
           |                |____________________| no  |___|     |
           |                          |yes                       |
           |                                                     |
 ___________________       ___________________                   |
| issue CONFIRMED    |_____|                   |_________________|
|____________________|CONFIRM|   WHAT_RECEIVED?   | DATA_INCOMPLETE
                            |                   |
                            |___________________|
 ___________________                 |  |  | other        ___________________
| issue CONFIRMED    |_____________|  |  |_______________ÿ| issue DEALLOCATE  |___ÿ(exit)
|____________________|CONFIRM_DEALLOC. |                 |    (ABEND_PROG)    |
           |                          |DATA_COMPLETE     |___________________|

        (exit)                      ___________________
                            |     Build MDS      |
                            |    Routing Unit    |
                            |                    |
                            |____________________|
                                       |

                            ___________________          °°°°°°°°°°°°°°°°°°°°°°
                            | send to MDS_ROUTER |______ÿ°                    °
                            |                    |       °     MDS_ROUTER     °
                            |____________________|       °                    °
                                       |                 °°°°°°°°°°°°°°°°°°°°°°

 ___________________       ___________________
| issue DEALLOCATE   |_____|    Did send to     |
| (ABEND_PROG)       |  no |  MDS_ROUTER work?  |
|____________________|     |____________________|
           |                          | yes

        (exit)                        ___
                                     | 1 |
                                     |___|
```

Figure 9-58. MDS_RECEIVE Processing (Part 1 of 2)

It performs the following processing:

1. GET_ATTRIBUTES is issued to get

   - Network-qualified NAU name for the session partner
   - Mode name
   - CGID

   This information is required in the MDS Routing Unit that will be built.

2. If mode name is not SNASVCMG or CPSVCMG, DEALLOCATE TYPE(ABEND_PROG) is issued and the TP ends.

3. RECEIVE_AND_WAIT FILL(LL) is issued.

4. The return code is examined:

   If return code is not OK, processing continues at connector box 2, labeled "Primary RC from RECEIVE_AND_WAIT was not OK."

   If the return is OK:

   a. If WHAT_RECEIVED is DATA_INCOMPLETE, processing continues at step 3.

   b. If WHAT_RECEIVED is DATA_COMPLETE:

      1) The MDS Routing Unit is prepared:

         - Direction INBOUND
         - Network-qualified NAU name
         - Mode Name
         - Session CGID
         - MDS Support Level CURRENT
         - Data received (MDS-MU)
         - Routing status INCOMPLETE

      2) The completed MDS Routing Unit is sent to the MDS_ROUTER.

      3) If that interprocess send fails for any reason, the TP issues:

         DEALLOCATE(ABEND_PROG)

```
            and terminates.

        4)  Processing continues at step 3.

    c.  If WHAT_RECEIVED is CONFIRM:

        1)  CONFIRMED is issued.

        2)  Processing continues at step 3.

    d.  If WHAT_RECEIVED is CONFIRM_DEALLOCATE:

        1)  CONFIRMED is issued.

        2)  This process terminates.

    e.  If WHAT_RECEIVED is any other value, the TP issues
        DEALLOCATE(ABEND_PROG) and terminates.


                      ___
                     | 2 | (Primary RC from RECEIVE_AND_WAIT was not OK)
                     |___|
                       |
  ___________________|
 |
 |
 |      ___________________________________
 |____ÿ| when ALLOCATION_ERROR            |
 |     |   or DEALLOCATE_NORMAL           |
 |     |   or DEALLOCATE_ABEND_PROG       |        ___________________
 |     |   or DEALLOCATE_ABEND_SVC        |______ÿ|  issue DEALLOCATE  |___ÿ(exit)
 |     |   or DEALLOCATE_ABEND_TIMER      |       |      (LOCAL)       |
 |     |   or RESOURCE_FAILURE_RETRY      |       |____________________|
 |     |   or RESOURCE_FAILURE_NO_RETRY   |
 |     |__________________________________|
 |
 |
 |      ___________________________________        ___________________
 |____ÿ|    otherwise (any other RC)      |______ÿ|  issue DEALLOCATE  |___ÿ(exit)
       |                                  |       |   (ABEND_PROG)     |
       |__________________________________|       |____________________|
```

Figure 9-59. MDS_RECEIVE Processing (Part 2 of 2)

1. If the return code is one of the following,

   ```
   ALLOCATION_ERROR
   DEALLOCATE_NORMAL
   DEALLOCATE_ABEND_PROG
   DEALLOCATE_ABEND_SVC
   DEALLOCATE_ABEND_TIMER
   RESOURCE_FAILURE_RETRY
   RESOURCE_FAILURE_NO_RETRY
   ```

   The conversation remains in DEALLOCATE state, and DEALLOCATE TYPE(LOCAL) must be issued to reset it before ending the TP.

2. If the return code is any other value, DEALLOCATE TYPE(ABEND_PROG) is issued to unconditionally end the conversation.

| 2.9.2.4.12 MSU_HANDLER Processing

	Process Name	MSU_HANDLER TP
	Description	This transaction program receives CP-MSUs sent by back-level nodes. The CP-MSUs are passed to the MDS_ROUTER.
	Input	CP-MSU received from remote node.
	Output	° MDS Routing Unit passed to MDS_ROUTER.

| The MSU_HANDLER transaction program is started by an Attach coming into
| the node from a remote instance of the MSU_HANDLER transaction program in
| a back-level node. It is illustrated in Figure 9-60.

```
|                          (TP started by incoming Attach)
```

```
|                                   |
|
|                            _________________
|                           |      Issue      |
|                           |  GET_ATTRIBUTES |
|                           |_________________|
|                                   |
|
|                            _________________ _________________
|                           | Mode name CPSVCMG |______ÿ| issue DEALLOCATE |___ÿ(exit)
|                           |   or SNASVCMG?    | no    |   (ABEND_PROG)   |
|                           |_________________|       |_________________|
|                                   | yes
|                                   |
|            _________________________ÿ|_____________| 1 |________
|           |                                        |___|         |
|           |                            _________________          |
|           |                           |      Issue      |         |
|           |                           | RECEIVE_AND_WAIT |         |
|           |                           |    FILL(LL)     |         |
|           |                           |_________________|         |
|           |                                   |                   |
|           |                                                       |
|           |                            _________________ ___      |
|           |                           | Is return code OK? |____ÿ| 2 |      |
|           |                           |_________________| no  |___|      |
|           |                                   |yes                  |
|           |                                                       |
| _________________ _________________                               |
| | Issue CONFIRMED |______| |_________________|
| |_________________|CONFIRM|   WHAT_RECEIVED?   | DATA_INCOMPLETE
|                           |                 |
|                           |_________________|
| _________________                 |  |  | other          _________________
| | Issue CONFIRMED |_____________|  |  |_____________ÿ| Issue DEALLOCATE |___ÿ(exit)
| |_________________|CONFIRM_DEALLOC. |                 |   (ABEND_PROG)   |
|           |                       |DATA_COMPLETE    |_________________|
|
|         (exit)
|                            _________________
|                           |    Build MDS    |
|                           |   Routing Unit  |
|                           |                 |
|                           |_________________|
|                                   |
|
|                            _________________        °°°°°°°°°°°°°°°°°°°°°°
|                           | Send to MDS_ROUTER |______ÿ°                  °
|                           |                    |       °    MDS_ROUTER    °
|                           |____________________|       °                  °
|                                   |                    °°°°°°°°°°°°°°°°°°°°°°
|
| _________________ _________________
| | Issue DEALLOCATE |______| Did send to     |
| | (ABEND_PROG)     |  no  | MDS_ROUTER work? |
| |_________________|       |_________________|
|           |                       | yes
|
|         (exit)
|                                  ___
|                                 | 1 |
|                                 |___|
```

Figure 9-60. MSU_HANDLER Processing (Part 1 of 2)

It performs the following processing:

1. GET_ATTRIBUTES is issued to get

   - Network-qualified LU name for the session partner
   - Mode name
   - CGID

   This information is required in the MDS Routing Unit that will be built.

2. If mode name is not SNASVCMG or CPSVCMG, DEALLOCATE TYPE(ABEND_PROG) is issued and the TP ends.

3. RECEIVE_AND_WAIT FILL(LL) is issued.

4. The return code is examined:

   If return code is not OK, processing continues at connector box 2, labeled "Primary RC from RECEIVE_AND_WAIT was not OK."

   If the return is OK:

   a. If WHAT_RECEIVED is DATA_INCOMPLETE, processing continues at step 3.

```
|     b.  If WHAT_RECEIVED is DATA_COMPLETE:


|         1)  The MDS Routing Unit is prepared:


|             °   Direction INBOUND
|             °   Network-qualified LU name
|             °   Mode Name
|             °   Session CGID
|             °   MDS Support Level BACK-LEVEL
|             °   Data received (CP-MSU)
|             °   Routing status INCOMPLETE


|         2)  The completed MDS Routing Unit is sent to the MDS_ROUTER.


|         3)  If that interprocess send fails for any reason, the TP issues:


|               DEALLOCATE(ABEND_PROG)


|             and terminates.


|         4)  Processing continues at step 3.


|     c.  If WHAT_RECEIVED is CONFIRM:


|         1)  CONFIRMED is issued.


|         2)  Processing continues at step 3.


|     d.  If WHAT_RECEIVED is CONFIRM_DEALLOCATE:


|         1)  CONFIRMED is issued.


|         2)  This process terminates.


|     e.  If WHAT_RECEIVED is any other value:  the TP issues
|         DEALLOCATE(ABEND_PROG) and terminates.


|                      ___
|                    | 2 | (Primary RC from RECEIVE_AND_WAIT was not OK)
|                    |___|
|                      |
|    __________________|
|   |
|   |      ________________________________
|   |____ÿ| When ALLOCATION_ERROR          |
|   |     |   or DEALLOCATE_NORMAL         |
|   |     |   or DEALLOCATE_ABEND_PROG     |        ____________________
|   |     |   or DEALLOCATE_ABEND_SVC      |______ÿ|  Issue DEALLOCATE  |___ÿ(exit)
|   |     |   or DEALLOCATE_ABEND_TIMER    |       |      (LOCAL)       |
|   |     |   or RESOURCE_FAILURE_RETRY    |       |____________________|
|   |     |   or RESOURCE_FAILURE_NO_RETRY |
|   |     |________________________________|
|   |
|   |
|   |      ________________________________        ____________________
|   |____ÿ|    Otherwise (any other RC)     |______ÿ|  Issue DEALLOCATE  |___ÿ(exit)
|         |                                 |       |   (ABEND_PROG)     |
|         |_________________________________|       |____________________|
```

| Figure 9-61. MSU_HANDLER Processing (Part 2 of 2)

```
| 1.  If the return code is one of the following:


|       ALLOCATION_ERROR
|       DEALLOCATE_NORMAL
|       DEALLOCATE_ABEND_PROG
|       DEALLOCATE_ABEND_SVC
|       DEALLOCATE_ABEND_TIMER
|       RESOURCE_FAILURE_RETRY
|       RESOURCE_FAILURE_NO_RETRY


|     the conversation remains in DEALLOCATE state, and DEALLOCATE
|     TYPE(LOCAL) must be issued to reset it before ending the TP.


| 2.  If the return code is any other value, DEALLOCATE TYPE(ABEND_PROG) is
|     issued to unconditionally end the conversation.
```

2.9.2.4.13 MDS_SON Processing

Process Name	MDS_SON TP
Description	This transaction program receives session outage notification from the LU for SNASVCMG sessions.
Input	SNASVCMG session outage information from LU
Output	"MDS Internal Signal - SNASVCMG SON (Session-Outage Notification)" in topic 2.9.2.4.4

The MDS_SON transaction program is defined to the LU (via the DEFINE_MODE verb) as the session-outage TP for all SNASVCMG sessions. This is a single-instance transaction program.

Implementation Note

In the architecture model, this control operator verb is issued to define that MDS_SON is to receive session-outage notification for the SNASVCMG mode name. The same technique is used to define that a session services TP is to receive session-outage notification for the CPSVCMG mode name (see *SNA APPN Architecture Reference*).

What is required is that MULTIPLE_DOMAIN_SUPPORT be notified of SNASVCMG session outages. Implementations may choose to provide this function in a manner different from the TP described here.

```
                    (PIP from LU)
                          |

                 ___________________
                | build SNASVCMG SON |
                |       signal       |
                |                    |
                |____________________|
                          |

                 ____________________        °°°°°°°°°°°°°°°°°°°°°°°
                |                    |       °                     °
                | send to MDS_ROUTER |______ÿ°     MDS_ROUTER      °
                |                    |       °                     °
                |____________________|       °°°°°°°°°°°°°°°°°°°°°°°
                          |

                        (exit)
```

Figure 9-62. Processing for MDS_SON

1. Whenever a session outage occurs, the LU will start the TP and pass it PIP data containing information identifying the session lost as well as the precise sense data describing the reason.

2. The TP prepares the SNASVCMG SON signal ("MDS Internal Signal - SNASVCMG SON (Session-Outage Notification)" in topic 2.9.2.4.4) and passes it to the MDS_ROUTER

3. The TP terminates.

| 2.9.2.5 MULTIPLE_DOMAIN_SUPPORT Optional Subset 1 (End Node Support)

Subtopics:

- 2.9.2.5.1 Functions Provided

| 2.9.2.5.1 Functions Provided

| This optional subset consists of the MDS_ROUTER functions described in the
| base subset, except for those specifically labeled as "network node only."
| The portions of the base MDS_ROUTER that have "network node only"
| functions are:

| ° "MDS Routing Algorithm" in topic 2.9.2.4.8.

| ° "Processing of CPSVCMG Session List" in topic 2.9.2.4.8.

2.9.2.6 MULTIPLE_DOMAIN_SUPPORT Optional Subset 2 (Network Node Support)

This optional subset consists of the MDS_ROUTER functions for a network node. It is required for all network node implementations. In addition to this optional subset, those portions of the MDS_ROUTER base subset designated "network node only" must be implemented. Those portions of the MDS_ROUTER base designated "end node only" are not implemented by a network node.

Subtopics:

- 2.9.2.6.1 Function Provided
- 2.9.2.6.2 Implementation Requirements

2.9.2.6.1 Function Provided

Optional Subset 2 (Network Node Support) is required for an implementation of an APPN network node. This subset provides the MDS_ROUTER function for interrogating CP directory services to determine the appropriate routing for MDS-MU destinations.

2.9.2.6.2 Implementation Requirements

The requirements for implementing MULTIPLE_DOMAIN_SUPPORT Optional Subset 3 (Network Node Support) are described by a model consisting of a subset of CP directory services, CP session services, and CP management services.

A Subset of CP Session Services:

The following signals are exchanged with CP session services over external protocol boundary 11, as illustrated in Figure 9-4 in topic 2.9.2.4.4.

ASSIGN_PCID Signal

Origin	MDS_ROUTER
Destination	CP Session Services
Data Content	1. Requesting Process type ° Session Manager ° Session Connector Manager ° Management Services 2. Requesting Process Identifier 3. Duplicate PCID ° Yes ° No (Always No for MS requests)
Initiates	Processing internal to CP Session Services. Session Services replies with the ASSIGN_PCID_RSP.

Note: This signal is required only as part of "MULTIPLE_DOMAIN_SUPPORT Optional Subset 2 (Network Node Support)" in topic 2.9.2.6

ASSIGN_PCID_RSP Signal

Origin	CP Session Services
Destination	MDS_ROUTER
Data Content	1. Fully Qualified PCID (X'60') Control Vector
Initiates	Processing described in "Directory Services Processing." This signal is the reply to ASSIGN_PCID.

Note: This signal is required only as part of "MULTIPLE_DOMAIN_SUPPORT Optional Subset 2 (Network Node Support)" in topic 2.9.2.6

A Subset of CP Directory Services:

The following signal is exchanged with CP directory services over external protocol boundary 20, as illustrated in Figure 9-4 in topic 2.9.2.4.4.

LOCATE_MESSAGE Signal

Description	Passed as a request from MDS_ROUTER to CP Directory Services. Directory Services replies with the same signal.
Data Content	1. Fully-Qualified PCID (X'60') control vector 2. Origin NAU name (on a request) 3. Origin CP name (on a request) 4. Destination NAU name (on a reply) 5. Destination CP name (on a reply) 6. Destination network node Server's CP name (on a reply) 7. Locate status ° Request ° Reply 8. Locate chain ° Keep ° Discard 9. Search executed ° Local-only ° Distributed 10. Sense data (on a reply) ° All 0's if successful ° Nonzero, otherwise
Initiates	° Request direction: Processing internal to CP Session Services. ° Reply direction: Processing described in "Directory Services Processing."

Note: This signal is required only as part of "MULTIPLE_DOMAIN_SUPPORT Optional Subset 2 (Network Node Support)" in topic 2.9.2.6

A Subset of CP Management Services:

Provides "Directory Services Processing."

Directory Services Processing

Process Name	Directory Services Processing
Description	This process builds LOCATE_MESSAGE requests to CP directory services, interprets the replies, and updates an MDS Routing Unit as appropriate with session partner information.
Input	"MDS Internal Signal - MDS Routing Unit" in topic 2.9.2.4.4
Output	The output from this process is one of the following: ° Active transaction list is updated. ° MDS Routing Unit is updated with routing information from directory services ° If directory services search fails, MDS Routing Unit will be updated with sense data so that an MDS error message can be created.

This process is illustrated in Figure 9-63.

```
                        (MDS Routing Unit(Outbound))
                                    |
                                    |
                                                   °°°°°°°°°°°°°°°°°°°°°°
                         ____________________      °         CP         °
                        |Prepare ASSIGN_PCID |     °  session services  °
                        |   and send to CP   |_____ÿ°                   °
                        |  session services  | PB 11 °                  °
                        |____________________|       °°°°°°°°°°°°°°°°°°°°°°
                                    |                          |
                                                             |
```

```
|                                 _____________________                    |
|                                |      Receive        |                   |
|                                |ASSIGN_PCID_RSP from |___________________|
|                                |CP session services  |      PB 11
|                                |_____________________|
|                                          |
|
|                                 _____________________
|                                |      Prepare        |       °°°°°°°°°°°°°°°°°°°°°°°
|                                |   LOCATE_MESSAGE    |       °         CP         °
|                                |for destination NAU  |______ÿ° directory services °
|                                |     in MDS-MU       | PB 20 °                    °
|                                |_____________________|       °°°°°°°°°°°°°°°°°°°°°°°
|                                          |                                |
|                                                                          |
|                                 _____________________                    |
|                                |      Receive        |                   |
|                                |   LOCATE_MESSAGE    |                   |
|                                |   reply from CP     |___________________|
|                                | directory services  |      PB 20
|                                |_____________________|
|                                          |
|                                                          ___________________
|                                 _____________________   | Update MDS Routing |
|                                |  resource found?    |______ÿ| Unit w/sense data  |
|                                |_____________________|  no   |    X'08A8 0001',   |
|                                          | yes               |Routing Status ERROR|
|                                          |                   |____________________|
|                                                                        |
|                                        ___                              |
|                                       | 1 |
|                                       |___|                       (exit)
```

Figure 9-63. Directory Services Support (Part 1 of 2)

The process is described below:

1. "ASSIGN_PCID_Signal" is prepared and sent to CP session services over external protocol boundary 11.

2. "ASSIGN_PCID_RSP_Signal" is received from CP session services.

3. A LOCATE_MESSAGE is built containing the following information:

   - FQPCID

     This is the value obtained from session services.

   - Origin NAU name:

     Network-qualified CP name for this network node.

   - CP name for node containing the origin NAU:

     Network-qualified CP name for this network node.

   - Destination NAU name:

     Network-qualified name of the LU (or CP) that directory services is to locate. This is the destination NAU in the MDS-MU.

   - CP name for node containing the destination NAU:

     This value is provided by directory services if the resource is located.

   - CP name for node providing network services for destination NAU:

     This value is provided by directory services if the resource is located.

   - Search executed

     - If the routing status of the MDS Routing Unit (OUTBOUND) is not ERROR, then a *local-only* search is requested.

       Directory services is instructed to satisfy the Locate request using local information only, without sending a directed Locate out of the node to verify the locally-cached information. The local information searched includes the local copy of the network topology, locally defined and registered directory entries, and cached directory entries.

```
|            If directory services does not find the resource in the
|            locally-available databases, MDS will request that a
|            distributed network search (via a central directory server,
|            when present) for the resource be performed.  This network
|            search is requested when MDS specifies the value distributed
|            in the search executed field of LOCATE_MESSAGE.


|        -   If the routing status of the MDS Routing Unit (OUTBOUND) is
|            ERROR, then a previous attempt to use local directory
|            information to route the MDS-MU was unsuccessful.  A
|            distributed search is requested in that case.


|        MDS attempts to use the local directory information in this manner
|        as an optimization.  The presumption is that the local information
|        will be accurate in most cases, allowing the MDS-MU to be routed
|        without invoking the overhead and delay of a distributed directory
|        search.  If the resource is not found in the local directory
|        cache, then a distributed search is unavoidable, and the overhead
|        and delay are accepted.  If the locally-cached information turns
|        out to be inaccurate (perhaps because the resource has left the
|        network or has moved to a different location), the MDS-MU will be
|        returned by the remote network node that was asked to forward it.
|        In that case, MDS in this network node will attempt to re-route
|        the MDS-MU by creating a Locate request with the value distributed
|        in the search executed field.  This will inform directory services
|        to perform its normal distributed search to locate the resource.
|        A side effect of that search is that the local cache will be
|        updated, as well.


| 4.  This message is passed to directory services on protocol boundary 20.


| 5.  A LOCATE_MESSAGE reply is received from directory services over
|     protocol boundary 20.


| 6.  If the resource was not found (indicated by nonzero sense data in the
|     LOCATE_MESSAGE reply):


|     a.  The MDS Routing Unit is updated:


|         °   Routing status ERROR
|         °   Sense data X'08A8 0001' (destination NAU unknown)
|         °   MDS error message required ORIGIN
|         °   Alert required NO


|     b.  This process exits.


|
|                                   ___
|                                  | 1 |
|                                  |___|
|                                    |
|
|                            ___________________
|                           |     Determine     |
|                           |destination NAU type|
|                           | from Locate reply |
|                           |___________________|
|                                    |
|
|                            ___________________
|                           |Destination NAU name|
|            ________________|same as destination |
|           |       YES     |  NN server name?  |
|           |(destination=NN)|___________________|
|           |                         | NO
|           |
|           |                ___________________
|           |               |Destination NAU name|_________________
|           |               |  same as CP name?  |       NO        |
|           |               |___________________|  (destination=LU)|
|           |                 YES     |                           |
|           |           (destination=EN)|                          |
|           |                         |                           |
|
| ___________________       ___________________       ___________________
| | Update MDS Routing |    ||  Processing of   ||    | Update MDS Routing |
| | Unit w/NNCP name,  |    ||     SNASVCMG    ||    |  Unit w/LU name,   |_____
| | mode name SNASVCMG,|    ||   session list   ||    | mode name SNASVCMG,|     |
| |routing verification|    ||___________________||    |routing verification|     |
| |     indicator      |              |                |     indicator      |     |
| |____________________|              |                |____________________|     |
|           |                                                                   |
|           |                ___________________       ___________________      |
|           |               |    Is MDS Level    |    | Update MDS Routing |     |
|           |               |    BACK-LEVEL?     |_____ÿ| Unit w/ENCP name,  |____ÿ|
|           |               |                    | yes | mode name SNASVCMG,|     |
|           |               |___________________|    |routing verification|     |
|           |                         |                |     indicator      |     |
|           |                         |                |____________________|     |
|           |                                                                   |
|           |                ___________________                                |
|           |               | Update MDS Routing |                              |
```

```
|          |                  | Unit w/NNCP name,  |                                |
|          |                  |mode name SNASVCMG, |                                |
|          |                  |routing verification|                                |
|          |                  |     indicator      |                                |
|          |                  |____________________|                                |
|          |                           |                                            |
|          |___________________________ÿ|____________________________________________|
|
|                                   (exit)
```

Figure 9-64. Directory Services Support (Part 2 of 2)

7. There are three cases to be considered for the directory reply:

a. Destination is a network node (inferred from Locate reply when the destination NAU name is the same as the destination network node server's CP name)

The route selected is an SNASVCMG session to the named NNCP. The MDS Routing Unit (OUTBOUND) is updated:

- Routing status is set to COMPLETE.
- Session partner network-qualified LU name is set to NNCP name.
- Session mode name is set to SNASVCMG

The Routing Verification Indicator in the Flags (X'90') subvector is set to 1, indicating that the directory information was verified. Even though directory services did not perform a network search, this setting is appropriate because information about network nodes comes from the topology data base, which is reliable.

b. Destination is an end node (inferred from Locate reply when the destination NAU name is the same as the destination CP name)

The Current Session List must be checked for MDS level information on the NNCP name. That check will produce the following cases:

1) MDS level is CURRENT

The MDS-MU is sent on the SNASVCMG session to the named NNCP.

The Routing Verification Indicator in the Flags (X'90') subvector is set based on the value returned in the LOCATE_MESSAGE reply:

- 1 if the directory information was verified (distributed search was performed)
- 0 if the directory information was not verified (local only search was performed)

The MDS Routing Unit (OUTBOUND) is updated:

- Routing Status is set to COMPLETE.
- Session partner network-qualified LU name is set to NNCP name.
- Session mode name is set to SNASVCMG

2) MDS level is BACK-LEVEL

Back-level network nodes do not support forwarding of MS messages to client end nodes, so the MDS-MU will be sent directly to the EN destination over an SNASVCMG session, bypassing the NNCP.

a) The MDS Routing Unit is updated:

- Routing Status is set to COMPLETE.
- Session partner network-qualified LU name is set to ENCP name.
- Session mode name is set to SNASVCMG

3) NNCP name not found (i.e., there is no current SNASVCMG session to it).

The MDS-MU is sent on the SNASVCMG session to the NNCP. The MDS Routing Unit (OUTBOUND) is updated:

- Routing Status is set to COMPLETE.
- Session partner network-qualified LU name is set to NNCP name.

```
|            °   Session mode name is set to SNASVCMG


|     c.  Destination is an LU.  (inferred from Locate reply when the
|         destination NAU name is not the same as either the destination CP
|         name nor the destination network node server's CP name)


|         The route selected is an SNASVCMG session to the LU (this is
|         necessary to support configurations that include a subarea
|         network).


|     d.  The Routing Verification Indicator in the Flags SV is set based
|         upon the LOCATE_MESSAGE reply from directory services:


|         °   1 if the directory information was verified (distributed
|             search was performed)
|         °   0 if the directory information was not verified (local only
|             search was performed)
```

| *Sequences Illustrating Routing Based on Directory Services Information*:

| The following figures illustrate key sequences performed by the MDS_ROUTER
| in a network node. The figures all show the MDS-MU initiated by a local
| application program on the network node, but in each case, the same
| processing would be done if the MDS-MU were received over a CPSVCMG
| session from a client end node.

| **Routing Based on Information Known to MDS_ROUTER**

```
|   _____________ Network Node NN A  _________________   ________________ Network node NN_B _________________
|  |              ________________ CP.MS ________________ | |________________ CP.MS _______________            |
|  |             |MS      MDS_     MDS_Re-    MDS_   || ||MDS_      MDS_Re-    MDS_      MS  |             |
|  |CP.SS CP.DS |Appl    Router   ceive TP   Send TP|| ||Send TP   ceive TP   Router    Appl| CP.DS CP.SS|
|  |             |_____________________________________|| ||_____________________________________|             |
|  |_____________________________________________________| |_____________________________________________________|
|    .      .     .         .         .         .          .         .         .         .     .      .
|    .      .     . MDS-MU  .         .         .          .         .         .         .     .      .
| 1  .      .     o________ÿo         .         .          .         .         .         .     .      .
|    .      .     .         |    MDS-MU         .          .         .         .         .     .      .
| 2  .      .     .         |___________________ÿo         .         .         .         .     .      .
|    .      .     .         .         .         |     MDS-MU          .         .         .     .      .
| 3  .      .     .         .         .         |_____________________ÿo         .         .     .      .
|    .      .     .         .         .         .          .         | MDS-MU  .         .     .      .
| 4  .      .     .         .         .         .          .         |________ÿo         .     .      .
|    .      .     .         .         .         .     Confirmed       |         .         .     .      .
| 5  .      .     .         .         .        o_____________________|         .         .     .      .
|    .      .     .         .        OK         |          .         .         .         .     .      .
| 6  .      .     .         o___________________|          .         .         .         .     .      .
|    .      .     .         .         .         .          .         .         .         .     .      .
```

| Figure 9-65. MDS Router Chooses Outbound Session Based on Local MDS
| Information

| Figure 9-65 illustrates a simple case: an MDS-MU that is routed based on
| information kept locally by the MDS_ROUTER.

| The items in the following list match the line numbers in the figure.

| 1. NN_A: An MS application program in the network node passes an MDS-MU
| to the MDS_ROUTER for forwarding. The MDS-MU could also have been
| received on a CPSVCMG session from a client end node.

| The MDS_ROUTER checks its local routing information for a match on the
| destination in the MDS Routing Information (X'1311') GDS variable.
| The local information includes:

| °   Active transaction control blocks

| °   Direct routing requested by a local application program

| °   Active SNASVCMG session list

| °   End node status list

| A match is found, and the appropriate outbound session is identified.

| The Routing Verification Indicator in the Flags (X'90') subvector of
| the MDS Routing Information (X'1311') GDS variable is set to 1,

indicating that the routing was based upon reliable information.

2. NN_A: The MDS-MU, along with the outbound session information, is passed to the MDS_SEND TP.

3. NN_A: The MDS_SEND TP allocates a conversation with MDS_RECEIVE in NN_B, sends the MDS-MU, and requests confirmation.

4. NN_B: The MDS_RECEIVE TP receives the MDS-MU and passes it to the MDS_ROUTER.

5. NN_B: MDS_RECEIVE issues the verb CONFIRMED, to signal MDS_SEND in NN_A that the MDS-MU was received.

6. NN_A: MDS_SEND signals the MDS_ROUTER that the MDS-MU was sent successfully.

Routing Based on Locally-Cached Directory Information

```
     _______________ Network Node NN_A _________________   _______________ Network Node NN_B _________________
    |               _______________ CP.MS _______________ | |_______________ CP.MS _______________|
    |              |MS      MDS_     MDS_Re-    MDS_   || ||MDS_      MDS_Re-    MDS_      MS  |            |
    |CP.DS CP.SS |Appl    Router   ceive TP   Send TP|| ||Send TP   ceive TP   Router    Appl| CP.DS CP.SS|
    |              |_____________________________________|| ||_____________________________________|            |
    |_____________________________________________________| |____________________________________________________|
      .      .     .         .         .         .          .         .         .         .     .      .
      .      .     . MDS-MU  .         .         .          .         .         .         .     .      .
 1    .      .     o________ÿo         .         .          .         .         .         .     .      .
      .         ASSIGN_PCID  |         .         .          .         .         .         .     .      .
 2    .      o______________|         .         .          .         .         .         .     .      .
      .      |  ASSIGN_PCID_RSP        .         .          .         .         .         .     .      .
 3    .      |______________ÿo         .         .          .         .         .         .     .      .
      LOCATE_MESSAGE (local) |         .         .          .         .         .         .     .      .
 4    o______________________|         .         .          .         .         .         .     .      .
      | LOCATE_MESSAGE (local) reply   .         .          .         .         .         .     .      .
 5    |______________________ÿo         .         .          .         .         .         .     .      .
      .      .     .         |    MDS-MU         .          .         .         .         .     .      .
 6    .      .     .         |___________________ÿo          .         .         .         .     .      .
      .      .     .         .         .         |     MDS-MU          .         .         .     .      .
 7    .      .     .         .         .         |____________________ÿo         .         .     .      .
      .      .     .         .         .         .          .         | MDS-MU  .         .     .      .
 8    .      .     .         .         .         .          .         |________ÿo         .     .      .
      .      .     .         .         .         .     Confirmed       |         .         .     .      .
 9    .      .     .         .         .         o____________________|         .         .     .      .
      .      .     .         .        OK         |          .         .         .         .     .      .
 10   .      .     .         o___________________|          .         .         .         .     .      .
      .      .     .         .         .         .          .         .         .         .     .      .
```

Figure 9-66. MDS Router Chooses Outbound Session Based on Directory Services Local Information

Figure 9-66 illustrates an MDS-MU that is routed based on information cached locally by control point directory services (no network search invoked and no verification preformed).

The items in the following list match the line numbers in the figure.

1. NN_A: An MS application program in the network node passes an MDS-MU to the MDS_ROUTER for forwarding. The MDS-MU could also have been received on a CPSVCMG session from a client end node.

2. NN_A: The MDS_ROUTER checks its local routing information for a match on the destination in the MDS Routing Information (X'1311') GDS variable.

   No match is found.

   A request for the assignment of a PCID (ASSIGN_PCID signal) is sent to control point session services. The PCID is required for directory search requests.

3. NN_A: CP.SS returns the PCID to the MDS_ROUTER via the ASSIGN_PCID_RSP signal.

4. NN_A: MDS_ROUTER sends a request for a search of the local directory cache to CP.DS. This request is the LOCATE_MESSAGE signal with the search executed indicator set to local-only.

5. NN_A: CP.DS locates the resource in the local directory cache and

```
|     passes the information back to the MDS_ROUTER as a LOCATE_MESSAGE
|     reply.


|     The MDS_ROUTER then decides what outbound session is to be used for
|     routing the MDS-MU: either an LU-LU session to the network node server
|     of the destination NAU, if the destination is an ENCP, or an LU-LU
|     session to the destination NAU itself, if the destination is an LU or
|     a NNCP.


|     The Routing Verification Indicator in the Flags (X'90') subvector of
|     the MDS Routing Information (X'1311') GDS variable is set to 0,
|     indicating that the routing was based upon unverified information.


| 6.  NN_A:  The MDS-MU, along with the outbound session information, is
|     passed to the MDS_SEND TP.


| 7.  NN_A:  The MDS_SEND TP allocates a conversation with MDS_RECEIVE in
|     NN_B, sends the MDS-MU, and requests confirmation.


| 8.  NN_B:  The MDS_RECEIVE TP receives the MDS-MU and passes it to the
|     MDS_ROUTER.


| 9.  NN_B:  MDS_RECEIVE issues the verb CONFIRMED, to signal MDS_SEND in
|     NN_A that the MDS-MU was received.


| 10. NN_A:  MDS_SEND signals the MDS_ROUTER that the MDS-MU was sent
|     successfully.
```

| **Routing Based on Non-Local Directory Information**

```
|    _______________ Network Node NN_A _________________   _________________ Network Node NN_B _________________
|   |                _________________ CP.MS _________________ | |_________________ CP.MS _________________|
|   |              |MS      MDS_     MDS_Re-    MDS_   || ||MDS_      MDS_Re-    MDS_      MS  |             |
|   |CP.DS CP.SS |Appl    Router   ceive TP   Send TP|| ||Send TP   ceive TP   Router    Appl| CP.DS CP.SS|
|   |              |_____________________________________|| ||_____________________________________|             |
|   |_____________________________________________________| |_____________________________________________________|
|     .      .     .          .          .          .          .          .          .          .     .      .
|     .      .     . MDS-MU   .          .          .          .          .          .          .     .      .
| 1   .      .     o________ÿo          .          .          .          .          .          .     .      .
|     .       ASSIGN_PCID    |          .          .          .          .          .          .     .      .
| 2   .      o_______________|          .          .          .          .          .          .     .      .
|     .      |  ASSIGN_PCID_RSP         .          .          .          .          .          .     .      .
| 3   .      |________________ÿo         .          .          .          .          .          .     .      .
|     .LOCATE_MESSAGE (local)|          .          .          .          .          .          .     .      .
| 4   o_______________________|          .          .          .          .          .          .     .      .
|     | LOCATE_MESSAGE (local) reply w/non-zero sense data   .          .          .          .     .      .
| 5   |_______________________ÿo         .          .          .          .          .          .     .      .
|    LOCATE_MESSAGE (distributed)        .          .          .          .          .          .     .      .
| 6   o_______________________|          .          .          .          .          .          .     .      .
|     | LOCATE_MESSAGE (distributed) reply       .          .          .          .          .     .      .
| 7   |_______________________ÿo         .          .          .          .          .          .     .      .
|     .      .     .          |    MDS-MU          .          .          .          .          .     .      .
| 8   .      .     .          |___________________ÿo          .          .          .          .     .      .
|     .      .     .          .          .          |     MDS-MU          .          .          .     .      .
| 9   .      .     .          .          .          |_____________________ÿo          .          .     .      .
|     .      .     .          .          .          .          .          | MDS-MU   .          .     .      .
| 10  .      .     .          .          .          .          .          |________ÿo          .     .      .
|     .      .     .          .          .          .     Confirmed       |          .          .     .      .
| 11  .      .     .          .          .         o_____________________|          .          .     .      .
|     .      .     .          .        OK          |          .          .          .          .     .      .
| 12  .      .     .         o___________________|          .          .          .          .     .      .
|     .      .     .          .          .          .          .          .          .          .     .      .
```

| Figure 9-67. MDS Router Chooses Outbound Session Based on Directory
| Services Network Search

| Figure 9-67 illustrates an MDS-MU which is routed based on information
| provided by control point directory services after invoking a distributed
| directory search.

| The items in the following list match the line numbers in the figure.

| 1. NN_A: An MS application program in the network node passes an MDS-MU
| to the MDS_ROUTER for forwarding. The MDS-MU could also have been
| received on a CPSVCMG session from a client end node.

| 2. NN_A: The MDS_ROUTER checks its local routing information for a match
| on the destination in the MDS Routing Information (X'1311') GDS
| variable.

| No match is found.

A request for the assignment of a PCID (ASSIGN_PCID signal) is sent to control point session services. The PCID is required for directory search requests.

3. NN_A: CP.SS returns the PCID to the MDS_ROUTER in the ASSIGN_PCID_RSP signal.

4. NN_A: MDS_ROUTER sends a request for a search of the local directory to CP.DS (LOCATE_MESSAGE with search executed set to local-only).

5. NN_A: CP.DS does not locate the destination in the local directory. A LOCATE_MESSAGE reply with non-zero sense data is passed back to the MDS_ROUTER.

6. NN_A: MDS_ROUTER sends a request for a network search to CP.DS (LOCATE_MESSAGE with search executed set to "distributed").

7. NN_A: CP.DS initiates a network search for the destination. The successful Locate results are passed back to the MDS_ROUTER in a LOCATE_MESSAGE reply with non-zero sense data.

The MDS_ROUTER then decides what outbound session is to be used for routing the MDS-MU. (either to the network node server of the destination NAU if the destination is an EN, or to the destination NAU itself, if the destination is a NN or an LU).

The Routing Verification Indicator in the Flags (X'90') subvector of the MDS Routing Information (X'1311') GDS variable is set to 1, indicating that the routing was based upon verified information.

8. NN_A: The MDS-MU, along with the outbound session information, is passed to the MDS_SEND TP.

9. NN_A: The MDS_SEND TP allocates a conversation with MDS_RECEIVE in NN_B, sends the MDS-MU, and requests confirmation.

10. NN_B: The MDS_RECEIVE TP receives the MDS-MU and passes it to the MDS_ROUTER.

11. NN_B: MDS_RECEIVE issues the verb CONFIRMED, to signal MDS_SEND in NN_A that the MDS-MU was received.

12. NN_A: MDS_SEND signals the MDS_ROUTER that the MDS-MU was sent successfully.

Destination Not Located by Directory Services

```
     _______________ Network Node NN_A _________________   ________________ Network Node NN_B _________________
    |               _______________ CP.MS _______________ | | _______________ CP.MS _______________             |
    |              |MS      MDS_     MDS_Re-    MDS_    || ||MDS_      MDS_Re-    MDS_      MS  |             |
    |CP.DS CP.SS |Appl    Router   ceive TP   Send TP|| ||Send TP   ceive TP   Router    Appl| CP.DS CP.SS|
    |              |_____________________________________|| ||_____________________________________|             |
    |_____________________________________________________| |_____________________________________________________|
       .      .     .          .         .         .           .         .         .         .     .      .
       .      .     . MDS-MU  .         .         .           .         .         .         .     .      .
  1    .      .     o________ÿo         .         .           .         .         .         .     .      .
       .       ASSIGN_PCID    |         .         .           .         .         .         .     .      .
  2    .      o______________|         .         .           .         .         .         .     .      .
       .      |  ASSIGN_PCID_RSP        .         .           .         .         .         .     .      .
  3    .      |______________ÿo         .         .           .         .         .         .     .      .
       .LOCATE_MESSAGE (local)|         .         .           .         .         .         .     .      .
  4    o______________________|         .         .           .         .         .         .     .      .
       | LOCATE_MESSAGE (local) reply w/non-zero sense data   .         .         .         .     .      .
  5    |______________________ÿo         .         .           .         .         .         .     .      .
      LOCATE_MESSAGE (distributed)      .         .           .         .         .         .     .      .
  6    o______________________|         .         .           .         .         .         .     .      .
       | LOCATE_MESSAGE (distributed) reply w/non-zero sense data       .         .         .     .      .
  7    |______________________ÿo         .         .           .         .         .         .     .      .
       .      . MDS error msg |         .         .           .         .         .         .     .      .
  8    .      .     o________|         .         .           .         .         .         .     .      .
       .      .     .          .         .         .           .         .         .         .     .      .
```

Figure 9-68. Error Case: Destination Resource Not Known to CP.DS

Figure 9-68 illustrates an error case, in which the destination in the MDS-MU could not be located by control point directory services (neither in the local directory nor by invoking a network search).

The items in the following list match the line numbers in the figure.

1. NN_A: An MS application program in the network node passes an MDS-MU to the MDS_ROUTER for forwarding. The MDS-MU could also have been received on a CPSVCMG session from a client end node.

2. NN_A: The MDS_ROUTER checks its local routing information for a match on the destination in the MDS Routing Information (X'1311') GDS variable.

   No match is found.

   A request for the assignment of a PCID (ASSIGN_PCID signal) is sent to control point session services. The PCID is required for directory search requests.

3. NN_A: CP.SS returns the PCID to the MDS_ROUTER in the ASSIGN_PCID_RSP signal.

4. NN_A: MDS_ROUTER sends a request for a search of the local directory to CP.DS (LOCATE_MESSAGE with search executed set to local-only).

5. NN_A: CP.DS does not locate the destination in the local directory. LOCATE_MESSAGE reply with non-zero sense data is passed back to the MDS_ROUTER.

6. NN_A: MDS_ROUTER sends a request for a network search to CP.DS (LOCATE_MESSAGE with search executed set to "distributed").

7. NN_A: CP.DS initiates a network search for the destination. The search is unsuccessful. LOCATE_MESSAGE reply with non-zero sense data is passed back to the MDS_ROUTER.

8. NN_A: The MDS_ROUTER creates an MDS error message indicating that the destination could not be located. This MDS error message is passed back to the originating application program (either locally or in a client end node).

Routing Error: Incorrect Cached Directory Information

```
     _______________ Network Node NN_A _______________   _______________ Network Node NN_B _______________
    |               _______________ CP.MS _____________ | _____________ CP.MS ____________                |
    |              |MS      MDS_     MDS_Re-    MDS_   || ||MDS_      MDS_Re-    MDS_      MS  |            |
    |CP.DS CP.SS   |Appl    Router   ceive TP   Send TP|| ||Send TP   ceive TP   Router    Appl| CP.DS CP.SS|
    |              |____________________________________|| ||____________________________________|            |
    |_____________________________________________________| |____________________________________________________|
      .      .     .         .         .         .         .         .         .         .     .      .
      .      .     . MDS-MU  .         .         .         .         .         .         .     .      .
 1    .      .     o________ÿo         .         .         .         .         .         .     .      .
      .        ASSIGN_PCID  |         .         .         .         .         .         .     .      .
 2    .      o______________|         .         .         .         .         .         .     .      .
      .      |  ASSIGN_PCID_RSP       .         .         .         .         .         .     .      .
 3    .      |______________ÿo         .         .         .         .         .         .     .      .
      LOCATE_MESSAGE (local) |         .         .         .         .         .         .     .      .
 4    o______________________|         .         .         .         .         .         .     .      .
      | LOCATE_MESSAGE (local) reply   .         .         .         .         .         .     .      .
 5    |______________________ÿo         .         .         .         .         .         .     .      .
      .      .     .         |    MDS-MU         .         .         .         .         .     .      .
 6    .      .     .         |___________________ÿo         .         .         .         .     .      .
      .      .     .         .         .         |     MDS-MU          .         .         .     .      .
 7    .      .     .         .         .         |_____________________ÿo         .         .     .      .
      .      .     .         .         .         .         .         | MDS-MU  .         .     .      .
 8    .      .     .         .         .         .         .         |________ÿo         .     .      .
      .      .     .         .         .         .     Confirmed       |         |         .     .      .
 9    .      .     .         .         .         o_____________________|         |         .     .      .
      .      .     .         .       OK          |         .         .         |         .     .      .
 10   .      .     .         o___________________|         .         .         |         .     .      .
      .      .     .         .         .         .         MDS error msg, MDS-MU |         .     .      .
 11   .      .     .         .         .         .         o___________________|         .     .      .
      .      .     .         .         MDS error msg, MDS-MU |         .         .         .     .      .
 12   .      .     .         .         o_____________________|         .         .         .     .      .
      .      .   MDS error msg, MDS-MU |         .         .         .         .         .     .      .
 13   .      .     .         o________|         .         .         .         .         .     .      .
      .      .     .         |         |     Confirmed       .         .         .         .     .      .
 14   .      .     .         |         |_____________________ÿo         .         .         .     .      .
      .      .     .         |         .         .         |         OK        .         .     .      .
 15   .      .     .                   .         .         |___________________ÿo         .     .      .
                        (to part 2)
```

Figure 9-69. Error Case: CP.DS Provides Incorrect Directory Information from Cache (Part 1 of 2)

Figure 9-69 illustrates a more complex error case. An MDS-MU (with a remote end node destination) is routed through the network node server

```
| based on information maintained locally by control point directory
| services (no network search invoked).  However, that locally cached
| directory information turns out to be incorrect, either because the end
| node has left the network, or because it has switched to a different
| network node server.


| The items in the following list match the line numbers in the figure.


| 1.  NN_A:  An MS application program in the network node passes an MDS-MU
|     to the MDS_ROUTER for forwarding.


| 2.  NN_A:  The MDS_ROUTER checks its local routing information for a match
|     on the destination in the MDS Routing Information (X'1311') GDS
|     variable.


|     No match is found.


|     A request for the assignment of a PCID (ASSIGN_PCID signal) is sent to
|     control point session services.  The PCID is required for directory
|     search requests.


| 3.  NN_A:  CP.SS returns the PCID to the MDS_ROUTER via the
|     ASSIGN_PCID_RSP signal.


| 4.  NN_A:  MDS_ROUTER sends a request for a search of the local directory
|     cache to CP.DS (LOCATE_MESSAGE with search executed set to
|     local-only).


| 5.  NN_A:  CP.DS locates the resource in the local directory cache and
|     passes the information back to the MDS_ROUTER as a LOCATE_MESSAGE
|     reply.


|     The MDS_ROUTER then decides what outbound session is to be used for
|     routing the MDS-MU.  The destination is an end node, and a session to
|     the network node server will be used.


|     The Routing Verification Indicator in the Flags (X'90') subvector of
|     the MDS Routing Information (X'1311') GDS variable is set to 0,
|     indicating that the routing was based upon unverified information.


| 6.  NN_A:  The MDS-MU, along with the outbound session information, is
|     passed to the MDS_SEND TP.


| 7.  NN_A:  The MDS_SEND TP allocates a conversation with MDS_RECEIVE in
|     NN_B, sends the MDS-MU, and requests confirmation.


| 8.  NN_B:  The MDS_RECEIVE TP receives the MDS-MU and passes it to the
|     MDS_ROUTER.


| 9.  NN_B:  MDS_RECEIVE issues the verb CONFIRMED, to signal MDS_SEND in
|     NN_A that the MDS-MU was received.


| 10. NN_A:  MDS_RECEIVE signals the MDS_ROUTER that the MDS-MU was sent
|     successfully.


| 11. NN_B:  The MDS_ROUTER examines the MDS-MU, but finds no match for the
|     destination in its EN status list.  An MDS error message is built
|     indicating that the intended destination is not in the domain of the
|     NN.  Because the Routing Verification Indicator of the Flags (X'90')
|     subvector has a value of 0 (routing was based on unverified
|     information), the original MDS-MU is imbedded within the MDS Error
|     message.  This will allow the MDS_ROUTER which forwarded the MDS-MU to
|     attempt to reroute it.


|     The MDS_ROUTER checks its local routing information for a match on the
|     Agent Unit-of-Work Correlator (X'1549') GDS variable or the
|     destination in the MDS Routing Information (X'1311') GDS variable of
|     the MDS error message.


|     A match is found.  This match could either be in the active
|     transaction control blocks if a reply is expected or in the other
|     local information.


|     If a match had not been found (i.e., the MDS-MU was a request without
|     reply), then a directory search would have been required to determine
|     the correct routing for the MDS error message.


|     The MDS-MU, along with the outbound session information, is passed to
|     the MDS_SEND TP.


| 12. NN_B:  The MDS_SEND TP allocates a conversation with MDS_RECEIVE in
```

```
|     NN_A, sends the MDS error message, and requests confirmation.


| 13. NN_A:  The MDS_RECEIVE TP receives the MDS-MU.  and passes it to the
|     MDS_ROUTER.


| 14. NN_A:  MDS_RECEIVE issues the verb CONFIRMED, to signal MDS_SEND that
|     the MDS-MU was received.


| 15. NN_B:  MDS_RECEIVE signals the MDS_ROUTER that the MDS-MU was sent
|     successfully.



|    _______________ Network Node NN_A _________________   _______________ Network Node NN_B _________________
|   |               _______________ CP.MS _______________ | |_______________ CP.MS _______________             |
|   |            |MS      MDS_     MDS_Re-    MDS_   || ||MDS_      MDS_Re-    MDS_      MS  |             |
|   |CP.DS CP.SS |Appl    Router   ceive TP   Send TP|| ||Send TP   ceive TP   Router    Appl| CP.DS CP.SS|
|   |            |____________________________________|| ||____________________________________|             |
|   |_____________________________________________________| |_____________________________________________________|
|                     (from part 1)
|     .      .      .                .         .           .         .         .         .     .      .
|     .      ASSIGN_PCID    |         .         .           .         .         .         .     .      .
| 16  .      o_______________|         .         .           .         .         .         .     .      .
|     .      |  ASSIGN_PCID_RSP        .         .           .         .         .         .     .      .
| 17  .      |_______________ÿo        .         .           .         .         .         .     .      .
|    LOCATE_MESSAGE (distributed)      .         .           .         .         .         .     .      .
| 18  o_______________________|         .         .           .         .         .         .     .      .
|     | LOCATE_MESSAGE (distributed) reply       .           .         .         .         .     .      .
| 19  |_______________________ÿo       .         .           .         .         .         .     .      .
|     .      .     .         |    MDS-MU         .           .         .         .         .     .      .
| 20  .      .     .         |_________________ÿo            .         .         .         .     .      .
|     .      .     .         .         .         | (different remote NN)         .         .     .      .
|     .      .     .         .         .         |       MDS-MU        .         .         .     .      .
| 21  .      .     .         .         .         |_____________________ÿo         .         .     .      .
|     .      .     .         .         .         .           .         | MDS-MU  .         .     .      .
| 22  .      .     .         .         .         .           .         |________ÿo         .     .      .
|     .      .     .         .         .         .     Confirmed       |         .         .     .      .
| 23  .      .     .         .         .        o_____________________|         .         .     .      .
|     .      .     .         .        OK         |           .         .         .         .     .      .
| 24  .      .     .        o_________________|           .         .         .         .     .      .
|     .      .     .         .         .         .           .         .         .         .     .      .
```

| Figure 9-70. Error Case: CP.DS Provides Incorrect Directory Information
| from Cache (Part 2 of 2)

| 16. NN_A: The MDS_ROUTER examines the report code in the MDS error
| message and determines that the routing failure occurred after using
| cached directory information (this knowledge is based upon the Routing
| Verification indicator in the MDS Routing Information (X'1311') GDS
| variable of the original MDS-MU)

| A request for the assignment of a PCID (ASSIGN_PCID signal) is sent to
| control point session services. The PCID is required for directory
| search requests.

| 17. NN_A: CP.SS returns the PCID to the MDS_ROUTER in the ASSIGN_PCID_RSP
| signal.

| 18. NN_A: The MDS_ROUTER sends a request for a network search for the
| destination to CP.DS (LOCATE_MESSAGE with search executed set to
| "distributed").

| 19. NN_A: CP.DS initiates a network search for the destination. The
| successful Locate results are passed back to the MDS_ROUTER.

| The MDS_ROUTER then decides what outbound session is to be used for
| routing the MDS-MU. A successful network search will have located the
| current network node server "NN_C") for the destination end node.

| The Routing Verification Indicator in the Flags (X'90') subvector of
| the MDS Routing Information (X'1311') GDS variable is set to 1,
| indicating that the routing was based upon verified information.

| 20. NN_A: The MDS-MU, along with the outbound session information, is
| passed to the MDS_SEND TP.

| 21. NN_A: The MDS_SEND TP allocates a conversation with MDS_RECEIVE in
| NN_C, sends the MDS-MU, and requests confirmation.

| 22. NN_C: The MDS_RECEIVE TP receives the MDS-MU and passes it to the
| MDS_ROUTER.

| 23. NN_C: MDS_RECEIVE issues the verb CONFIRMED, to signal MDS_SEND in
| NN_A that the MDS-MU was received.

24. NN_A: MDS_RECEIVE signals the MDS_ROUTER that the MDS-MU was sent successfully.

Routing Error: Unknown MS Application Program

```
     _______________ Network Node NN_A _________________      _______________ Network Node NN_B _________________
    |               _______________ CP.MS _______________ |    | _______________ CP.MS _______________            |
    |              |MS      MDS_     MDS_Re-    MDS_    ||    ||MDS_      MDS_Re-    MDS_      MS  |             |
    |CP.DS CP.SS |Appl    Router   ceive TP   Send TP||    ||Send TP   ceive TP   Router    Appl| CP.DS CP.SS|
    |              |_____________________________________||    ||_____________________________________|             |
    |_____________________________________________________|    |_____________________________________________________|
      .      .     .         .         .         .             .         .         .         .     .      .
      .      .     . MDS-MU  .         .         .             .         .         .         .     .      .
 1    .      .     o________ÿo         .         .             .         .         .         .     .      .
      .      .     .         |    MDS-MU         .             .         .         .         .     .      .
 2    .      .     .         |___________________ÿo            .         .         .         .     .      .
      .      .     .         .         .         |        MDS-MU          .         .         .     .      .
 3    .      .     .         .         .         |_______________________ÿo            .         .     .      .
      .      .     .         .         .         .             .         | MDS-MU  .         .     .      .
 4    .      .     .         .         .         .             .         |________ÿo         .     .      .
      .      .     .         .         .         .        Confirmed      |         |         .     .      .
 5    .      .     .         .         .        o_______________________|         |         .     .      .
      .      .     .         .        OK         |             .         .         |         .     .      .
 6    .      .     .         o___________________|             .         .         |         .     .      .
      .      .     .         .         .         .             .    MDS error msg  |         .     .      .
 7    .      .     .         .         .         .             o___________________|         .     .      .
      .      .     .         .         .     MDS error msg      |         .         .         .     .      .
 8    .      .     .         .         o_______________________|         .         .         .     .      .
      .      .     .     MDS error msg |                       .         .         .         .     .      .
 9    .      .     .         o________|                        .         .         .         .     .      .
      .      .     .         |         |       Confirmed        .         .         .         .     .      .
 10   .      .     .         |         |_______________________ÿo         .         .         .     .      .
      .      . MDS error msg |         .         .             |         .         .         .     .      .
 11   .      .     o________|         .         .             |         .         .         .     .      .
      .      .     .         .         .         .             |        OK         .         .     .      .
 12   .      .     .         .         .         .             |___________________ÿo         .     .      .
      .      .     .         .         .         .             .         .         .         .     .      .
```

Figure 9-71. Error Case: Routing Failure after the First Hop.. The MDS error message is returned to the sending application.

Figure 9-71 illustrates the case of an undeliverable MDS-MU that was not caused by locally cached directory services information.

The items in the following list match the line numbers in the figure.

1. NN_A: An MS application program in the network node passes an MDS-MU to the MDS_ROUTER for forwarding.

   The MDS_ROUTER checks its local routing information for a match on the destination in the MDS Routing Information (X'1311') GDS variable.

   A match is found.

   The MDS_ROUTER then decides what outbound session is to be used for routing the MDS-MU.

2. NN_A: The MDS-MU, along with the outbound session information, is passed to the MDS_SEND TP.

3. NN_A: The MDS_SEND TP allocates a conversation with MDS_RECEIVE in NN_B, sends the MDS-MU, and requests confirmation.

4. NN_B: The MDS_RECEIVE TP receives the MDS-MU and passes it to the MDS_ROUTER.

5. NN_B: MDS_RECEIVE issues the verb CONFIRMED, to signal MDS_SEND in NN_A that the MDS-MU was received.

6. NN_A: MDS_SEND signals the MDS_ROUTER that the MDS-MU was sent successfully.

7. NN_B: The MDS_ROUTER is unable to deliver the MDS-MU because the destination application program is unknown.

   An MDS error message is created indicating the nature of the problem.

   The MDS_ROUTER checks its local routing information for a match on the Agent Unit-of-Work Correlator (X'1549') GDS variable or the

```
|     destination in the MDS Routing Information (X'1311') GDS variable of
|     the MDS error message.


|     A match is found.  This match could either be in the active
|     transaction control blocks if a reply is expected or in the other
|     local information.


|     The MDS_ROUTER then decides what outbound session is to be used for
|     routing the MDS error message.


| 8.  NN_B:  The MDS-MU, along with the outbound session information, is
|     passed to the MDS_SEND TP.


| 9.  NN_B:  The MDS_SEND TP allocates a conversation with MDS_RECEIVE in
|     NN_A, sends the MDS error message, and requests confirmation.


| 10. NN_A:  The MDS_RECEIVE TP receives the MDS-MU and passes it to the
|     MDS_ROUTER.


| 11. NN_A:  MDS_RECEIVE issues the verb CONFIRMED, to signal MDS_SEND in
|     NN_B that the MDS-MU was received.


| 12. NN_A:  The MDS_ROUTER examines the report code in the MDS error
|     message and passes it back to the originating application program.
|     Since the error was not one related to directory information, no retry
|     is performed in the MDS_ROUTER.


| 13. NN_B:  MDS_SEND signals the MDS_ROUTER that the MDS-MU was sent
|     successfully.
```

| 2.9.2.7 MULTIPLE_DOMAIN_SUPPORT Optional Subset 3 (High-Performance Option)

```
  This optional subset provides extensions to MULTIPLE_DOMAIN_SUPPORT to
  meet the performance requirements of MS application programs with higher
  transaction rates.


| MULTIPLE_DOMAIN_SUPPORT, as described in the base and optional subsets 1
| and 2, uses brief conversations over shared sessions (those with the
  SNASVCMG and CPSVCMG mode names) to minimize the overall network impact
  for transporting MS data.  In addition, MDS uses LU 6.2 confirmations (via
  the CONFIRM/CONFIRMED verbs) to guarantee the reliable delivery of the
  application program data.


  For MS application programs with higher transaction rates, however, this
  use of brief conversations, shared sessions, and confirmations imposes
  constraints on throughput.  The transaction rates of these application
  programs are better met through the use of persistent conversations over
  dedicated sessions between the origin and destination nodes and the
  elimination of transport-level confirmations.  In place of transport-level
  confirmations, these application programs may either implement application
  program-level error detection (such as the use of consecutive sequence
  numbers on messages) or accept the occasional loss of messages for those
  cases in which data integrity is not critical.
```

Subtopics:

- 2.9.2.7.1 Function Provided
- 2.9.2.7.2 Implementation Requirements

2.9.2.7.1 Function Provided

```
| Optional Subset 3 (High-Performance Option) requires the functions
  described in the MDS base.  This optional subset provides for the
  transport of MDS-MUs over dedicated user-mode sessions (not the shared
  SNASVCMG and CPSVCMG mode sessions).  The MDS_ROUTER uses a different set
  of service transaction programs to perform this transport (MDS_HP_SEND and
  MDS_HP_RECEIVE) in a way that minimizes LU 6.2 path length in the origin
  and destination nodes.  Minimal error detection and reporting is provided,
  so MS application programs that select this transport option must make
  provisions for data integrity.


  The restrictions that apply to the use of this option are:


  1.  An MS application program may be registered only for a single
      user-defined mode name.  This is the mode name that will be used for
      when the MS application program sends an MDS-MU.  There is no
      restriction that an incoming MDS-MU must come over sessions with this
```

mode name, however. It may continue to use the default routing of MDS (over CPSVCMG and SNASVCMG sessions).

2. The user-mode names must have available session limits of at least (2,1,1), so that MULTIPLE_DOMAIN_SUPPORT can allocate a (contention-winner, contention-loser) pair of sessions. The actual session limits of the mode name can be higher than (2,1,1), allowing other TPs to use the mode as well, but MDS will not allocate more than 2 sessions using the mode name to a given destination.

3. An MS application program using MDS HPO must implement some error detection and recovery scheme appropriate for its data. MDS HPO will notify the MS application program whenever possible about lost data, but in some cases MDS-MUs may be lost without notification. When an error is detected for a given mode name, every MS application program registered for that mode name is notified of the problem. The MS application program must be able to perform its own recovery.

4. There is no coordination between MDS and MS_CAPS for user-mode sessions. Application program functions that rely upon an MS Capabilities exchange for initialization are restricted to the use of the default sessions (CPSVCMG and SNASVCMG mode names).

2.9.2.7.2 Implementation Requirements

The requirements for implementing MULTIPLE_DOMAIN_SUPPORT Optional Subset 1 (High-Performance Option) are described as a set of changes to MDS as described in "MULTIPLE_DOMAIN_SUPPORT Base Subset" in topic 2.9.2.4 consisting of a subset of CP management services, including:

° Modifications to the following protocol boundaries:

- "Protocol Boundary O--Register Application Program Name" in topic 2.9.1.11

```
 ___ New optional parameter ____________________________________
|                                                               |
| 3.  Session mode name (must be present if MS application      |
|     program will request MDS high-performance transport of any |
|     MDS-MUs)                                                  |
|                                                               |
|_______________________________________________________________|
```

- "Protocol Boundary O--MDS-MU Received" in topic 2.9.1.12

```
 ___ New optional parameter ____________________________________
|                                                               |
| 2.  MDS high-performance indicator (present if MDS-MU was     |
|     received by MDS high-performance transport)               |
|                                                               |
|_______________________________________________________________|
```

- "Protocol Boundary R--Send MDS-MU" in topic 2.9.1.13

```
 ___ New parameter value _______________________________________
|                                                               |
| Addition of a new value to the parameter for type of routing  |
| requested (currently only DIRECT and DEFAULT are defined).    |
| The value HIGH_PERFORMANCE indicates that the MDS-MU is to be |
| sent over a user-mode session (the mode must have been        |
| previously specified on Protocol Boundary O--Register         |
| Application Program Name).                                    |
|                                                               |
|_______________________________________________________________|
```

° Modifications to the MDS_ROUTER processing described in "MULTIPLE_DOMAIN_SUPPORT Base Subset" in topic 2.9.2.4

° Addition of two service transaction programs:

- MDS_HP_SEND

  Sends MDS-MUs over user-mode sessions

- MDS_HP_RECEIVE

  Receives MDS-MUs over user-mode sessions

Modifications to MDS_ROUTER:

To implement MULTIPLE_DOMAIN_SUPPORT Optional Subset 1 (High-Performance Option), the following modifications are made to the MDS_ROUTER:

- "MDS Internal Signal - MDS Routing Unit" in topic 2.9.2.4.4

  New parameter values

  Addition of the value "HIGH_PERFORMANCE" to

  2. Type of routing requested

  Addition of the following values to

  3. Routing status

  - "TERMINATE CONVERSATION NORMALLY"
  - "TERMINATE CONVERSATION ABNORMALLY"

  Addition of the value "User-defined mode" to session mode name for:

  8. Inbound session information

  9. Outbound session information

- "Local Application Program Routing List Processing for Register" in topic 2.9.2.4.8

  Additional processing

  The session mode name specified in Protocol Boundary O, if present, is recorded in the application program routing list.

- "Processing Send MDS-MU" in topic 2.9.2.4.8

  Additional processing

  When the MDS Routing Unit is prepared from Protocol Boundary R, the requested routing type may be HIGH_PERFORMANCE as well as DEFAULT or DIRECT. If an MS application program requests HIGH_PERFORMANCE but it has not previously registered the session mode name to be used, an error indication is returned to the application program.

- "MDS Routing Algorithm" in topic 2.9.2.4.8

  Additional processing

  The following processing is added after

  6. If Type of Routing Requested is DIRECT...

  If Type of Routing Requested is HIGH_PERFORMANCE, the MDS Routing Unit is completed as follows:

  1. Session Partner Name is set to the destination location name in MDS Routing Information (X'1311') GDS variable.

  2. Session Mode Name is set to the user-mode name recorded in the application program routing list for this MS application program.

  3. The Routing Verification Indicator in the Flags (X'90') subvector of the MDS Routing Information (X'1311') GDS variable is set to 1.

  4. Routing Status is set to COMPLETE.

  5. The MDS Routing Unit is passed to "MDS_SEND Manager" in topic 2.9.2.4.8.

  6. This process ends.

- "MDS_SEND Manager" in topic 2.9.2.4.8

  Additional processing

  This process is modified to maintain a list of all active instances of the MDS_HP_SEND and MDS_HP_RECEIVE TPs in addition to the MDS_SEND TP. When an MDS Routing Unit (OUTBOUND) is processed

```
| that contains a session mode name other than SNASVCMG or CPSVCMG,  |
| the MDS_SEND manager assumes that the TP name is MDS_HP_SEND.      |
| Since any number of user-defined mode names may be active to the   |
| same session partner, the list of active instances of MDS_HP_SEND  |
| must be searched for a matching mode name, as well.                |
|                                                                     |
| When an MDS Routing Unit (INBOUND) with routing status ERROR is    |
| received from MDS_HP_RECEIVE, the list of active TPs is searched   |
| for an MDS_HP_SEND for the same session partner and mode name.  If |
| one is found, an MDS Routing Unit (OUTBOUND) is prepared with the  |
| routing status set to TERMINATE_CONVERSATION, and is passed to it. |
|                                                                     |
| When an MDS Routing Unit (REPLY TO OUTBOUND) with routing status   |
| ERROR is received from MDS_HP_SEND, the list of active TPs is       |
| searched for an MDS_HP_RECEIVE for the same session partner and     |
| mode name.  If one if found, an MDS Routing Unit (OUTBOUND) is      |
| prepared with the routing status set to TERMINATE_CONVERSATION,     |
| and is passed to it.                                                |
|                                                                     |
|_____________________________________________________________________|
```

- "Routing MDS-MU Received to Local MS Application Programs" in topic 2.9.2.4.8

The following processing is added after

2. If a matching entry is *not* found:

```
 ___ Additional processing ____________________________________________
|                                                                     |
| If a matching entry is found and the MDS Routing Unit (INBOUND)     |
| indicates that the session mode name is not SNASVCMG or CPSVCMG,    |
| then MDS-MU was sent using the MDS high-performance option.  If     |
| the application program did not previously register that it         |
| supports the use of HPO, then the MDS-MU must be rejected.          |
|                                                                     |
| 1.  The Routing Status flag is set to ERROR.                        |
|                                                                     |
| 2.  Sense data is set to X'08A8 0011' (MDS HPO not supported by MS  |
|     application program).                                           |
|                                                                     |
| 3.  "Building MDS Error Messages and Alerts" in topic 2.9.2.4.8 is  |
|     invoked.                                                        |
|                                                                     |
| 4.  The MDS error message (in an MDS Routing Unit) is passed to     |
|     "MDS Routing Algorithm" in topic 2.9.2.4.8.                     |
|                                                                     |
|_____________________________________________________________________|
```

The following processing is added after

3. If a matching entry is found:

```
 ___ Additional processing ____________________________________________
|                                                                     |
| If a matching entry is found and the MDS Routing Unit (INBOUND)     |
| indicates that the session mode name is not SNASVCMG or CPSVCMG,    |
| then MDS-MU was sent using the MDS high-performance option.  If     |
| the MDS Routing Unit (INBOUND) carries the sense value X'08A8       |
| 0012' (Irrecoverable failure of user-mode session), a failure has   |
| been detected on one of the user-mode sessions in use by MDS HPO.   |
| Each MS application program using that mode name must be notified   |
| of the error so that it can determine if any error recovery is      |
| appropriate.                                                        |
|                                                                     |
| An MDS error message is built and passed to each application        |
| program registered for the indicated mode name.  The MDS error      |
| message contains the following:                                     |
|                                                                     |
| °   The destination is the local MS application program.            |
|                                                                     |
| °   The origin is the local MDS_ROUTER.                             |
|                                                                     |
| °   The Agent Unit-of-Work Correlator (X'1549') GDS variable        |
|     contains some unique value.                                     |
|                                                                     |
| °   The sense data is set to X'08A8 0012' (Irrecoverable failure    |
|     of user-mode session).                                          |
|                                                                     |
| °   The reported-on location is the remote node to which            |
|     communication has failed.                                       |
|                                                                     |
|_____________________________________________________________________|
```

- "Processing MDS Routing Unit (Inbound) from TPs" in topic 2.9.2.4.8

```
 ___ Additional processing ____________________________________________
|                                                                     |
| When MDS_HP_RECEIVE receives an error return code from              |
| RECEIVE_AND_WAIT, it creates an MDS Routing Unit (Inbound) with     |
| the session partner and mode name information, and with routing     |
| status ERROR.  Processing MDS Routing Unit (Inbound) in that case   |
```

```
| must do the following:                                              |
|                                                                     |
| 1.  Invoke "MDS_SEND Manager" in topic 2.9.2.4.8 so that the        |
|     active copy of MDS_HP_SEND for that session partner and mode    |
|     name can be instructed to terminate the conversation.           |
|                                                                     |
| 2.  Invoke "Routing MDS-MU Received to Local MS Application         |
|     Programs" in topic 2.9.2.4.8 so that an MDS error message       |
|     describing the loss of the session partner can be passed to     |
|     each local MS application program registered for the mode       |
|     name.                                                           |
|                                                                     |
|_____________________________________________________________________|
```

Overview of Transaction Programs for MDS HPO:

```
 ____________________________________________
|                                            |
|             M D S _ R O U T E R            |
|                                            |
|____________________________________________|
                  "                  |
                  |                  |
      MDS Routing |        MDS Routing |
   Unit (inbound) |    Unit (outbound) |
                  |                  |
                  |

        ______________      _____________
     ______________   |   _____________   |
     | MDS_HP_RECEIVE |  |   | MDS_HP_SEND  |  |
     |  X'23F0F0F2'   |__|   |              |__|
     |________________|      |______________|
              "                  |
  RECEIVE_AND_|           SEND_DATA |
  WAIT (after |            (after   |
    attach)   |           ALLOCATE) |
              |
 ____________________________________________
|                                            |
|                L  U     6 . 2              |
|                                            |
|____________________________________________|
            o o                  o o
            o o     user mode    o o
            o o     sessions     o o
            o o                  o o
```

Figure 9-72. Service Transaction Programs for MULTIPLE_DOMAIN_SUPPORT Optional Subset 1 (High-Performance Option). Arrows illustrate the direction of data flow, not control information.

Figure 9-72 illustrates the service transaction programs for MULTIPLE_DOMAIN_SUPPORT Optional Subset 1 (High-Performance Option) and their relationship to the LU, the sessions, and the MDS router. Information about these transaction programs is summarized in Table 9-15.

Table 9-15. LU 6.2 Transaction Programs for MULTIPLE_DOMAIN_SUPPORT Optional Subset 1 (High-Performance Option)

Name (hex)	Descriptive name	Function
	MDS_HP_SEND	MULTIPLE_DOMAIN_SUPPORT program for sending MDS-MUs. over user-mode sessions, as described in MULTIPLE_DOMAIN_SUPPORT Optional Subset 1 (High-Performance Option). This TP is never the target of an Attach, so no registered name is required.
X'23F0F0F2'	MDS_HP_RECEIVE	MULTIPLE_DOMAIN_SUPPORT transaction program for receiving MDS-MUs user-mode sessions, as described in MULTIPLE_DOMAIN_SUPPORT Optional Subset 1 (High-Performance Option).

LU 6.2 Verb Sequences:

The following diagrams illustrate the sequences of LU 6.2 verbs that make up the interactions of the MDS_HP_SEND and MDS_HP_RECEIVE service transaction programs.

These illustration are based on the following implementation assumptions. Other implementation designs are possible.

1. MDS_HP_SEND issues SEND_DATA for a buffer that contains the entire MDS-MU.

   An equally valid implementation choice would be to issue SEND_DATA multiple times, sending a portion of the MDS-MU each time. This might be necessary because of implementation restrictions on send buffer size, for example. The MDS_HP_RECEIVE TP in another implementation will not be aware of the method used by the sending TP.

2. MDS_HP_RECEIVE issues RECEIVE_AND_WAIT FILL(LL) one or more times as needed to receive the MDS-MU. An MDS-MU longer than the receive buffer would force the TP to issue RECEIVE_AND_WAIT multiple times (until DATA_COMPLETE were indicated by the LU). The application program logical record is reassembled by MDS_HP_RECEIVE as the parts of the record are received.

   An equally valid implementation choice would be to issue RECEIVE_AND_WAIT one time with a buffer length of 2, just to get the total length of the MDS-MU, then to allocate a buffer of exactly the right size so that RECEIVE_AND_WAIT can be issued a single time for the rest of the MDS-MU.

These illustrations do not show all the LU 6.2 verb parameters required in an actual implementation, nor do they necessarily show the exact syntax required. More information is given in the descriptions of the TPs.

Sending MDS-MUs:

```
 ____________________________________________________________________
|                                                                    |
|       MDS_HP_SEND                              MDS_HP_RECEIVE      |
|      ______________________________________________________________|_
| 1     Allocate                                                     |
|         .                                                          |
|         .                                                          |
|         RC=OK                                                      |
| 2     Send_data (MDS-MU_1)                                         |
|         RC=OK                                                      |
| 3     Confirm _________________                                    |
|         .                     |                                    |
| 4       .                     |____________ÿ Receive_and_Wait      |
|         .                                      RC=OK               |
|         .                                      What=DATA   (MDS-MU_1)|
| 5       .                                    Receive_and_Wait      |
|         .                                      RC=OK               |
|         .                                      What=Confirm        |
| 6       .                     _______________ Confirmed            |
|         .                    |                 RC=OK               |
| 7       .                    |               Receive_and_Wait      |
|         .                    |                  .                  |
| 8       RC=OK ________________|                 .                  |
| 9     Send_data (MDS-MU_2)                       .                  |
|         RC=OK                                    .                  |
| 10    Send_data (MDS-MU_3)                       .                  |
|         RC=OK                                    .                  |
| 11    Flush ___________________                  .                  |
|         RC=OK                 |                  .                  |
| 12                            |_____________ÿ    RC=OK              |
|                                                  What=DATA   (MDS-MU_2)|
| 13                                             Receive_and_Wait    |
|                                                  RC=OK             |
|                                                  What=DATA   (MDS-MU_3)|
| 14                                             Receive_and_Wait    |
|                                                  .                 |
|                                                  .                 |
| 15    Deallocate(confirm)_____                   .                 |
|         .                     |                  .                 |
| 16      .                     |_____________ÿ    RC=OK             |
|         .                                        What=Confirm_deallocat|
| 17      .                     _______________ Confirmed            |
|         .                    |                 RC=OK               |
| 18      .                    |               Deallocate(local)     |
|         .                    |                  RC=OK              |
| 19      RC=OK ________________|                                     |
|                                                                    |
|____________________________________________________________________|
```

Figure 9-73. Verb Sequence: Sending MDS-MUs

Figure 9-73 illustrates the establishment of a long-running conversation between MDS_HP_SEND and MDS_HP_RECEIVE, the sending of MDS-MUs, and the eventual termination of the conversation.

1. MDS_HP_SEND allocates the conversation.

2. The first MDS-MU is sent.

3. Since this is the first MDS-MU to be sent on this conversation, CONFIRM is issued. This verb will cause the data to be sent immediately and will suspend the TP until a response is received from the remote TP, verifying that the conversation is indeed ready for transport of MDS-MUs. Subsequent MDS-MUs will not be confirmed.

4. The MDS-MU is received.

   MDS_HP_RECEIVE uses FILL(LL) to receive the MDS-MU. Multiple receives may be necessary to receive the entire logical record (dependent upon implementation buffers).

5. MDS_HP_RECEIVE issues RECEIVE_AND_WAIT, and is given the CONFIRM request.

6. CONFIRMED is issued.

7. MDS_HP_RECEIVE goes into receive state.

8. MDS_HP_SEND receives the indication that the first MDS-MU was received successfully.

9. The second MDS-MU is sent.

10. The third MDS-MU is sent.

11. FLUSH is issued.

12. The second MDS-MU is received.

13. The third MDS-MU is received.

14. MDS_HP_RECEIVE goes into receive state.

15. After some period of time, MDS_HP_SEND is instructed by MDS_ROUTER to terminate the conversation. This action may be taken because of operator input or because an installation-specified inactivity timeout has occurred.

    DEALLOCATE (CONFIRM) is issued.

16. The CONFIRM_DEALLOCATE indication is received.

17. CONFIRMED is issued.

18. MDS_HP_RECEIVE DEALLOCATES the conversation locally.

19. MDS_HP_SEND receives the indication that DEALLOCATE(CONFIRM) completed OK.

MDS_HP_RECEIVE TP Failure:

```
 ___________________________________________________________________
|                                                                   |
|        MDS_HP_SEND                              MDS_HP_RECEIVE    |
|        _________________________________________________________|__
| 1      Allocate                                                   |
|          .                                                        |
|          .                                                        |
|          RC=OK                                                    |
| 2      Send_data (MDS-MU_1)                                       |
|          RC=OK                                                    |
| 3      Confirm _________________                                  |
|          .                      |                                 |
| 4        .                      |____________ÿ Receive_and_Wait  |
|          .                                        RC=OK           |
|          .                                        What=DATA   (MDS-MU_1)|
| 5        .                                      Receive_and_Wait  |
|          .                                        RC=OK           |
|          .                                        What=Confirm    |
| 6        .                      _______________ Confirmed        |
|          .                     |                  RC=OK           |
| 7        .                     |                Receive_and_Wait  |
|          .                     |                  .               |
| 8        RC=OK _________________|                 .              |
| 9      Send_data (MDS-MU_2)                        .               |
|          RC=OK                                     .               |
```

```
| 10     Send_data (MDS-MU_3)                     .                     |
|          RC=OK                                  .                     |
| 11     Flush _________________                  .                     |
|          RC=OK                 |                .                     |
| 12                             |_____________ÿ  RC=OK                 |
|                                                 What=DATA   (MDS-MU_2)|
| 13                                            Receive_and_Wait        |
|                                                 RC=OK                 |
| 14                                              What=DATA             |
|                                                     (part of MDS-MU_3)|
|                                                                       |
| 15                                                *** Error ***       |
|                                                 (program failure)     |
|                                                                       |
| 16                              _______________ Deallocate(Abend_Prog)  |
|                                |                RC=OK                 |
|                                |                                      |
| 17     Send_data (MDS-MU_4)    |                                      |
|          RC=Dealloc_ _________|                                     |
|             Abend_Prog                                                |
| 18     Deallocate(local)                                              |
|          RC=OK                                                        |
|                                                                       |
|_______________________________________________________________________|
```

Figure 9-74. Verb Sequence: MDS_HP_RECEIVE TP Failure

Figure 9-74 illustrates an error case. The first MDS-MU was successfully sent and received with confirmation. The second MDS-MU was also successfully sent and received. The third MDS-MU was not successfully received because of a program failure which caused MDS_HP_RECEIVE to issue DEALLOCATE(ABEND_PROG). This verb causes the conversation to be terminated, but MDS_HP_SEND is not notified of the failure until it issues its next verb, a SEND_DATA for a fourth MDS-MU. MDS_HP_SEND reports a send failure for the fourth MDS-MU to the MDS_ROUTER, but there is no indication that the previous MDS-MU was not successfully received.

1. A conversation is Allocated.

2. The first MDS-MU is sent.

3. CONFIRM is issued.

4. The MDS-MU is received.

5. MDS_HP_RECEIVE goes into receive state, and the CONFIRM indication is received.

6. CONFIRMED is issued.

7. MDS_HP_RECEIVE goes into receive state

8. CONFIRMED indication is received.

9. The second MDS-MU is sent.

10. The third MDS-MU is sent.

11. FLUSH is issued.

12. The second MDS-MU is received.

13. MDS_HP_RECEIVE goes into receive state.

14. The first buffer of the third MDS-MU is received.

15. MDS_HP_RECEIVE experiences a program error (e.g., unable to allocate storage).

16. MDS_HP_RECEIVE issues DEALLOCATE (ABEND_PROG).

17. Some time later, MDS_HP_SEND issues SEND_DATA to send the next MDS-MU. It receives an error return code of DEALLOCATE_ABEND_PROG.

18. MDS_HP_SEND issues DEALLOCATE (LOCAL) to terminate the conversation locally.

An error indication is passed back to the MDS_ROUTER for the MDS-MU that could not be successfully sent. Since the success of MDS-MUs sent earlier on the conversation is not clear, MDS_ROUTER reportd the failure to all MS application programs that have registered for use of the session mode name. These MS application programs have the responsibility for initiating appropriate error recovery (such as

attempting to resend previous records).

If MDS_HP_RECEIVE in this node is a conversation over the same mode name with MDS_HP_SEND in the remote node, the MDS_ROUTER signals MDS_HP_RECEIVE to terminate the conversation with DEALLOCATE (ABEND_PROG).

Session Outage:

```
 ___________________________________________________________________
|                                                                   |
|       MDS_HP_SEND                              MDS_HP_RECEIVE     |
|       _________________________________________________________|__
| 1     Allocate                                                    |
|         .                                                         |
|         .                                                         |
|         RC=OK                                                     |
| 2     Send_data (MDS-MU_1)                                        |
|         RC=OK                                                     |
| 3     Confirm _______________                                     |
|         .                     |                                   |
| 4       .                     |___________ÿ Receive_and_Wait      |
|         .                                    RC=OK                |
|         .                                    What=DATA   (MDS-MU_1)|
| 5       .                                  Receive_and_Wait       |
|         .                                    RC=OK                |
|         .                                    What=Confirm         |
| 6       .                    _______________ Confirmed            |
|         .                   |                RC=OK                |
| 7       .                   |              Receive_and_Wait       |
|         .                   |                  .                  |
| 8       RC=OK _______________|                 .                 |
| 9     Send_data (MDS-MU_2)                       .                 |
|         RC=OK                                    .                 |
| 10    Send_data (MDS-MU_3)                       .                 |
|         RC=OK                                    .                 |
| 11    Flush _______________                      .                 |
|         RC=OK               |                    .                 |
|                             |_ÿ º º º            .                 |
|                                                  .                 |
| 12                         *** Error ***         .                 |
|                           (session outage)       .                 |
|                                  | |             .                 |
| 13                               | |__________ÿ  RC=Resource_      |
|                                  |                Failure_retry    |
| 14                               |            Deallocate(local)    |
|                                  |              RC=OK              |
|                                  |                                 |
| 15    Send_data                  |                                 |
|         RC=Resource_ ___________|                                |
|           Failure_retry                                           |
|                                                                   |
|___________________________________________________________________|
```

Figure 9-75. Verb Sequence: Session Outage (Part 1 of 2)

Figure 9-75 illustrates another error case. The first MDS-MU was successfully sent and received. The second and third MDS-MUs are not successfully received because of a session outage. MDS_HP_SEND is notified of the failure when it attempts to send a fourth MDS-MU. It attempts to re-allocate the conversation and resend that MDS-MU. If the second attempt had also failed, MDS_HP_SEND would reported a send failure to the MDS_ROUTER. Note that the loss of the second and third MDS-MUs was not detected by either MDS_HP_SEND or the MDS_ROUTER. The MS application program using the transport must implement an error detection and recovery mechanism of its own. A simple technique, such as assigning consecutive sequence numbers to each record, can be used to detect a lost record.

1. A conversation is allocated.

2. The first MDS-MU is sent.

3. CONFIRM is issued.

4. The MDS-MU is received.

5. MDS_HP_RECEIVE goes into receive state, and the CONFIRM indication is received.

6. CONFIRMED is issued.

7. MDS_HP_RECEIVE goes into receive state.

8. CONFIRMED indication is received.

9. The second MDS-MU is sent.

10. The third MDS-MU is sent.

11. FLUSH is issued.

12. Session outage occurs. The MDS-MUs are lost.

13. MDS_HP_RECEIVE sees the failure as an error RC from RECEIVE_AND_WAIT.

14. MDS_HP_RECEIVE issues DEALLOCATE (LOCAL).

15. MDS_HP_SEND issues SEND_DATA to send a fourth MDS-MU. It sees the previous failure as the RC RESOURCE_FAILURE_RETRY.

    There is no indication that the previous sends were unsuccessful.

```
 ____________________________________________________________________
|                                                                    |
|        MDS_HP_SEND                                MDS_HP_RECEIVE   |
|        __________________________________________________________|__
| 16     Deallocate(local)                                           |
|          RC=OK                                                     |
| 17     Allocate                                                    |
|          .                                                         |
|          .                                                         |
|          RC=OK                                                     |
| 18     Send_data (MDS-MU_4)                                        |
|          RC=OK                                                     |
| 19     CONFIRM _________________                                   |
|          RC=OK                  |                                  |
| 20                              |______________ÿ Receive_and_Wait  |
|                                                  RC=OK             |
|                                                  What=DATA (MDS-MU_4) |
| 21                                             Receive_and_Wait    |
|                                                  .                 |
|                                                  .                 |
|                                                  .                 |
|                                                                    |
|____________________________________________________________________|
```

Figure 9-76. Verb Sequence: Session Outage (Part 2 of 2)

16. MDS_HP_SEND issues DEALLOCATE (LOCAL) to prepare for another allocation attempt.

17. Since the LU had indicated that retry was possible, MDS_HP_SEND attempts to allocate another conversation. MDS_HP_SEND will attempt to retry allocation three times (assuming that the LU indicates retry is possible) before reporting a failure back to the MDS_ROUTER.

18. The fourth MDS-MU is sent again.

19. CONFIRM is issued.

20. The MDS-MU is received.

21. MDS_HP_RECEIVE goes into receive state, ready to receive the CONFIRM indication.

Allocation Failure:

```
 ____________________________________________________________________
|                                                                    |
|        MDS_HP_SEND                                MDS_HP_RECEIVE   |
|        __________________________________________________________|__
| 1      Allocate (when_conwinner_allocated)                         |
|          .                                                         |
|          .                                                         |
|          RC=allocation_failure_retry                               |
| 2      Deallocate(local)                                           |
|          RC=OK                                                     |
| 3      Allocate (when_conwinner_allocated)                         |
|          .                                                         |
|          .                                                         |
|          RC=allocation_failure_retry                               |
| 4      Deallocate(local)                                           |
|          RC=OK                                                     |
| 5      Allocate (when_conwinner_allocated)                         |
|          .                                                         |
|          .                                                         |
|          RC=allocation_failure_retry                               |
```

```
| 6      Deallocate(local)                                          |
|          RC=OK                                                    |
| 7      (notify MDS router--                                       |
|         allocation failure)                                       |
|                                                                   |
|                                                                   |
|___________________________________________________________________|
```

Figure 9-77. Verb Sequence: MDS_HP_SEND Unable To Allocate Conversation

Figure 9-77 illustrates the retries which MDS_HP_SEND will attempt to allocate a conversation on a user mode session.

1. An attempt is made to allocate a conversation on a contention-winner session. The return code indicates that the ALLOCATE failed, but can be retried.

2. MDS_HP_SEND issues DEALLOCATE (LOCAL) to prepare for another allocation attempt.

3. An attempt is made to allocate a conversation on a contention-winner session again. The ALLOCATE fails.

4. MDS_HP_SEND issues DEALLOCATE (LOCAL) to prepare for another allocation attempt.

5. An attempt is made to allocate a conversation a third and final time. If this ALLOCATE had succeeded, MDS_HP_SEND would have gone on to send whatever data it had.

6. MDS_HP_SEND issues DEALLOCATE (LOCAL) to go to reset state.

7. The MDS_ROUTER is notified of the allocation failure.

   If MDS_HP_RECEIVE in this node is a conversation over the same mode name with MDS_HP_SEND in the remote node, the MDS_ROUTER signals MDS_HP_RECEIVE to terminate the conversation with DEALLOCATE (ABEND_PROG).

Error Sequences:

The following diagrams illustrate how error detection is coordinated between MDS_HP_SEND and MDS_HP_RECEIVE in a node. Since there are two active conversations between two nodes using MDS HPO (each node has one for sending and another for receiving), any failure on one of those conversations must cause the other to be ended as well. The error cases considered are the following:

° A network failure causes an outage on one of the two conversations, while the other remains active.

° A network failure causes an outage on both conversations.

° The failure of one TP causes its conversation to be terminated abnormally.

Normal Conversation Setup:

```
      ____________________ Node A ____________________          ____________________ Node B ____________________
     |MS       MDS_          MDS_HP_          MDS_HP_|        |MDS_HP_          MDS_HP_          MDS_      MS  |
     |Appl     Router        Receive TP       Send TP|        |Send TP          Receive TP       Router    Appl|
     |_____________________________________________|        |_____________________________________________|
       .         .              .               .                .               .               .         .
      Register_Appl_Name        .               .                .               .        Register_Appl_Name
 1     o________ÿo              .               .                .               .               o________o
       .  MDS-MU  .             .               .                .               .               .         .
 2     o________ÿo              .               .                .               .               .         .
       .         |            Start TP           .                .               .               .         .
 3     .         |_____________________________ÿo                .               .               .         .
       .         |            MDS-MU             .                .               .               .         .
 4     .         |_____________________________ÿo                .               .               .         .
       .         .              .           Allocate             .               .               .         .
       .         .              .           Send_Data            .               .               .         .
       .         .              .           Confirm              .               .               .         .
 5     .         .              .               |_____________________________ÿo                .         .
       .         .              .               .                .       Get_TP_Properties       .         .
       .         .              .               .                .       Receive_and_Wait        .         .
       .         .              .               .                .               |    MDS-MU     .         .
 6     .         .              .               .                .               |_____________ÿo          .
       .         .              .               .                .       Receive_and_Wait        |         .
       .         .              .               .                .           Confirmed           |         .
```

```
 7     .         .              .              o_________________________________|              |         .
       .         .              .              .                .               .              | MDS-MU  .
 8     .         .              .              .                .               .              |________ÿo
       .         .              .              .                .               .              . MDS-MU  |
 9     .         .              .              .                .               .             o________|
       .         .              .              .                .            Start TP          |         .
10     .         .              .              .                o_________________________________|         .
       .         .              .              .                .            MDS-MU            |         .
11     .         .              .              .                o_________________________________|         .
       .         .              .              .             Allocate            .              .         .
       .         .              .              .             Send_Data           .              .         .
       .         .              .              .             Confirm             .              .         .
12     .         .              o_________________________________|             .              .         .
       .         .       Get_TP_Properties     .                .              .              .         .
       .         .       Receive_and_Wait      .                .              .              .         .
       .         .     MDS-MU    |             .                .              .              .         .
13     .         o_______________|             .                .              .              .         .
       .         |       Receive_and_Wait      .                .              .              .         .
       .         |           Confirmed         .                .              .              .         .
14     .         |               |_________________________________ÿo          .              .         .
       .  MDS-MU |               .              .                .              .              .         .
15     o_________|              .              .                .              .              .         .
       .         .              .              .                .              .              .         .
```

Figure 9-78. Normal Sequence: Establishment of Parallel Conversations Between Two Nodes. This normal sequence sets up each of the error sequences which follow.

Figure 9-78 illustrates the normal establishment of the parallel conversations between two nodes using MDS HPO. This sequence is understood to precede each of the error sequence that follows.

1. The MS application programs in both node A and node B register with the MDS_ROUTER1

2. node A: The MS application program passes an MDS-MU to the MDS_ROUTER for transport to the corresponding MS application program in Node B.

3. node A: There is not an active copy of MDS_HP_SEND for this mode name to Node B, so the TP is started.

4. Node A: The MDS-MU is passed to MDS_HP_SEND.

5. Node A: MDS_HP_SEND issues:

   ALLOCATE
   SEND_DATA
   CONFIRM

   to cause the MDS-MU to be sent.

6. Node B: MDS_HP_RECEIVE is attached, and it issues:

   GET_TP_PROPERTIES
   RECEIVE_AND_WAIT

   to receive the MDS-MU, which is then passed to the MDS_ROUTER.

7. Node B: MDS_HP_RECEIVE issues:

   RECEIVE_AND_WAIT

   and receives an indication the CONFIRM was requested. It issues:

   CONFIRMED

8. Node B: The MDS-MU is passed to the destination MS application program.

9. Node B: The MS application program prepares an MDS-MU (perhaps a reply) for the original MS application program, and passes it to the MDS_ROUTER.

10. Node B: There is not an active copy of MDS_HP_SEND for this mode name to node A, so the TP is started.

11. Node B: The MDS-MU is passed to MDS_HP_SEND.

12. Node B: MDS_HP_SEND issues:

```
      ALLOCATE
      SEND_DATA
      CONFIRM
```

to cause the MDS-MU to be sent.

13. Node A: MDS_HP_RECEIVE is attached, and it issues:

```
      GET_TP_PROPERTIES
      RECEIVE_AND_WAIT
```

to receive the MDS-MU, which is then passed to the MDS_ROUTER.

14. Node A: MDS_HP_RECEIVE issues:

```
      RECEIVE_AND_WAIT
```

and receives an indication the CONFIRM was requested. It issues:

```
      CONFIRMED
```

15. Node A: The MDS-MU is passed to the destination MS application program.

Outage of One Conversation:

```
      ___________________ Node A ___________________          ___________________ Node B ___________________
     |MS       MDS_        MDS_HP_        MDS_HP_|         |MDS_HP_       MDS_HP_        MDS_      MS   |
     |Appl     Router      Receive TP     Send TP|         |Send TP       Receive TP     Router    Appl|
     |_____________________________________________|         |_____________________________________________|
      .          .            .            .                 .            .            .         .
      .  MDS-MU  .            .            .                 .            .            .         .
1     o_________ÿo            .            .                 .            .            .         .
      .         |          MDS-MU           .                 .            .            .         .
2     .         |______________________ÿo                    .            .            .         .
      .          .            .         Send_Data             .            .            .         .
      .          .            .           Flush               .            .            .         .
3     .          .            .             |____ÿ ° ° °      .            .            .         .
      .          .            .            .                  .            .            .         .
4     .          .            .            .   ***ERROR***    _____________ÿo           .         .
      .          .            .            .                  .     Receive_and_Wait     .         .
      .          .            .            .                  .       RC=Resource_Failure .         .
5     .          .            .            .                  .     Deallocate(local)    .         .
      .          .            .            .                  .            |             .         .
      .          .            .            .                  .            |  error report .         .
6     .          .            .            .                  .            |_____________ÿo         .
      .          .            .            .                  .     terminate conversation  |         .
7     .          .            .            .                 o_________________________________|         .
      .          .            .            .           Deallocate(abend)     .            .         .
8     .          .           o_____________________________|               .            .         .
      .          .      Receive_and_Wait   .                 .            .            .         .
      .          .        RC=Deallocate_abend .              .            .            .         .
9     .          .      Deallocate(local)  .                 .            .            .         .
      .          .            |            .                 .            .            .         .
      .          . error report  |         .                 .            .            .         .
10    .         o_____________|            .                .            .            .         .
      .         |  terminate conversation  .                .            .            .         .
11    .         |______________________ÿo                   .            .            .         .
      .          .            .       Deallocate(abend)      .            .            .         .
12    .          .            .                              .            .            .         .
      .          .            .            .                 .            .            .         .
```

Figure 9-79. Error Scenario: Failure Affecting only one of the Sessions. The error is reported to the MDS_HP_RECEIVE TP in Node B, which notifies its MDS_ROUTER, which subsequently causes the MDS_HP_SEND TP to DEALLOCATE its conversation. In Node A, the error report from MDS_HP_RECEIVE causes the same sequence of events to take place. Since MDS_HP_SEND in Node A had not attempted another SEND_DATA, it was not aware of the outage.

Figure 9-79 illustrates how the outage of one conversation is detected and the other conversation is accordingly shut down. Recovery is driven by the MDS_HP_SEND side of each conversation (not illustrated here). The normal conversation establishment show in Figure 9-78 is understood to have taken place before the sequence of events illustrated here.

1. Node A: The MS application program passes an MDS-MU to the MDS_ROUTER for transport to the corresponding MS application program in node B.

2. Node A: The MDS_HP_SEND TP is already active, so the MDS-MU is passed to it.

3. Node A: MDS_HP_SEND issues:

   SEND_DATA
   FLUSH

   to cause the MDS-MU to be sent.

4. A network outage has caused the failure of the underlying session being used for traffic from node A to node B. This outage occurs after MDS_HP_SEND has issued its verbs, so it is not notified.

5. Node B: MDS_HP_RECEIVE is in receive state (RECEIVE_AND_WAIT) was previously issued). It is driven by the LU and given a return code indicating that a resource failure has occurred. It issues DEALLOCATE(LOCAL).

6. Node B: MDS_HP_RECEIVE passes an error notification to the MDS_ROUTER.

7. Node B: MDS_ROUTER passes a TERMINATE_CONVERSATION to MDS_HP_SEND, which is still actively in a conversation with Node A. This is the case because the network outage did not affect the underlying session of that conversation (it may have traversed a different path from the other session).

8. Node B: MDS_HP_SEND issues DEALLOCATE(ABEND_PROG) to terminate the conversation.

9. Node A: MDS_HP_RECEIVE is in receive state, so it is passed a return code indicating that its conversation partner has ended the conversation abnormally. It issues DEALLOCATE(LOCAL) to reset its state.

10. Node A: MDS_HP_RECEIVE passes an error notification to the MDS_ROUTER.

11. Node A: MDS_ROUTER passes a TERMINATE_CONVERSATION to MDS_HP_SEND, which is still active, although its conversation has already been terminated. It is not aware of this state because it has not issued any verb by which the LU could notify it of the error.

12. Node A: MDS_HP_SEND issues DEALLOCATE(ABEND_PROG) to terminate the conversation and reset its state.

Outage of Both Conversations:

```
      _________________ Node A _________________          _________________ Node B _________________
     |MS      MDS_          MDS_HP_         MDS_HP_|     |MDS_HP_         MDS_HP_         MDS_     MS  |
     |Appl    Router        Receive TP      Send TP|     |Send TP         Receive TP      Router   Appl|
     |_______________________________________________|     |_______________________________________________|
       .        .              .               .              .              .              .        .
       .  MDS-MU .              .               .              .              .              .        .
1      o_________ÿo             .               .              .              .              .        .
       .         |            MDS-MU            .              .              .              .        .
2      .         |_______________________________ÿo             .              .              .        .
       .         .              .            Send_Data          .              .              .        .
       .         .              .              Flush            .              .              .        .
3      .         .              .               |____ÿ ° ° °    .              .              .        .
       .         .              .               .              .              .              .        .
4      .         .              .               .   ***ERROR***  .              .              .        .
       .         .              .               .      ||       .              .              .        .
       .         .              .               .      ||_______________________ÿo             .        .
       .         .              .               .      |        .      Receive_and_Wait        .        .
5      .         .              .               .      |        .        RC=Resource_Failure_Retry       .
       .         .              .               .      |        .      Deallocate(local)       .        .
       .         .              .               .      |        .              |               .        .
       .         .              .               .      |        .              |  error report .        .
6      .         .              .               .      |        .              |_______________ÿo       .
       .         .              .               .      |        .    terminate conversation     |        .
7      .         .              .               .      |        o_________________________________|        .
       .         .              .               .      |   Deallocate(abend)     .              .        .
8      .         .              .               .      |                       .              .        .
       .         .              o_______________________|        .              .              .        .
       .         .      Receive_and_Wait        .              .              .              .        .
9      .         .        RC=Resource_Failure_Retry             .              .              .        .
       .         .      Deallocate(local)       .              .              .              .        .
       .         .                |              .              .              .              .        .
       .         .  error report |               .              .              .              .        .
10     .         o_______________|               .              .              .              .        .
       .         |    terminate conversation     .              .              .              .        .
11     .         |_______________________________ÿo             .              .              .        .
```

```
        .         .              .       Deallocate(abend)        .              .              .        .
12      .         .              .                                .              .              .        .
        .         .              .              .                 .              .              .        .
```

Figure 9-80. Error Scenario: Failure Affecting both Contention-winner and Contention-loser Sessions. On both sides, the error is reported first to the MDS_HP_RECEIVE TPs, which notify the MDS_ROUTER, and subsequently cause the MDS_HP_SEND TPs to DEALLOCATE their conversations.

Figure 9-80 illustrates how the outage of both conversations is detected by the MDS_HP_RECEIVE TP in each node, and how the MDS_HP_SEND TP in each node is accordingly shut down. Recovery is driven by the MDS_HP_SEND side of each conversation (not illustrated here). The normal conversation establishment show in Figure 9-78 is understood to have taken place before the sequence of events illustrated here.

1. Node A: The MS application program passes an MDS-MU to the MDS_ROUTER for transport to the corresponding MS application program in Node B.

2. Node A: The MDS_HP_SEND TP is active, so the MDS-MU is passed to it.

3. Node A: MDS_HP_SEND issues:

   SEND_DATA
   FLUSH

   to cause the MDS-MU to be sent.

4. A network outage has caused the failure of the underlying sessions being used for traffic between Node A and Node B. This outage occurs after MDS_HP_SEND has issued its verbs, so it is not notified.

5. Node B: MDS_HP_RECEIVE is in receive state (RECEIVE_AND_WAIT) was previously issued). It is driven by the LU and given a return code indicating that a resource failure has occurred. It issues DEALLOCATE(LOCAL).

6. Node B: MDS_HP_RECEIVE passes an error notification to the MDS_ROUTER.

7. Node B: MDS_ROUTER passes a TERMINATE_CONVERSATION to MDS_HP_SEND, which is not aware of the outage.

8. Node B: MDS_HP_SEND issues DEALLOCATE(ABEND_PROG) to terminate the conversation.

9. Node A: MDS_HP_RECEIVE is in receive state, so it is passed a return code indicating that a resource failure has occurred. It issues DEALLOCATE(LOCAL) to reset its state.

10. Node A: MDS_HP_RECEIVE passes an error notification to the MDS_ROUTER.

11. Node A: MDS_ROUTER passes a TERMINATE_CONVERSATION to MDS_HP_SEND, which is still active, although its conversation has already been terminated. It is not aware of this state because it has not issued any verb by which the LU could notify it of the error.

12. Node A: MDS_HP_SEND issues DEALLOCATE(ABEND_PROG) to terminate the conversation and reset its state.

TP Failure:

```
                  _____________________ Node A _____________________          _____________________ Node B _____________________
                 |MS       MDS_           MDS_HP_          MDS_HP_|         |MDS_HP_          MDS_HP_          MDS_      MS   |
                 |Appl     Router         Receive TP       Send TP|         |Send TP          Receive TP       Router    Appl|
                 |_____________________________________________________|         |_____________________________________________________|
                   .         .              .                .                .               .               .         .
                   .  MDS-MU .              .                .                .               .               .         .
1                  o_________ÿo             .                .                .               .               .         .
                   .         |            MDS-MU             .                .               .               .         .
2                  .         |_____________________________ÿo                .               .               .         .
                   .         .              .            Send_Data            .               .               .         .
                   .         .              .              Flush              .               .               .         .
3                  .         .              .                |_______________________________ÿo               .         .
                   .         .              .                .                .       Receive_and_Wait        .         .
4                  .         .              .                .                .          ***ERROR***          .         .
                   .         .              .                .                .               |               .         .
5                  .         .              .                .                .       Deallocate(abend)       .         .
```

```
        .          .              .              .                .              |  error report .         .
6       .          .              .              .                .              |_______________ÿo        .
        .  MDS-MU  .              .              .                .              |               |         .
7       o__________ÿo             .              .                .              |               |         .
        .          |            MDS-MU            .                .              |               |         .
8       .          |_______________________________ÿo              .              |               |         .
        .          .              .           Send_Data            .              |               |         .
9       .          .              .            RC=Deallocate_ _____________________|               |        .
        .          .              .               Abend_Prog        .              .               |         .
10      .          .              .           Deallocate(local)     .              .               |         .
        .          . error report .               |                 .              .               |         .
11      .          o_____________________________|                .              .               |        .
        .          |  terminate conversation      .                 .              .               |         .
12      .          |______________ÿo              .                 .              .               |         .
        .          .        Deallocate(abend)     .                 .              .               |         .
13      .          .              |               .                 .              .               |         .
        .          .              |               .                 .       terminate conversation  |         .
14      .          .              |               .                o_______________________________|        .
        .          .              |               .                |              .                .         .
15      .          .              |___________________________ÿDeallocate(abend)   .                .         .
        .          .              .               .                              .               .        .
        .          .              .               .                .              .                .         .
```

Figure 9-81. Error Scenario: Failure of MDS_HP_RECEIVE TP. The MDS_HP_RECEIVE TP is unable to successfully receive an MDS-MU (perhaps because of storage constraints), so it DEALLOCATES its conversation. It notifies it MDS_ROUTER about the error. Meanwhile, MDS_HP_SEND in Node A attempts to send another MDS-MU. This fails, because the LU is aware of the DEALLOCATE(ABEND), and the MDS_ROUTER in Node A instructs MDS_HP_RECEIVE to terminate its conversation.

Figure 9-79 illustrates how the failure of one TP is detected to shut the other conversation down. Recovery is driven by the MDS_HP_SEND side of each conversation (not illustrated here). The normal conversation establishment show in Figure 9-78 is understood to have taken place before the sequence of events illustrated here.

1. Node A: The MS application program passes an MDS-MU to the MDS_ROUTER for transport to the corresponding MS application program in node B.

2. Node A: The MDS_HP_SEND TP is active, so the MDS-MU is passed to it.

3. Node A: MDS_HP_SEND issues:

   SEND_DATA
   FLUSH

   to cause the MDS-MU to be sent.

4. Node B: MDS_HP_RECEIVE is in receive state (RECEIVE_AND_WAIT) was previously issued). It is driven by the LU and receives the first part of the MDS-MU, but an internal error (insufficient storage), causes it to fail.

5. Node B: MDS_HP_RECEIVE issues DEALLOCATE(ABEND_PROG).

6. Node B: MDS_HP_RECEIVE passes an error notification to the MDS_ROUTER.

7. Node A: The MS application program passes another MDS-MU to the MDS_ROUTER for transport to the MS application program in Node B.

8. Node A: The MDS-MU is passed to MDS_HP_SEND

9. Node A: MDS_HP_SEND issues:

   SEND_DATA

   and receives a return code indicating that the conversation has been abnormally terminated by MDS_HP_RECEIVE.

10. Node A: MDS_HP_SEND issues DEALLOCATE(LOCAL).

11. Node A: MDS_HP_SEND passes an error notification to the MDS_ROUTER.

12. Node A: MDS_ROUTER passes a TERMINATE_CONVERSATION to MDS_HP_RECEIVE, which, of course, is not aware that MDS_HP_RECEIVE in the remote node has failed.

13. Node A: MDS_HP_RECEIVE issues DEALLOCATE(ABEND_PROG) to terminate the conversation and reset its state.

14. Node B: Meanwhile, MDS_ROUTER passes a TERMINATE_CONVERSATION to MDS_HP_SEND, which is still actively in a conversation with node A.

15. Node B: MDS_HP_SEND issues DEALLOCATE(ABEND_PROG) to terminate the conversation, which has actually already been terminated by node A.

MDS_HP_SEND Processing:

Process Name	MDS_HP_SEND TP
Description	This transaction program sends MDS-MUs over the sessions indicated by the MDS_ROUTER
Input	"MDS Internal Signal - MDS Routing Unit" in topic 2.9.2.4.4
Output	° MDS-MU sent over session indicated in MDS Routing Unit. ° MDS Routing Unit updated with sense data if status is ERROR. ° MDS Routing Unit updated with session status information.

This transaction program sends outbound messages as directed by the MDS_ROUTER over user mode sessions. It will allocate a conversation with MDS_HP_RECEIVE in the destination node over the indicated mode name, and then maintain that conversation until an error occurs or the MDS_ROUTER indicates that the conversation is to be terminated.

The processing of this TP is illustrated in Figure 9-82 and succeeding charts. The description of the processing in each chart follows it.

Implementation Note

The LU 6.2 verb syntax and sequences described in this section are based upon *Systems Network Architecture Transaction Programmer's Reference Manual for LU Type 6.2*. However, IBM product implementations of LU 6.2 may provide APPC programming interfaces that differ in some details from the architecturally defined verb interface. In particular, as an optimization, IBM products sometimes allow combinations of verbs to be issued in a single invocation of the programming interface. These variations are not apparent to LU 6.2 TPs in other nodes, since equivalent network flows are created.

Implementers of MULTIPLE_DOMAIN_SUPPORT should consult the appropriate documentation for the LU 6.2 product that they will be using in order to determine which product-specific programming sequences are equivalent to the verb sequences that appear in this book.

MDS_HP_SEND ALLOCATE Processing

```
|            |                  |     exhausted?     | no
|            |                  |____________________|
|            |                           |yes
|            |                                        °°°°°°°°°°°°°°°°°°°°°°
|            |                                        °                    °
|            |_________________ÿ|       issue        |  °     MDS_ROUTER     °
|                               | DEALLOCATE(LOCAL)  |  °                    °
|____________                   |____________________|  °°°°°°°°°°°°°°°°°°°°°°
             |                           |                        "
                                                               |
 ____________________        ____________________        ____________________
|       issue        |       | Update MDS Routing |       |  Send MDS Routing  |
|   GET_ATTRIBUTES   |    __ÿ|Unit w/status ERROR,|______ÿ| Unit to MDS_ROUTER |
|                    |   |   | sense X'08A8 0012' |       |                    |
|____________________|   |   |____________________|       |____________________|
             |             |
                          |                                          |
                            |                                        |
 ____________________       |   ____________________                 |
| Update MDS Routing |      |   | Take next MDS      |                |
|Unit w/session info.|      |___| Routing Unit from  |________________|   ___
|                    |          | input queue        |____________________| 4 |
|____________________|          |____________________|                    |___|
             |                           |queue empty

                                       (exit)
              ___
            | 2 |
            |___|
```

Figure 9-82. MDS_HP_SEND Processing (Part 1 of 4)

The ALLOCATE processing is invoked when the MDS_ROUTER starts an instance of MDS_HP_SEND and passes an MDS Routing Unit (OUTBOUND) to it.

1. ALLOCATE for a contention-winner session is issued: (2)

```
ALLOCATE LU_NAME(<session-partner>)
         MODE_NAME(<session mode name>)
         TPN(MDS_HP_RECEIVE)
         TYPE(BASIC_CONVERSATION)
         RETURN_CONTROL(WHEN_CONWINNER_ALLOCATED)
         SYNC_LEVEL(CONFIRM)
```

2. The ALLOCATE return code is examined to determine the next step:

   a. OK

      1) GET_ATTRIBUTES is issued.

      2) The MDS Routing Unit is updated with the session CGID information.

      3) Processing continues at connector box 2, labeled successful ALLOCATION.

   b. ALLOCATION_ERROR.ALLOCATION_FAILURE_RETRY

      1) The LU is indicating that the contention-winner session could not be activated, but that an allocation retry may be successful.

      2) DEALLOCATE TYPE(LOCAL) is issued to reset the state of the TP.

      3) The same ALLOCATE verb is issued up to two more times (continuing at step 1).

   c. Any other return code:

      1) DEALLOCATE TYPE(LOCAL) is issued to reset the state of the TP.

      2) No further attempt is made to ALLOCATE Processing continues at step 3.

3. If the ALLOCATE ended with any other return code:

   a. DEALLOCATE TYPE(LOCAL) is issued to reset the state of the TP.

   b. The MDS Routing Unit is updated:

- Routing Status is set to ERROR.
- Sense data is set to X'08A8 0012' (irrecoverable failure of user-mode session)
- Direction is set to REPLY TO OUTBOUND
- MDS error message required is set to ORIGIN

c. The MDS Routing Unit is sent to MDS_ROUTER.

d. Any additional MDS Routing Units in the TP input queue are also updated in the same manner and sent back to MDS_ROUTER.

e. The transaction program ends.

Send Processing

Figure 9-83. MDS_HP_SEND Processing (Part 2 of 4)

1. SEND_DATA DATA(<MDS-MU>) is issued.

2. If the primary RC is *not* OK, processing continues at connector box 3.

3. If this was the first SEND_DATA issued after the successful completion of ALLOCATE, then

   CONFIRM

   is issued. Confirmation is required for the first MDS-MU sent on the conversation so that MDS_HP_SEND can verify that the correct level of support exists in the remote node. For example, if the MDS_HP_RECEIVE TP does not exist in the remote node, the ALLOCATE verb will be successful, but the actual attachment of the TP (which occurs after

the first FLUSH or CONFIRM) will fail.

4. If the primary RC is *not* OK, processing continues at connector box 3.

5. If the primary RC is OK, the MDS Routing Unit is updated:

   - Direction REPLY TO OUTBOUND

   - Session information

   - Routing status COMPLETE

6. The MDS Routing Unit is returned to the MDS_ROUTER.

7. If the process input queue is empty (no additional MDS-MUs to be sent), the LU must be instructed to send any MDS-MUs that it has buffered. The TP issues

   FLUSH

   If no data is buffered, this verb simply returns an OK return code without sending anything to the remote node.

8. If the primary RC is *not* OK, processing continues at connector box 3.

9. A receive is issued to the process input queue. The process will be posted when the next MDS Routing Unit is placed in the queue by the MDS_ROUTER.

Figure 9-84. MDS_HP_SEND Processing (part 3 of 4)

The MDS Routing Unit received from the MDS_ROUTER is examined:

1. If the Routing status is not one of:

   TERMINATE_CONVERSATION_NORMALY

TERMINATE_CONVERSATION_ABNORMALY

then it contains an MDS-MU to be sent. Processing continues at 1.

2. If the Routing status is

TERMINATE_CONVERSATION_ABNORMALLY

an error has been detected by MDS_HP_RECEIVE in this node, and the MDS_ROUTER is instructing MDS_HP_SEND to terminate its communication, as well. The following is done:

a. DEALLOCATE(ABEND_PROG) is issued

b. If there are any additional MDS Routing Units remaining in the input queue, each is updated with sense data X'08A8 0012' (failure of user-mode session) and is returned to the MDS_ROUTER, which will send MDS error messages to the MS application programs.

c. This process exits.

3. If the Routing status is

TERMINATE_CONVERSATION_NORMALLY

The MDS_ROUTER is instructing MDS_HP_SEND to shut down this conversation after any enqueued MDS Routing Units have been processed. The following is done:

a. If the input queue from the MDS_ROUTER contains any MDS Routing Units in addition to the TERMINATE_CONVERSATION_NORMALLY signal, they must be processed normally before shutting the conversation down. The TERMINATE_CONVERSATION_NORMALLY MDS Routing Unit is moved to the end of the input queue (after the other MDS Routing Units), and processing resumes at 9.

b. If the input queue does not contain any additional MDS Routing Units, the conversation is shut down:

1) DEALLOCATE(CONFIRM) is issued

2) If RC is OK, this process exits.

3) If RC is not OK, the MDS Routing Unit is updated:

- Direction REPLY TO OUTBOUND
- Status ERROR
- Sense data X'08A8 0012' (failure of user-mode session)
- MDS error message required ORIGIN

This process exits.

```
                 ___
                | 3 | (Primary RC was not OK)
                |___|
                  |

          _________________
         |      issue      |
         | DEALLOCATE(LOCAL)|
         |_________________|
                  |
 _________________|
|
|
|    ___________________________
|   |when RC=RESOURCE_FAILURE_RETRY|                                  ___
|___ÿ|     or ALLOCATION_ERROR.    |_________________________________ÿ| 1 |
|    |       TP_NOT_AVAILABLE_RETRY |                                  |___|
|    |_____________________________|
|
|    ___________________________   _________________
|   |  when RC=ALLOCATION_ERROR.  |  |Update MDS Routing |
|___ÿ|          TPN_NOT_RECOGNIZED |_____ÿ|Unit w/status ERROR,|_____
|    |                             |  |sense X'08A8 0010', |     |
|    |_____________________________|  |Dir. REPLY TO OUTB. |     |
|                                     |___________________|     |
|                                                               |
|                                      _________________        |
|    ___________________________      |Update MDS Routing |     |
|___ÿ|          otherwise          |_____ÿ|Unit w/status ERROR,|     |
     |_____________________________|  "  |sense X'08A8 000B', |     |
                                       |  |Dir. REPLY TO OUTB. |     |
```

```
|   |____________________|     |
|            |_______________|
|
|   ____________________          °°°°°°°°°°°°°°°°°°°°°°
|   |  Send MDS Routing  |        °                    °
|   | Unit to MDS_ROUTER |___ÿ°     MDS_ROUTER         °
|   |____________________|        °                    °
|            |                    °°°°°°°°°°°°°°°°°°°°°°
|
|   ____________________
|   |  Additional MDS    |
|___| Routing Units in   |
yes |   input queue?     |
    |____________________|
             | no

          (exit)
```

Figure 9-85. MDS_HP_SEND Processing (Part 4 of 4)

1. DEALLOCATE TYPE(LOCAL) is issued to reset the TP state.

2. When RC = ALLOCATION_ERROR.TPN_NOT_RECOGNIZED, the remote node does not support the MDS high-performance option.

   The MDS Routing Unit is updated with these values:

   - Direction REPLY TO OUTBOUND
   - Status ERROR
   - Sense data X'08A8 0010' (MDS HPO not supported by remote node)
   - MDS error message required ORIGIN

3. For any other return code, the MDS Routing Unit is updated with these values:

   - Direction REPLY TO OUTBOUND
   - Status ERROR
   - Sense data X'08A8 000B' (TP failure in remote node)
   - MDS error message required ORIGIN

   Note: Sense data X'08A8 000B' applies only to a single MDS-MU - it is not the equivalent of session-outage notification. An irrecoverable TP failure in a remote node may be a transient error and subsequent communication may succeed. The MS application program receiving an MDS error message with that sense data will pursue application program-level recovery, such as switching to a backup focal point, but other application program-level traffic should not be impacted by this error.

4. The MDS Routing Unit is sent to the MDS_ROUTER.

5. If additional MDS Routing Units are in the TP input queue, they are also updated with the same values and sent to the MDS_ROUTER.

MDS_HP_RECEIVE Processing:

Process Name	MDS_HP_RECEIVE TP
Description	This transaction program receives MDS-MUs sent by remote instances of the MDS_HP_SEND TP. The MDS-MUs are passed to the MDS_ROUTER.
Input	MDS-MU received from remote node.
Output	- MDS Routing Unit passed to MDS_ROUTER.

The MDS_HP_RECEIVE transaction program is started by an Attach coming into the node from a remote instance of the MDS_HP_SEND transaction program. It is illustrated in Figure 9-86.

```
                    (TP started by incoming ATTACH)
                                 |

                      ___________________
                     |       issue       |
                     |  GET_ATTRIBUTES   |
                     |___________________|   ___
        ____________________________ÿ|_____________| 1 |_________
       |                                      |___|          |
       |                      ___________________             |
       |                     |       issue       |            |
       |                     |  RECEIVE_AND_WAIT |            |
```

```
          |                 |     FILL(LL)       |                 |
          |                 |____________________|                 |
          |                          |                             |
          |                                                        |
          |                  ___________________   ___             |
          |                 | Is return code OK? |____ÿ| 2 |       |
          |                 |____________________| no  |___|       |
          |                          |yes                          |
          |                                                        |
 ___________________       ___________________                     |
| issue CONFIRMED   |_____|                   |_________________|
|___________________|CONFIRM|   WHAT_RECEIVED?   | DATA_INCOMPLETE
                            |                   |
                            |___________________|
 ___________________ CONFIRM_DEALLOC | |  | other          ___________________
| issue CONFIRMED   |_______________| |  |_______________ÿ| issue DEALLOCATE  |
|___________________|                 |                  |    (ABEND_PROG)    |
         |                          |DATA_COMPLETE      |____________________|
                                                                 |
      (exit)
                            ___________________
                            |     Build MDS     |        ___________________
                            |    Routing Unit   |       |Create MDS Routing  |
                            |___________________|       |Unit w/status ERROR,|
                                     |                  |sense X'08A8 0012', |
°°°°°°°°°°°°°°°°°°°°°°                                  |Dir. INBOUND        |
°                    °      ___________________         |____________________|
°     MDS_ROUTER     °______| send to MDS_ROUTER |                |
°                    °      |___________________|                                °°°°°°°°°°°°°°°°°°°°°°°°
°°°°°°°°°°°°°°°°°°°°°°               |                                            °                      °
                                                        ___________________       °     MDS_ROUTER       °
 ___________________       ___________________         | send to MDS_ROUTER |___ÿ°                      °
| issue DEALLOCATE  |_____|    Did send to    |         |____________________|    °°°°°°°°°°°°°°°°°°°°°°°°
| (ABEND_PROG)      |  no |  MDS_ROUTER work? |                |
|___________________|     |___________________|               (exit)
          |                          | yes

        (exit)
                            ___________________
                   ___      |  Check queue from |
                  | 1 |_____| MDS_ROUTER: signal |
                  |___|  no |    to terminate   |
                            |   conversation?   |
                            |___________________|
                                     | yes

                            ___________________
                            |  issue DEALLOCATE |
                            |    (ABEND_PROG)   |
                            |___________________|
                                     |

                                  (exit)
```

Figure 9-86. MDS_HP_RECEIVE Processing (Part 1 of 2)

It performs the following processing:

1. GET_ATTRIBUTES is issued to get

   - Network-qualified LU name for the session partner
   - Mode name
   - CGID

   This information is required in the MDS Routing Unit which will be built.

2. RECEIVE_AND_WAIT FILL(LL) is issued.

3. The return code is examined:

   If return code is not OK, processing continues at connector box 2, labeled "Primary RC from RECEIVE_AND_WAIT was not OK."

   If the return is OK:

   a. If WHAT_RECEIVED is DATA_INCOMPLETE, processing continues at step 2.

   b. If WHAT_RECEIVED is DATA_COMPLETE:

      1) The MDS Routing Unit is prepared:

         - Direction INBOUND
         - Network-qualified LU name
         - Mode name
         - Session CGID
         - MDS support level CURRENT
         - Data received (MDS-MU)
         - Routing status INCOMPLETE

2) The completed MDS Routing Unit is sent to the MDS_ROUTER.

3) If that interprocess send fails for any reason, the TP issues:

DEALLOCATE(ABEND_PROG)

and terminates. Since the TP is unable to communicate with the MDS_ROUTER, there is no way for it to report this failure.

4) The TP checks its input queue from the MDS_ROUTER for an MDS Routing Unit with the status TERMINATE_CONVERSATION. If that signal is there, the TP issues:

DEALLOCATE(ABEND_PROG)

and terminates.

5) Processing continues at step 2.

c. If WHAT_RECEIVED is CONFIRM:

1) CONFIRMED is issued.

2) Processing continues at step 2.

d. If WHAT_RECEIVED is CONFIRM_DEALLOCATE:

1) CONFIRMED is issued.

2) This process terminates.

e. If WHAT_RECEIVED is any other value:

1) The TP issues DEALLOCATE(ABEND_PROG).

2) An MDS Routing Unit is prepared with:

- Direction INBOUND
- Network-qualified LU name
- Mode name
- Sense data X'08A8 0012' (Unrecoverable failure of user mode session)
- Routing status ERROR

3) The MDS Routing Unit is passed to the MDS_ROUTER.

4) This process terminates.

```
                  ___
                 | 2 | (Primary RC from RECEIVE_AND_WAIT was not OK)
                 |___|
                   |
 __________________|
|
|     ________________________________
|____ÿ| when ALLOCATION_ERROR          |
|     |   or DEALLOCATE_NORMAL         |
|     |   or DEALLOCATE_ABEND_PROG     |          ___________________
|     |   or DEALLOCATE_ABEND_SVC      |______ÿ|  issue DEALLOCATE  |___ÿ(exit)
|     |   or DEALLOCATE_ABEND_TIMER    |       |      (LOCAL)       |
|     |   or RESOURCE_FAILURE_RETRY    |       |____________________|
|     |   or RESOURCE_FAILURE_NO_RETRY |
|     |________________________________|
|
|
|     ________________________________          ___________________
|____ÿ|    otherwise (any other RC)    |______ÿ|  issue DEALLOCATE  |___ÿ(exit)
      |                                |       |   (ABEND_PROG)     |
      |________________________________|       |____________________|
```

Figure 9-87. MDS_HP_RECEIVE Processing (Part 2 of 2)

1. If the return code is one of the following,

ALLOCATION_ERROR

```
          DEALLOCATE_NORMAL
          DEALLOCATE_ABEND_PROG
          DEALLOCATE_ABEND_SVC
          DEALLOCATE_ABEND_TIMER
          RESOURCE_FAILURE_RETRY
          RESOURCE_FAILURE_NO_RETRY

       the conversation remains in DEALLOCATE state, and DEALLOCATE
       TYPE(LOCAL) must be issued to reset it before ending the TP.

   2.  If the return code is any other value, DEALLOCATE TYPE(ABEND_PROG) is
       issued to unconditionally end the conversation.
```

---

(2) The LU 6.2 verb examples shown illustrate the key options required for this design. They may not show all required parameters.

| 2.9.2.8 MULTIPLE_DOMAIN_SUPPORT Optional Subset 4 (Transport Confirmation Option)

```
   This optional subset provides extensions to MULTIPLE_DOMAIN_SUPPORT
   allowing MS application programs to specify whether or not session-level
   confirmation is to be performed by the MDS transaction programs.
   "MULTIPLE_DOMAIN_SUPPORT Base Subset" in topic 2.9.2.4 uses the LU 6.2
   CONFIRM/CONFIRMED verbs to verify that each MDS-MU is successfully
   transported between nodes.  This is the default function performed by
   every implementation of MULTIPLE_DOMAIN_SUPPORT.  "MULTIPLE_DOMAIN_SUPPORT
   Optional Subset 3 (High-Performance Option)" in topic 2.9.2.7, on the
   other hand, eliminates session-level confirmation for MDS-MUs in order to
   achieve higher levels of throughput over user-mode sessions.  This
   optional subset allows the MS application program to override the normal
   use of confirmation in either case:

   °   To omit session-level confirmation by base MDS (normally every MDS-MU
       is confirmed).

   °   To request session-level confirmation by MDS high-performance option
       TPs (normally MDS-MUs are not confirmed).

   Note that for base MDS this option applies only to the first session hop
   for default routing in an APPN network.

   This optional subset affects only confirmation at the session level
   between the MDS service transaction programs.  MULTIPLE_DOMAIN_SUPPORT
   does not provide for application program-level confirmation of MDS-MU
   delivery except through the use of application program replies.
```

Subtopics:

- 2.9.2.8.1 Function Provided
- 2.9.2.8.2 Implementation Requirements

2.9.2.8.1 Function Provided

```
 | Optional Subset 4 (Transport Confirmation Option) allows MS application
   programs to override the normal use of session-level confirmation by the
   MDS service transaction programs:

   °   MDS_SEND for MULTIPLE_DOMAIN_SUPPORT Base Subset

   °   MDS_HP_SEND for MULTIPLE_DOMAIN_SUPPORT Optional Subset 3
       (High-Performance Option).
```

2.9.2.8.2 Implementation Requirements

```
   This optional subset is described as a set of changes to MDS as described
   in "MULTIPLE_DOMAIN_SUPPORT Base Subset" in topic 2.9.2.4:  consisting of
   a subset of CP management services including:

   °   Modifications to the following protocol boundary:
```

- "Protocol Boundary R--Send MDS-MU" in topic 2.9.1.13

```
 ___ New field in Data Content ___________________________________
|                                                                  |
| 3.  Session-level confirmation indicator for MDS service TPs     |
|                                                                  |
|     °   Default                                                  |
|                                                                  |
|         For base MDS, confirmation will be used; for             |
|         MULTIPLE_DOMAIN_SUPPORT Optional Subset 1                |
|         (High-Performance Option), confirmation will not be      |
|         used.                                                    |
|                                                                  |
|     °   Do not confirm MDS-MU (overrides default)                |
|                                                                  |
|     °   Confirm MDS-MU (overrides default)                       |
|                                                                  |
|__________________________________________________________________|
```

- "MDS Internal Signal - MDS Routing Unit" in topic 2.9.2.4.4

```
 ___ New field in Data Content ___________________________________
|                                                                  |
| 15. Session-level confirmation indicator for MDS service TPs     |
|                                                                  |
|     °   Default                                                  |
|                                                                  |
|         For base MDS, confirmation will be used; for             |
|         MULTIPLE_DOMAIN_SUPPORT Optional Subset 1                |
|         (High-Performance Option), confirmation will not be      |
|         used.                                                    |
|                                                                  |
|     °   Do not confirm MDS-MU (overrides default)                |
|                                                                  |
|     °   Confirm MDS-MU (overrides default)                       |
|                                                                  |
|__________________________________________________________________|
```

Modifications to MDS_SEND:

This TP is modified to issue FLUSH rather than CONFIRM when the MDS Routing Unit contains the "Do not confirm MDS-MU" indication.

Modifications to MDS_HP_SEND:

This TP is modified to issue CONFIRM when the MDS Routing Unit contains the "Confirm MDS-MU" indication.

2.9.2.9 Functions Common to MS Application Programs Using MDS

Subtopics:

- 2.9.2.9.1 Registering with MDS
- 2.9.2.9.2 Common MDS-MU Parsing
- 2.9.2.9.3 Rejecting an MDS-MU with an Unsupported Application GDS Variable
- 2.9.2.9.4 Building Negative Replies for MDS-MUs
- 2.9.2.9.5 Canceling an Outstanding Unit of Work with an MDS Error Message

---

2.9.2.9.1 Registering with MDS

Every MS application program that sends or receives MDS-MUs must register itself with MDS when it first becomes active. This enables MDS to route MDS requests, replies, and error messages to the application program when they are received. Even an application program, such as EP_ALERT, that only sends, but never receives, MDS requests must register itself with MDS, because MDS error messages must still be routed to such an application program by MDS.

An application program registers itself with MDS via internal MS protocol boundary O (Register Application program Name). The details of this protocol boundary are described in "Protocol Boundary O--Register Application Program Name" in topic 2.9.1.11. The application program registering itself passes its name to MDS and the ACTION indicator of ADD via this protocol boundary. In most cases the name may be either the 4-byte architecturally-defined one for the application program, e.g., X'23F0F3F0' (EP_ALERT), or a one-to eight-character installation-defined

name. One application program, however, *must* register under its architecturally-defined name: MS_CAPS (X'23F0F1F1').

Whenever it receives a registration request via protocol boundary O, MDS always accepts it, entering the application program name specified on the request into its application program routing table.

Implementation Note

The architecture model does not discover MS application program failures except as interprocess communication failures. If an MS application program process fails and is reinitialized, the MDS_ROUTER will accept reregistration of the application program name.

Implementations may manage this interaction between the MDS_ROUTER and the MS application programs differently.

When the MS application program no longer needs to send or receive MDS-MUs, it may invoke protocol boundary O with the ACTION indicator set to DELETE. When that indication is used, MDS will remove an existing entry from the routing list for the application program name.

See "Processing Register Application Program Name" in topic 2.9.2.4.8 for details of MDS's processing of a registration request it receives via protocol boundary O. Appendix D, "MS Transaction Program Names" in topic D.0 lists all of the MS application programs that use the MDS transport, and thus register themselves with MDS.

2.9.2.9.2 Common MDS-MU Parsing

Management services application programs using the MDS transport must parse the MDS-MUs they receive. The initial steps of this parsing are the same, regardless of the application program involved. Subsequent steps vary, depending on the particular management services major vectors present in the MDS-MU.

Some application programs, for example, EP_COMMON_OPERATIONS_SERVICES, can receive the same management services command in either of two forms: as an NMVT on the SSCP-PU session, or as an MDS-MU on a CP-CP session. In order to simplify the task of such an application program, the parsing involved in processing requests is divided into parts:

- **Common NMVT Parsing:** These are the parsing steps common to all NMVT requests, regardless of the major vector involved. These steps are documented in "Parsing of NMVTs" in topic 2.10.7.7.

- **Common MDS-MU Parsing:** These are the parsing steps common to all MDS-MUs. They are documented below, in Table 9-16.

- **Major-Vector-Unique Parsing:** A unique set of parsing steps is documented for each *request* major vector. Parsing of data major vectors is, in general, not described in detail, since there is ordinarily no recovery action that an EP_XXXX application program can take if it finds out that it has sent a faulty major vector.

  Major-vector-unique parsing starts at the point where both of the common parsing sequences leave off: after the major vector key has been identified as one that the receiver supports. Thus it is easy for an application program such as EP_COMMON_OPERATIONS_SERVICES to start with either of the common parsing sequences, depending on the form in which a request reached it, then switch over to the major-vector-unique parsing.

  Parsing sequences for each of the management services request major vectors appear in "EP_XXXX Parsing of Individual Management Services Major Vectors" in topic 2.10.8.

Table 9-16. Sense Data Returned by an MS Application After Receipt of an MDS-MU

Order of Checking	Sense Data	Condition	Notes
1	X'080C 0012'	Application GDS variable in an MDS-MU not recognized	1
2	X'080C 0005'	Major vector key not supported	2

Note:

```
| 1.  See "Rejecting an MDS-MU with an Unsupported Application GDS        |
|     Variable" in topic 2.9.2.9.3 for a description of how this error    |
|     is reported.                                                        |
|                                                                         |
| 2.  This sense data is placed in the Sense Data (X'7D') subvector of    |
|     the SNA Condition Report (X'1532') GDS variable.  The SNA           |
|     Condition Report is enveloped in a CP-MSU (X'1212') GDS variable,   |
|     which is sent as the application GDS variable in an MDS reply       |
|     directed to the originator of the request.                          |
|_________________________________________________________________________|
```

2.9.2.9.3 Rejecting an MDS-MU with an Unsupported Application GDS Variable

MDS provides a general mechanism for transferring data between management services application programs. One aspect of this generality is that for MDS requests and replies, MDS does not in any way limit the application program GDS variables that the MDS-MU contains. (3) Thus an application program that uses the MDS transport must be prepared to respond negatively to an MDS-MU containing an application program GDS variable that it does not support.

Ordinarily, when an application program responds negatively to a request, the reply is an MDS-MU whose application program GDS variable is the one used by the application program and its partners. If, for example, a management services application program rejects a command because the resource identified in the command is unknown to it, or because it does not currently have sufficient resources to execute the command, it replies with an MDS-MU whose application program GDS variable is a CP-MSU. The specific exception code is contained in a structure inside the CP-MSU.

If, however, the application program GDS variable itself is not supported by the receiver, then the reply must be handled differently. To ensure that the two application programs have some common ground upon which to communicate, the receiver responds not with an MDS reply, but with an MDS error message. The application program GDS variable is thus X'1532' (SNA Condition Report), the GDS variable that all application programs using the MDS transport must recognize.

The MDS-MU that an application program sends when it receives an MDS-MU with an unsupported application program GDS variable is built as follows:

- Origin Location:

  - NETID and NAU name: The values for the application program's NAU

  - Application program name: The name under which the application program registered itself with MDS.

- Destination Location: The NETID, NAU name, and application program name from the Origin Location field of the MDS-MU being rejected.

- Flags:

  - MDS message type: MDS error message
  - First MDS message indicator: 1
  - Last MDS message indicator: 1

- Agent UOW Correlator: The value from the MDS-MU being rejected.

- Application program GDS variable: an SNA Condition Report (X'1532') GDS variable containing one structure: the SNA Report Code (X'7D') subvector, containing the SNA sense data X'080C 0012' (application program GDS variable in MDS-MU not supported). The format of the SNA Condition Report appears in *SNA Formats*.

---

(3) Since an MDS error message must be processed by MDS itself, as well as by the application program to which it is delivered, MDS requires that the application program GDS variable in an MDS error message be X'1532' (SNA Condition Report).

2.9.2.9.4 Building Negative Replies for MDS-MUs

"Rejecting an MDS-MU with an Unsupported Application GDS Variable" in topic 2.9.2.9.3 describes a special case, in which a receiver of an MDS-MU does not support the application program GDS variable. The majority of MDS-MU parsing errors involve some element further down in the hierarchy

of elements that make up an MDS-MU: the major vector key (recall that this is the subject of the second step of the *common* MDS-MU parsing), a subvector or subfield key, or a value contained in a subvector or subfield.

Table 9-17 shows the three formats for reporting MDS-MU errors. All three contain a X'7D' subvector with an SNA sense data to identify the precise error that was detected. They differ, however, in the structures surrounding the X'7D' subvector. Each of the following sections indicates whether Format 2 or Format 3 is used for error reporting if the request is received in a MDS-MU.

It is important to note that of these three formats for reporting errors via an MDS-MU, only Format 1 has an MDS message type of "MDS error message" (as identified in the Flags (X'90') subvector in the MDS Routing Information (X'1311') GDS variable). Formats 2 and 3 have MDS message types of either MDS request or MDS reply. The SNA Condition Report (X'1532') GDS variable is a generic encoding which is used in the MDS error message as well as in other contexts. The presence of the SNA Condition Report (X'1532') GDS variable in an MDS-MU does *not* mean that the MDS-MU is an MDS error message.

An MDS error message is sent by an MS application program only when it is unable to communicate at even the most basic level (recognition of a received GDS variable variable) with another MS application program. If a received GDS variable is recognized, then any error reporting is within the context of that GDS variable (for example, by an SNA Condition Report (X'1532') GDS variable enveloped within a CP-MSU or a Sense Data (X'7D') subvector within a major vector.

Table 9-17. Error Report Formats used by MS Application Programs

Format 1 -- MDS Error Message	
|___MDS header___ÿ|_application_ÿ| GDS variable | HHHHHHHHHHHHHHHHH | SNACR (X'7D') |	The X'7D' subvector is in an SNA Condition Report (X'1532') GDS variable that occupies the application program GDS variable slot within the MDS-MU. This format, described in greater detail in "Rejecting an MDS-MU with an Unsupported Application GDS Variable" in topic 2.9.2.9.3, is used to report the error detected at step 1 of MDS-MU common parsing, application program GDS variable not supported. The MDS message type is "MDS error message"
Format 2 -- SNACR in a CP-MSU	
|___MDS header___ÿ|__application GDS___ÿ| variable | HHHHHHHHHHHHHHHHH |CP-MSU (SNACR (X'7D'))|	The X'7D' subvector is again in an SNA Condition Report (X'1532') GDS variable, but the SNA Condition Report (X'1532') GDS variable itself is encapsulated in a CP-MSU that occupies the MDS-MU's application program GDS variable slot. This format is used for reporting the error detected at step 2 of MDS-MU common parsing, major vector key not supported. For some major vectors (those for which commonality with NMVT parsing is not a consideration), this format is also used for reporting errors detected during major-vector-unique parsing. The MDS message type is "MDS request" or "MDS reply" *not* "MDS error message"
Format 3 -- Reply MV with Sense Data SV in a CP-MSU	
|___MDS header___ÿ|____application GDS____ÿ|| variable || HHHHHHHHHHHHHHHHH |CP-MSU (Reply MV (X'7D'))||	The X'7D' subvector is in the reply major vector corresponding to the request major vector that was received, and the reply major vector is encapsulated in a CP-MSU that occupies the MDS-MU's application program GDS variable slot. This format is used for reporting errors detected during major-vector-unique parsing, for those major vectors for which commonality with NMVT parsing is a consideration. The MDS message type is "MDS request" or "MDS reply" *not* "MDS error message"

Each of the sections in "EP_XXXX Parsing of Individual Management Services Major Vectors" in topic 2.10.8 indicates whether Format 2 or Format 3 is used for error reporting if the request is received in a MDS-MU.

Errors detected by MS_CAPS in the parsing of the MS Capabilities (X'80F0') major vector are always reported using Format 2; see "Summary of MS Capabilities Parsing" in topic 2.9.3.9.

2.9.2.9.5 Canceling an Outstanding Unit of Work with an MDS Error Message

An application program using MDS needs to be able to cancel an outstanding transaction in certain circumstances. An example is an XXXX_NETOP application program that has sent a request for data to the corresponding EP_XXXX application program in a remote node. After a fixed interval of time, the requested data may no longer be of interest to the XXXX_NETOP application program.

In order to cancel an outstanding transaction, an application program must cause cleanup to take place both at its partner application program and at the MDS components between the two application programs. It accomplishes

this by sending a special, application program-level MDS error message canceling the transaction. This error message is constructed as follows:

- Origin Location:
  - NETID and NAU name: The values for the application program's NAU
  - Application program name: The name under which the application registered itself with MDS.
- Destination Location: The NETID, NAU name, and application program name for the partner application program
- Flags:
  - MDS message type: MDS error message
  - First MDS message indicator: 1
  - Last MDS message indicator: 1
- Agent UOW Correlator: The value for the transaction being canceled
- Application program GDS variable: an SNA Condition Report (X'1532') GDS variable containing one structure: the SNA Report Code (X'7D') subvector, containing the SNA sense data X'08A9 0003' (unit of work canceled by MS application program). The format of the SNA Condition Report appears in *SNA Formats*.

Each MDS component treats an MDS error message of this type as an ordinary MDS error message, performing the cleanup described in "MDS Router Processing" in topic 2.9.2.4.8. The partner application program takes the appropriate application program-level cleanup action, defined by its own application program-level architecture.

Table 9-18 summarizes the SNA sense data used in this case.

Table 9-18. Sense Data Returned by MS Application (transaction cancelation)

Order of Checking	Sense Data	Condition	Notes
N/A	X'08A9 0003'	Unit of work canceled by reporting MS application. The unit of work has been canceled because of a timeout in the reporting MS application.	1
N/A	X'08A9 0006'	MS Application internal failure. The unit of work has been canceled because of a problem internal to the reporting MS application.	

Notes:

1. The sense data is sent in the Sense Data (X'7D') subvector of the SNA Condition Report (X'1532') GDS variable. This GDS variable is included in an MDS Error Message directed to the MS application partner.

2.9.3 MS_CAPS Function Set

This function set is concerned with the sending and receiving of messages between nodes in the network to establish a focal point to entry point relationship for a category of management services.

```
° |__________|     |_________________________|           °
°                   (N)"(O)|(R)|(Q)"(U)|(S)"               °
°                      |   |   |   |   |   |   MS_CAPS      °
°                      |   |   |   |   |   |   Function Set °
°°°°°°°°°°°°°°°°°°°°°°°|°°°|°°°|°°°|°°°|°°°|°°°°°°°°°°°°°°°°°
                       |   |   |   |   |   |
                       |         |      |
                    ...............................
                    .                             .
                    .   MULTIPLE_DOMAIN_SUPPORT   .
                    .                             .
                    ...............................
```

[1]XXXX represents ALERT
or OPERATIONS_MGMT

Figure 9-88. MS_CAPS Function Set

Subtopics:

- 2.9.3.1 Protocol Boundaries with Components Outside MS_CAPS
- 2.9.3.2 Prerequisite Function Sets
- 2.9.3.3 Overview of Subsets
- 2.9.3.4 MS_CAPS Base Subset (APPN End Node Support)
- 2.9.3.5 MS_CAPS Optional Subset 1 (Have a Backup or Implicit FP)
- 2.9.3.6 MS_CAPS Optional Subset 2 (Be a Sphere-of-Control End Node)
- 2.9.3.7 MS_CAPS Optional Subset 3 (Base Network Node Support)
- 2.9.3.8 MS_CAPS Optional Subset 4 (Have a Subarea Focal Point)
- 2.9.3.9 Summary of MS Capabilities Parsing
- 2.9.3.10 Application Program Registration with MS_CAPS

2.9.3.1 Protocol Boundaries with Components Outside MS_CAPS

° Input

- Internal MS protocol boundary N (MDS Error Message Information)

  Whenever an MDS error message is received by MULTIPLE_DOMAIN_SUPPORT, it passes a copy of it to MS Capabilities across protocol boundary N.

- Internal MS protocol boundary Q (MDS-MU Received)

  Whenever an MDS-MU is received by the MDS function set which is destined for the MS_CAPS function set, MDS passes the entire MDS-MU across protocol boundary Q.

  The data passed across this protocol boundary is described in "Protocol Boundary Q--MDS-MU Received" in topic 2.9.1.12.

- Internal MS protocol boundary S (Forward CP Status)

  This protocol boundary is used to pass CP status information to MS_CAPS after initialization via protocol boundary U. The data passed is described in "Protocol Boundary S--Forward CP Status" in topic 2.9.1.14.

  The creation of this data is described in "Preparing Forward CP Status" in topic 2.9.2.4.8.

- Internal MS protocol boundary T (Register MS Category)

  In the process of initialization, management services application programs register themselves with the MS_CAPS function set group via protocol boundary T if they want focal point information. The application program passes its name and the category for which it is registering. The details of this protocol boundary are described in "Protocol Boundary T--Register MS Category" in topic 2.9.1.15.

  The process by which an application program registers itself with MS_CAPS is described in "Application Program Registration with MS_CAPS" in topic 2.9.3.10.

° Output

- Internal MS protocol boundary O (Register Application program Name)

An application program utilizing the MDS transport registers itself with the MULTIPLE_DOMAIN_SUPPORT function set group via this protocol boundary. The details of this protocol boundary are described in "Protocol Boundary O--Register Application Program Name" in topic 2.9.1.11.

The process by which an application program registers itself with MULTIPLE_DOMAIN_SUPPORT is described in "Registering with MDS" in topic 2.9.2.9.1.

- Internal MS protocol boundary R (Send MDS-MU)

  Whenever the MS_CAPS function set wishes to communicate with other MS_CAPS in other nodes (to exchange MS Capabilities flows), it creates an MS Capabilities major vector, packages it in a CP-MSU, which is itself placed in an MDS-MU, and passes this structure across protocol boundary R to the MDS function set for transmission.

  The data passed across this protocol boundary is described in "Protocol Boundary R--Send MDS-MU" in topic 2.9.1.13.

- Internal MS protocol boundary U (Begin Sending CP Status Information)

  This protocol boundary is used by MS_CAPS to indicate that MDS is to begin sending CP status information via protocol boundary S. In an APPN end node, the CP status information passed concerns the network node server. In an APPN network node, the CP status information passed concerns the client end nodes. in the network node domain. All current status information is passed back immediately; subsequent status changes will then be reported as well.

  The data passed across this protocol boundary is described in "Protocol Boundary U--Begin Sending CP Status Information" in topic 2.9.1.16.

  This signal is sent as soon as MS_CAPS initializes so that it will receive the CP Status information as soon as possible. It is sent each time MS_CAPS initializes.

- Internal MS protocol boundary V (Current Focal Point)

  The identity of an application program's current focal point is passed across protocol boundary V from the MS_CAPS function set group. The application program must have previously registered itself with MS_CAPS via protocol boundary T. The details of this protocol boundary are described in "Protocol Boundary V--Current Focal Point" in topic 2.9.1.17.

  The process by which an application program registers itself with MS_CAPS is described in "Application Program Registration with MS_CAPS" in topic 2.9.3.10.

2.9.3.2 Prerequisite Function Sets

See "Role Requirements for Management Services Components" in topic 2.8.1 for information on the relationships between this function set and other function sets.

2.9.3.3 Overview of Subsets

```
|                 ________________ __________
|                |   Base         | Be a      |
|                |   Network      | SOC       |
|                |   Node         | End Node  |
|                |   Support      |           |
|                |                |           |_________
|                |   optional     | optional  | Have a  |
|                |   subset 3     | subset 2  | Subarea |
|                |________________|___________| FP      |
|                |     Have a Backup or       |         |
|                |     Implicit FP            |         |
|                |                            | optional |
|                |     optional subset 1      | subset 4 |
|                |____________________________|__________|
|                |       Base APPN End Node Support      |
|                |                                       |
```

```
|              |              base subset              |
|              |_____________________________________|
```

Figure 9-89. Base and Optional Subsets of MS_CAPS Function Set

The MS_CAPS subsets are described in the following sections:

- "MS_CAPS Base Subset (APPN End Node Support)" in topic 2.9.3.4
- "MS_CAPS Optional Subset 1 (Have a Backup or Implicit FP)" in topic 2.9.3.5
- "MS_CAPS Optional Subset 2 (Be a Sphere-of-Control End Node)" in topic 2.9.3.6
- "MS_CAPS Optional Subset 3 (Base Network Node Support)" in topic 2.9.3.7
- "MS_CAPS Optional Subset 4 (Have a Subarea Focal Point)" in topic 2.9.3.8

2.9.3.4 MS_CAPS Base Subset (APPN End Node Support)

Subtopics:

- 2.9.3.4.1 Functions Provided
- 2.9.3.4.2 Formats Supported
- 2.9.3.4.3 Implementation Requirements

2.9.3.4.1 Functions Provided

This is the base MS_CAPS support needed by APPN end nodes to support receiving focal point information from their network node servers. This also provides the support needed to get focal point information to local application programs on a node.

2.9.3.4.2 Formats Supported

```
Network Node                                End Node
_____________________________________________________
  .     MS Caps[FP Notification (X'E1')]      .
  ___________________________________________ÿ
  .                                           .
```

Figure 9-90. MS Capabilities Flow Supported by MS_CAPS Base Subset

This subset supports the receipt of the following management services major vector and subvectors in MDS-MUs:

- MS Capabilities (X'80F0') major vector
- Focal Point Notification (X'E1') subvector
- Focal Point Identification (X'21') subvector

The MS Capabilities flow supported by this subset is illustrated in Figure 9-90.

2.9.3.4.3 Implementation Requirements

The requirements for implementing the MS Capabilities base subset are escribed by a model consisting of CP Management Services.

A Subset of CP Management Services

- Provides the following processes:
  - Registering Application Programs

-- Receives registration from local application programs for each category they are interested in

-- Enters the application program names into the MS Capabilities table

-- Starts the process which notifies application programs of focal point information, if a focal point is entered in the table for that category already.

- Receiving CP Status Updates

  -- Receives copies of CP Status messages from MDS

  -- Processes these messages for CP Capabilities information about its network node server.

- Receiving MS Capabilities Requests

  -- Receives requests (only requests without replies are accepted)

  -- Parses request, fills in MS Capabilities table with any appropriate information

  -- Starts the process which notifies application programs of focal point information

- Notifying Application Programs of Focal Point Identity

  -- Builds an MS Capabilities (X'80F0') major vector, if one was not passed to it, containing a Focal Point Notification (X'E1') subvector and a Focal Point Identification (X'21') subvector

  -- Passes this information, and an indication that the routing to be used is default, to the application program that just registered, or to all the application programs for a category, depending on the process that started it (the information it received).

- MDS Error Message Processing

  -- Builds MDS error messages for exception conditions it detects.

  -- Receives and processes MDS error messages.

Registering Application Programs:

This process receives information across protocol boundary T about local application programs. Local application programs "register" themselves with MS_CAPS if they are interested in knowing the focal point for a certain category of management services data. An application program registers for each category in which it is interested.

MS_CAPS built a MS Capabilities table to keep focal point information in, when it initialized. It adds the application program information it receives from this registration to that table. An example table is inTable 9-19 .

If there is already focal point information available in the MS Capabilities table for the category this application program registered for, this process starts the process described in "Notifying Application Programs of Focal Point Identity," passing it the application program name and category that focal point information should be sent for.

Table 9-19. MS Capabilities Table for End Nodes

Category	Domain Focal Point			Local Appls
	Status	Routing	Focal Point Name	
X'23F0F3F1'	Active	Default	NET1 NODE1 X'23F0F3F1'	X'23F0F3F0' USERAP1
X'23F0F1F6'	Active	Default	NET1 NODE1 X'23F0F1F6'	X'23F0F1F7' USERAP9

USERCAT	Reset	Default	NET1 NODE1 USERAP2	USERAP3
Note: X'23F0F3F1' = ALERT_NETOP X'23F0F3F0' = EP_ALERT X'23F0F1F7' = OPERATIONS_MGMT_NETOP X'23F0F1F6' = EP_OPERATIONS_MGMT				

Table 9-20. Description of MS Capabilities Table Information. Each row of the MS Capabilities table contains the information described below.

Category

The name of the MS category associated with the focal points and application programs in this row. A row in the MS Capabilities Table is created when an MS application program program registers for a category not already listed in the table, or when a focal point relationship is established for a category not already in the table.

Domain Focal Point Information

° Status

- Reset:

  There is no focal point for the category, either because no Focal Point Notification (X'E1') subvector has been received for the category, or because a previous focal point for the category was reset by a subsequent Focal Point Notification (X'E1') subvector with null values for the focal point.
- Active:

  There is a focal point for the category (a Focal Point Notification (X'E1') subvector with non-null names has been received)

° Routing

- Default:

  For a base end node, this is the only value defined.

° Focal Point Name

- NETID name
- NAU name
- MS application program name

Local application programs

A list of local MS application programs which have registered with MS_CAPS for focal point information about this category.

Receiving CP Status Updates:

This function is started when MS_CAPS receives data across protocol boundary S. Across it, the MDS function set passes a CP Status message. This process parses the message for CP Capabilities information about its network node server. MS_CAPS must keep track of who its network node server is. It will receive information about the focal point for this end node's domain from the network node server. If this end node changes network node domains (i.e., gets a new network node server) this end node should erase any focal point information that it received from its former network node server, and replace it with new information from its network node server, when it gets it. The CP Capabilities exchange contains an indication of whether the network node server supports sending its end nodes MS Capabilities information.

If the CP Capabilities indicates that the network node server does not support sending MS Capabilities requests and replies (byte 8, bit 6 of CP Capabilities is set to 0) this end node must assume it will not receive focal point information from its network node server and must take the following action. This process then fills in the MS Capabilities table with the following information and passes the category to the process described in "Notifying Application Programs of Focal Point Identity."

° Category = ALERT
° Focal Point Identification = NETID.CPNAME of the network node server
° Focal Point Application program name = ALERT_NETOP

Receiving MS Capabilities Requests:

This process is started via MS protocol boundary Q. This process handles

MDS message type 'request without reply'. If it receives any other type of message, it rejects it and sends an MDS error message ("MDS Error Message Processing") with sense data X'080C 0013'.

Note: This restriction on MDS message type applies only to "MS_CAPS Base Subset (APPN End Node Support)" in topic 2.9.3.4. Other MS_CAPS optional subsets support MDS 'request with reply'.

If this message is a request without reply it checks to see that the application program GDS variable ID is X'1212'. If it is not, it rejects it and sends an MDS error message with sense data X'080C 0012'. If it is, this MDS-MU contains a CP-MSU.

Note: An MDS error message is used only for reporting the errors described above (unsupported MDS message type and unsupported GDS variable). Other MS Capabilities parsing errors (described below) are reported using "Format 2" in Table 9-17 in topic 2.9.2.9.4.

Table 9-21. Contents of MS Capabilities Request (X'E1') - Focal Point Notification

Structure Name	MV/SV/SF	Description of Contents
MS Capabilities	X'80F0'	
Focal Point Notification	X'E1'	
Focal Point Identification	X'21'	Identifies the current focal point (note 1)
MS Category	X'01'	Category name
Focal Point Identification Flags	X'02'	All flags 0
Focal Point NETID	X'10'	Network ID of focal point
Focal Point NAU Name	X'11'	NAU name of focal point
Focal Point Appl. program ID	X'12'	MS appl. program name of focal point

Note:

1. Focal Point Identification (X'21') subvector always identifies the *current* focal point.

The format of the MS Capabilities MV accepted by this optional subset is illustrated in Table 9-21.

This process parses the CP-MSU. The only major vector value this process accepts is X'80F0' (MS Capabilities). Any other value is rejected with sense data X'080C 0005'. The only allowed command subvector value is X'E1'. Any other value is rejected with sense data X'080C 0006'. This process also requires a X'21' subvector in the same major vector. If it is not present, this is rejected with sense data X'086C 2100'. Refer to Table 9-39 in topic 2.9.3.9 for a complete list of the the sense data associated with processing of the MS Capabilities major vector.

Once the request has been parsed, this process fills in the MS Capabilities table with the focal point identification information that it received. It fills in the routing column with 'E1' as this is not a SOC node. It starts the process described in "Notifying Application Programs of Focal Point Identity," passing it only the category for which the new focal point information was received.

Notifying Application Programs of Focal Point Identity:

This function is responsible for notification of local application programs of a focal point's identity. It is started by several processes. When it is started because of the registration of a new application program, that application program name and the category it is registering for is passed to it. This process then builds an MS Capabilities (X'80F0') major vector containing a Focal Point Notification (X'E1') subvector and a Focal Point Identification (X'21') subvector with the focal point information for that category from the MS Capabilities table (if there is any). This process completes the MS Capabilities major vector and passes it and an indication of the routing type to use (from the table) to that application program via protocol boundary V (described in "Protocol Boundary V--Current Focal Point" in topic 2.9.1.17). If there is no focal point information in the table for that category, no MS Capabilities is sent. This process then terminates.

When this process is started because of the receipt of an MS Capabilities request, only the category is passed to it, and this process builds and

sends MS Capabilities (X'80F0') major vectors containing an Focal Point Notification (X'E1') subvector and a Focal Point Identification (X'21') subvector containing the focal point information obtained from the MS Capabilities table, as well as an indication of the routing type (also obtained from the table) to each of the application programs listed for that category, via protocol boundary V (described in "Protocol Boundary V--Current Focal Point" in topic 2.9.1.17). If the network node server sent empty focal point identification subfields, then this node does not currently have a focal point. This node will notify its application programs of this by sending empty focal point identification subfields in the MS Capabilities major vector that it builds.

Empty Focal Point Identification Subfields:

The loss of a focal point is reported by the creation of MS Capabilities (X'80F0') major vector with an Focal Point Notification (X'E1') subvector and a Focal Point Identification (X'21') subvector with the following subfield contents:

- MS Category (X'01') - category name
- Focal Point Identification Flags (X'02')
- Focal Point NETID (X'10') - null
- Focal Point NAU Name (X'11') - null
- Focal Point Application program ID (X'12') - null

A "null" name is one with a length of 0 (i.e., the subfield consists of only a length and key) or one which has no significant characters (all blanks).

MDS Error Message Processing:

This process builds MDS error messages. It is passed the sense codes that are to be included from the process detecting the error condition. It passes the completed MDS error message to MDS for sending via protocol boundary R (described in "Protocol Boundary R--Send MDS-MU" in topic 2.9.1.13).

This process also receives MDS error message information via protocol boundary N (described in "Protocol Boundary N--MDS Error Message Information" in topic 2.9.1.10). MDS passes a copy of every MDS error message that it processes for which the origin application program name is X'23F0F1F0' (MDS_ROUTER) to MS_CAPS. This process determines if the MDS error message impacts any of the focal points in the MS Capabilities table.

The following exceptions are made:

- When the sense data in the MDS error message is X'08A9 0007' (MS application router re-initialization MDS transaction exception). In this case no MS_CAPS recovery action is required.

- When the sense data in the MDS error message is X'0000 0000'. This is passed by MDS to notify MS_CAPS of the normal UNBIND of the last SNASVCMG session to a session partner when there are no impacted MDS tranactions (i.e., no actual MDS error messages generated). This is not an error situation, since the SNASVCMG sessions may be deactivated by the LU when they are not in use to conserve resources. If SNASVCMG sessions are deactivated because of network connectivity failures, MDS will pass the non-zero UNBIND sense data, and MS_CAPS will act accordingly (following steps).

If this is not an exception case, the focal point names (NETID, NAU, and MS application program) in the MS Capabilities table are searched for every match on the Reported on location (NETID and NAU) from the signal:

If a match is found on location, the MS application program name from the signal is examined:

- If the reported on MS application program is X'23F0F1F1' (MS_CAPS), then all MS application programs in the reported on node, including the focal point application program in the MS Capabilities table, are impacted. The reported on application program is set to MS_CAPS when the MDS_ROUTER in this node reports on an SNASVCMG session outage.

- If the reported on MS application program is some other value, then it is compared to the focal point application program name in the MS Capabilities table entry.

If a match is found in either case, the MS Capabilities table is updated. The focal point node is removed from the table, and this process starts the process described in "Notifying Application Programs of Focal Point Identity," passing it the category for which the MS Capabilities table was just updated.

This process then terminates.

The processing of focal point information is summarized in finite-state machine format in "MS_CAPS FSM: Focal Point Status for Base End Node."

MS_CAPS FSM: Focal Point Status for Base End Node

Function:	This finite-state machine describes the processing of a single row in the MS Capabilities table, which is the information associated with a single category. Refer to Table 9-19 for a description of the information in the MS Capabilities table. The only processing described is that associated with the maintenance of the focal point information and the receipt of MS Capabilities messages. Not included is the maintenance of local MS application program program status for each category.
Inputs:	1. Focal Point Notification (X'E1') subvector received from network node server 2. "Protocol Boundary N--MDS Error Message Information" in topic 2.9.1.10 The report code (sense data) in the MDS error message information is examined: a. X'08A9 0007' MDS application router reinitialized. The MDS_ROUTER at the focal point experienced some failure, but recovery has taken place. No entry point action is required. b. If the sense data is any other nonzero value, it indicates a communication failure with the focal point application program. The appropriate recovery action will be taken by MS_CAPS in the network node server.
States:	01 Reset No focal point for category 02 Domain focal point -- active The end node is using the focal point it learned about from a Focal Point Notification (X'E1') subvector received from its network node server.

	States	
Inputs	**Reset** **01**	**Domain FP active** **02**
SV X'E1' from network node server	02a	02a
MDS Error Msg Info (PB N)	/	01a

Output Code	**Function**
a	1. FP information updated. 2. Focal Point Notification (X'E1') subvector sent to local application programs (may carry names with null values if FP is not active).

2.9.3.5 MS_CAPS Optional Subset 1 (Have a Backup or Implicit FP)

Subtopics:

- 2.9.3.5.1 Functions Provided
- 2.9.3.5.2 Formats Supported
- 2.9.3.5.3 Implementation Requirements

2.9.3.5.1 Functions Provided

This is the MS Capabilities support needed to support having a backup focal point or an implicit focal point. It includes the ability to establish implicit focal point relationships, retry implicit primary focal points, a process for switching to a backup focal point, and changes to associated processes to make these functions work. It also allows for local input of focal point definition information. The implicit focal point information must be recorded on non-volatile storage so that it is available when the node is initialized.

2.9.3.5.2 Formats Supported

```
 Focal Point                                   Entry Point
_____________________________________________________________
  .        MS Caps[EP Auth. Request (X'63')]        .
  _____________________________________________________
  .         MS Caps[EP Auth. Reply (X'64')]         .
  _____________________________________________________ÿ
  .    MS Caps[FP Auth. Reply (X'62') (revoke)]     .
  _____________________________________________________
  .                                                 .
```

Figure 9-91. MS Capabilities Flows Supported by MS_CAPS Optional Subset 1

This subset supports the receipt of the following management services major vector and subvectors in MDS-MUs:

- MS Capabilities (X'80F0') major vector
- Entry Point Authorization Reply (X'64') subvector
- Focal Point Identification (X'21') subvector

This subset supports the sending of the following management services major vector and subvectors in MDS-MUs:

- MS Capabilities (X'80F0') major vector
- Focal Point Authorization Reply (X'62') subvector
- Entry Point Authorization Request (X'63') subvector
- Focal Point Identification (X'21') subvector

The MS Capabilities flows supported by this optional subset are illustrated in Figure 9-91.

2.9.3.5.3 Implementation Requirements

The requirements for implementing MS_CAPS optional subset 1 are described by a model consisting of CP Management Services.

A Subset of CP Management Services

- In addition to the functions provided in MS_CAPS subsets prerequisite to this subset, this subset provides the following processes:

  - Establishing Implicit Focal Point Relationships

    -- Initiates implicit focal point requests when this node has been system or operator defined with focal point information.

    -- Whenever an implicit focal point name is changed by operator input, the previous focal point must be revoked by the sending of a Focal Point Authorization Reply (X'62') subvector with the Authorization Revoked (X'30') subfield. This revocation

must be attempted even when the previous focal point is not currently active.

- Retry Implicit Focal Point Mechanism

  -- Initiates retry of implicit focal point requests when this node has been system or operator defined with focal point information

- Save Implicit Focal Point Information to Non-Volatile Storage

  -- Implicit focal point names must be saved in non-volatile storage so that it will be available when the node is initialized.

- Changing to a Backup Focal Point

  -- Performs switches to a backup focal point when an MDS error message indicating a transport error to the focal point is received.

- Receiving MS Capabilities Replies

  -- Receives and processes X'64' replies to MS Capabilities requests sent from this node.

° This subset provides additional function to the following process:

- Sending MS Capabilities Requests (Change is in addition to what was described in "Sending MS Capabilities Requests")

  -- Builds and sends MS Capabilities requests containing the X'63' subvector.

Establishing Implicit Focal Point Relationships:

If this node has been system or operator defined with focal point information, this process is started immediately after MS_CAPS initializes or after focal point entries in the MS Capabilities table have been exhausted to establish the focal point -- entry point relationship. It reads in the system or operator defined entries, and fills in the MS Capabilities table. If the focal point information from operator or system input revises the previously stored implicit focal point information, then the previous focal point must be revoked by sending MS Capabilities with Focal Point Authorization Reply (X'62') subvector indicating "revoked." These entries are labeled 'pending' until a response is received to the request for services. It then starts the process defined in "Sending MS Capabilities Requests" to request the services of the focal point. If all the focal point information in the MS Capabilities table is exhausted, that is, the focal point -- entry point relationship has been broken for both the primary and backup (if applicable) focal points, this process is started to re-read in the system or operator defined focal point information and start the establishment over again.

Retry Implicit Focal Point Mechanism:

This process periodically re-starts "Establishing Implicit Focal Point Relationships" if that process was previously unsuccessful. The algorithm which determines when this retry should take place is defined by individual implementations, and not by the architecture.

Changing to a Backup Focal Point:

This process is started by "MDS Error Message Processing" in topic 2.9.3.4.3 when it receives an MDS error message pertaining to any of the focal points in its MS Capabilities table This process then changes the corresponding entry in the MS Capabilities table for that category, replacing the current entry with the backup entry, entering its current status as 'pending'. This process then starts the process described in "Sending MS Capabilities Requests" to request the services of this focal point.

If there are no entries in the MS Capabilities table as backups, and this focal point has been system or operator defined with focal point information, this process starts the process described in "Establishing

Implicit Focal Point Relationships" and terminates.

Sending MS Capabilities Requests:

This process is used to build MS Capabilities requests. It is passed the focal point name and an indication of whether this is to be a request for primary or backup service.

Table 9-22. Contents of MS Capabilities Request (X'63') - Implicit

Structure Name	MV/SV/SF	Description of Contents
MS Capabilities	X'80F0'	
Entry Point Auth. Request	X'63'	
Primary Authorization Request	X'10'	
Focal Point Identification	X'21'	Identifies the current focal point (note 1)
MS Category	X'01'	Category name
Focal Point Identification Flags	X'02'	All flags 0
Focal Point NETID	X'10'	Network ID of focal point
Focal Point NAU Name	X'11'	NAU name of focal point
Focal Point Appl. program ID	X'12'	MS appl. program name of focal point

Note:

1. Focal Point Identification (X'21') subvector always identifies the *current* focal point. For an entry point request for a primary focal point (an *implicit* focal point), the current focal point is the one to which the entry point is sending the request.

The MS Capabilities built for a primary request is illustrated in Table 9-22.

Table 9-23. Contents of MS Capabilities Request (X'63') - Backup

Structure Name	MV/SV/SF	Description of Contents
MS Capabilities	X'80F0'	
Entry Point Auth. Request	X'63'	
Backup Authorization Request	X'30'	
Focal Point Identification	X'21'	Identifies the current focal point (note 1)
MS Category	X'01'	Category name
Focal Point Identification Flags	X'02'	All flags 0
Focal Point NETID	X'10'	Network ID of focal point
Focal Point NAU Name	X'11'	NAU name of focal point
Focal Point Appl. program ID	X'12'	MS appl. program name of focal point

Note:

1. Focal Point Identification (X'21') subvector always identifies the *current* focal point. For an entry point request for a backup focal point, the current focal point is the one to which the entry point is sending the request.

The MS Capabilities built for a backup request is illustrated in Table 9-23. This process builds the appropriate MS Capabilities major vector, places it within a CP-MSU, builds the MDS-MU header information, and passes the completed MDS-MU across protocol boundary R for sending. This process keeps a correlation control block with the unit-of-work correlator in it, to correlate this request with its reply.

Receiving MS Capabilities Replies:

This process is started via protocol boundary Q. This process first attempts to match the correlator to one of its outstanding requests. If it does not match any, the CP-MSU is discarded.

Note: This condition should not occur, since an uncorrelated MDS reply would have been rejected by MDS and would not have been passed to MS_CAPS.

If the reply correlator matches, the correlation control block can be cleaned up, and the content of the reply is parsed to see whether it was accepted or rejected.

If it was rejected, this process starts the process described in "Establishing Implicit Focal Point Relationships" and terminates.

If the reply correlator matches, the correlation control block is deleted, and the content of the MS Capabilities reply (Entry Point Authorization Reply (X'64') subvector) is examined to see whether the MS Capabilities request (Entry Point Authorization Request (X'63') subvector) was accepted or rejected.

Table 9-24. Contents of MS Capabilities Reply (X'64') - Reject

Structure Name	MV/SV/SF	Description of Contents
MS Capabilities	X'80F0'	
Entry Point Auth. Reply	X'64'	
Authorization Request Rejected	X'20'	
Authorization status code	--	X'01' Function not supported X'02' Sender is not a focal point for the category X'03' Sender unable to support request at this time
Focal Point Identification	X'21'	Identifies the current focal point (note 1)
MS Category	X'01'	Category name
Focal Point Identification Flags	X'02'	All flags 0
Focal Point NETID	X'10'	Network ID of focal point
Focal Point NAU Name	X'11'	NAU name of focal point
Focal Point Appl. program ID	X'12'	MS application program name of focal point

Note:

1. Focal Point Identification (X'21') subvector always identifies the *current* focal point. For an entry point reply from a focal point, the current focal point is the one sending the reply. For a reject reply, the subfield values in this subvector simply echo what was sent by the entry point in the request.

If the request was rejected (as illustrated in Table 9-24), this process starts the process described in "Establishing Implicit Focal Point Relationships" and terminates.

Table 9-25. Contents of MS Capabilities Reply (X'64') - Accept

Structure Name	MV/SV/SF	Description of Contents
MS Capabilities	X'80F0'	
Entry Point Auth. Reply	X'64'	
Authorization Request Accepted	X'10'	
Focal Point Identification	X'21'	Identifies the current focal point (note 1)
MS Category	X'01'	Category name
Focal Point Identification Flags	X'02'	All flags 0
Focal Point NETID	X'10'	Network ID of focal point
Focal Point NAU Name	X'11'	NAU name of focal point

Focal Point Appl. program ID	X'12'	MS appl. program name of focal point (note 2)

Note:

1. Focal Point Identification (X'21') subvector always identifies the *current* focal point. For an entry point reply from a focal point, the current focal point is the one sending the reply.

2. The MS application program name returned by the focal point in the X'12' subfield contains the name that the entry point is to use as the destination for the category. This MS application program name sent by the focal point supersedes the MS application program name which the entry point sent on the Entry Point Authorization Request (X'63') subvector.

If it was accepted (as illustrated in Table 9-25), the entry in the MS Capabilities table is updated, indicating the focal point is active (as opposed to pending) and completes the entry with the routing type of 'direct', as this node will use direct routing to its focal point, and is implicitly in its sphere of control. This process then starts the "Notifying Application Programs of Focal Point Identity" in topic 2.9.3.4.3 and "Sending MS Capabilities Requests" processes, passing them the category for which new focal point information was obtained. This process then terminates.

See Table 9-26 for an illustration of the MS Capabilities table maintained by an end node which supports only the base and this optional subset.

Table 9-26. MS Capabilities Table for End Nodes Supporting Backup and Implicit Focal Points

Category	Primary Focal Point			Backup Focal Point			Domain Focal Point			Local Appls
	Stat.	Rout.	FP Name	Stat.	Rout.	FP Name	Stat.	Rout.	FP Name	
X'23F0F3F1'	act.	dir.	NET1 NODE1 X'23F0F3F1	inact.	dir.	NET1 NODE2 X'23F0F3F1	act.	def.	NET1 NODE4 X'23F0F3F1	X'23F0F3F0' USERAP1
X'23F0F1F7'	act.	dir.	NET1 NODE1 X'23F0F1F7	inact.	dir.	NET1 NODE2 X'23F0F1F7	act.	def.	NET1 NODE4 X'23F0F1F7	X'23F0F1F6' USERAP4
USERCAT	act.	dir.	NET1 NODE1 USERAP2	inact.	dir.		inact.	def.		USERAP3

Note:

X'23F0F3F1' = ALERT_NETOP
X'23F0F3F0' = EP_ALERT
X'23F0F1F7' = OPERATIONS_MGMT_NETOP
X'23F0F1F6' = EP_OPERATIONS_MGMT

Table 9-27. Description of MS Capabilities Table Information. Each row of the MS Capabilities table contains the information described below.

Category

The name of the MS category associated with the focal points and application programs in this row. A row in the MS Capabilities table created when an MS application program program registers for a category not already listed in the table, or when a focal point relationship is established for a category not already in the table.

Primary Focal Point Information

° Status

- Inactive
- Active

° Routing

- DEFAULT

  All traffic for this focal point will be routed using the *default* routing option of MULTIPLE_DOMAIN_SUPPORT. This value applies only to a domain focal point used by an end node.

- DIRECT

  All traffic for this focal point will be routed using the *direct* routing option of MULTIPLE_DOMAIN_SUPPORT.

° Focal Point Name

```
|                                                                      |
|     -   NETID name                                                   |
|     -   NAU name                                                     |
|     -   MS application program name                                  |
|__________________________________________________________________________|
| Backup Focal Point Information                                       |
|                                                                      |
| °   Status                                                           |
|                                                                      |
|     -   Inactive                                                     |
|     -   Active                                                       |
|                                                                      |
| °   Routing                                                          |
|                                                                      |
|     -   DIRECT                                                       |
|                                                                      |
| °   Focal Point Name                                                 |
|                                                                      |
|     -   NETID name                                                   |
|     -   NAU name                                                     |
|     -   MS application program name                                  |
|__________________________________________________________________________|
| Domain Focal Point Information                                       |
|                                                                      |
| °   Status                                                           |
|                                                                      |
|     -   Inactive                                                     |
|     -   Active                                                       |
|                                                                      |
| °   Routing                                                          |
|                                                                      |
|     -   DEFAULT                                                      |
|                                                                      |
| °   Focal Point Name                                                 |
|                                                                      |
|     -   NETID name                                                   |
|     -   NAU name                                                     |
|     -   MS application program name                                  |
|__________________________________________________________________________|
| Local application programs                                           |
|                                                                      |
| A list of local MS application programs that have registered with    |
| MS_CAPS for focal point information about this category.             |
|__________________________________________________________________________|
```

2.9.3.6 MS_CAPS Optional Subset 2 (Be a Sphere-of-Control End Node)

Subtopics:

- 2.9.3.6.1 Functions Provided
- 2.9.3.6.2 Formats Supported
- 2.9.3.6.3 Implementation Requirements

2.9.3.6.1 Functions Provided

This optional subset provides the functions that are necessary for an end node to be in a sphere of control (SOC) of a focal point. This differs from "MS_CAPS Base Subset (APPN End Node Support)" in topic 2.9.3.4, since in that subset, an end node does not communicate directly with the focal point; rather, it communicates indirectly through its network node server.

2.9.3.6.2 Formats Supported

```
Focal Point                                                End Node
_____________________________________________________________________
  .              MS Caps[FP Auth. Request (X'61')]              .
  ______________________________________________________________ÿ
  .               MS Caps[FP Auth. Reply (X'62')]               .
  ______________________________________________________________
  .                                                             .
```

Figure 9-92. MS Capabilities Flows Supported by MS_CAPS Optional Subset 2

This subset supports the receipt of the following management services major vector and subvectors in an MDS-MU:

- MS Capabilities (X'80F0') major vector
- Focal Point Authorization Request (X'61') subvector
- Focal Point Notification (X'E1') subvector
- Focal Point Identification (X'21') subvector

It supports the sending of the following management services major vector and subvectors in an MDS-MU:

- MS Capabilities (X'80F0') major vector
- Focal Point Authorization Reply (X'62') subvector
- Focal Point Identification (X'21') subvector

Figure 9-92 illustrates the MS Capabilities exchange supported by this subset.

2.9.3.6.3 Implementation Requirements

The requirements for implementing the MS Capabilities optional subset 1 are described by a model consisting of CP Management Services. "MS_CAPS Optional Subset 1 (Have a Backup or Implicit FP)" in topic 2.9.3.5 is a prerequisite to this subset.

A Subset of CP Management Services

- In addition to the functions provided in MS_CAPS subsets prerequisite to this subset, this subset provides the following processes:

  - Sending MS Capabilities Replies

    -- Builds an MDS-MU containing a MS Capabilities major vector with a Focal Point Authorization Reply (X'62') subvector (and accompanying subvectors) in reply to receipt of a Focal Point Authorization Request (X'61') subvector (and accompanying subvectors).

    -- Passes these MDS-MUs to MDS for sending.

- This process provides additional function in the following processes:

  - Receiving MS Capabilities Requests (Change in addition to function in "Receiving MS Capabilities Requests" in topic 2.9.3.4.3)

    -- Accepts MS Capabilities requests that contain the X'61' subvector (accepts request with reply message types).

  - MDS Error Message Processing (Change in addition to function in "MDS Error Message Processing" in topic 2.9.3.4.3)

    -- Processes MDS Error Message Information (passed by MDS via protocol boundary N) to update MS Capabilities table, initiate recovery, or to notify local application programs and client end nodes when a focal point is lost.

  - In addition to the functions described in "Changing to a Backup Focal Point" in topic 2.9.3.5.3, this optional subset includes the following function:

    -- Focal point identification received in a Focal Point Notification (X'E1') subvector from a network node server is saved even if the end node currently has a focal point for the category (Focal Point Authorization Request (X'61') subvector has been received directly from the focal point). The focal point identified in the X'E1' will become the active focal point for the category whenever the focal point identified in the X'61' is no longer available.

    Since "MS_CAPS Optional Subset 1 (Have a Backup or Implicit FP)" in topic 2.9.3.5 is a prerequisite to this optional subset, the same logic applies to backup focal points and implicit focal points. The focal point for a category received in a Focal Point Notification (X'E1') subvector will be saved and used whenever the SOC end node does not have an active focal point acquired by a X'63'/X'64' exchange.

The MS Capabilities table maintained by a SOC end node is illustrated in Table 9-28.

Table 9-28. MS Capabilities Table for SOC End Nodes

Category	Primary Focal Point					Backup Focal Point			Domain Focal Point			Local Appls
	Stat.	Recov.	Rout.	Rank	FP Name	Stat.	Rout.	FP Name	Stat.	Rout.	FP Name	

X'23F0F3F1'	act.	FP	dir.	expl.	NET1 NODE1 X'23F0F3F1	inact.	dir.	NET1 NODE2 X'23F0F3F1	act.	def.	NET1 NODE4 X'23F0F3F1	X'23F0F3F0' USERAP1
X'23F0F1F7'	act.	FP	dir.	expl.	NET1 NODE1 X'23F0F1F7	inact.	dir.	NET1 NODE2 X'23F0F1F7	act.	def.	NET1 NODE4 X'23F0F1F7	X'23F0F1F6' USERAP41
USERCAT	act.	EP	dir.	impl.	NET1 NODE1 USERAP2	inact.	dir.		inact.	def.		USERAP3

Note:

X'23F0F3F1' = ALERT_NETOP
X'23F0F3F0' = EP_ALERT
X'23F0F1F7' = OPERATIONS_MGMT_NETOP
X'23F0F1F6' = EP_OPERATIONS_MGMT

Table 9-29. Description of MS Capabilities Table Information. Each row of the MS Capabilities table contains the information described below.

Category

The name of the MS category associated with the focal points and application programs in this row. A row in the MS Capabilities table created when an MS application program program registers for a category not already listed in the table, or when a focal point relationship is established for a category not already in the table.

Primary Focal Point Information

° Status

- Inactive
- Active

° MS Capabilities recovery responsibility

- FP (focal point will initiate recovery by sending MS Capabilities)
- EP (entry point will initiate recovery by sending MS Capabilities)

This value is based upon the setting of the MS Capabilities Recovery Responsibility indicator (byte 4 bit 0) in Focal Point Authorization Request (X'61') subvector received from an explicit focal point. It is set to EP for an implicit focal point.

° Routing

- DEFAULT

  All traffic for this focal point will be routed using the *default* routing option of MULTIPLE_DOMAIN_SUPPORT. This value applies only to a domain focal point used by an end node.

- DIRECT

  All traffic for this focal point will be routed using the *direct* routing option of MULTIPLE_DOMAIN_SUPPORT.

° Rank

This field actually contains the focal point type:

1. Explicit
2. Implicit

Rank refers to the preference given to a focal point of one type of a focal point of another type. The list of focal point types in order of rank is:

1. Explicit primary
2. Implicit primary
3. Backup
4. Domain FP (from network node server)

The rank is used to determine whether or not the entry point will accept a particular Focal Point Authorization Request (X'61') subvector request. If the rank of the focal point sending the request is lower than the current focal point, the entry point will reject the request. If it is equal or higher in rank, the request will be accepted and the previous focal point revoked.

Note that a backup focal point has the same rank whether its identity originally came from implicit definition or was learned in a Focal Point Authorization Request (X'61') subvector request from an explicit primary.

If a local operator at the entry point overrides the current focal point with new implicit imformation, that new focal point is accepted regardless of the ranking list.

° Focal Point Name

- NETID name

```
|     -   NAU name                                                      |
|     -   MS application program name                                   |
|_______________________________________________________________________|
| Backup Focal Point Information                                        |
|                                                                       |
| °   Status                                                            |
|                                                                       |
|     -   Inactive                                                      |
|     -   Active                                                        |
|                                                                       |
| °   Routing                                                           |
|                                                                       |
|     -   DIRECT                                                        |
|                                                                       |
| °   Focal Point Name                                                  |
|                                                                       |
|     -   NETID name                                                    |
|     -   NAU name                                                      |
|     -   MS application program name                                   |
|_______________________________________________________________________|
| Domain Focal Point Information                                        |
|                                                                       |
| °   Status                                                            |
|                                                                       |
|     -   Inactive                                                      |
|     -   Active                                                        |
|                                                                       |
| °   Routing                                                           |
|                                                                       |
|     -   DEFAULT                                                       |
|                                                                       |
| °   Focal Point Name                                                  |
|                                                                       |
|     -   NETID name                                                    |
|     -   NAU name                                                      |
|     -   MS application program name                                   |
|_______________________________________________________________________|
| Local application programs                                            |
|                                                                       |
| A list of local MS application programs that have registered with     |
| MS_CAPS for focal point information about this category.              |
|_______________________________________________________________________|
```

Receiving MS Capabilities Requests:

This process is started via protocol boundary Q. This process provides function in addition to that in "Receiving MS Capabilities Requests" in topic 2.9.3.4.3 as follows: It supports receiving messages of the type 'request with reply'.

If this message is a request (with or without reply) it checks to see if the application program GDS variable is X'1212'. If it is not, it rejects this with sense data X'080C 0012'. If it is, this MDS-MU contains a CP-MSU.

Note: An MDS error message is used only for reporting the errors described above (unsupported MDS message type and unsupported GDS variable). Other MS Capabilities parsing errors (described below) are reported using "Format 2" in Table 9-17 in topic 2.9.2.9.4.

This process parses the CP-MSU. The only major vector value this process accepts is X'80F0'. Any other value is rejected with sense data X'080C 0005'. The only command subvector this process accepts is X'61'. Any other value is rejected with the sense data X'080C 0006'. It also requires a X'21' subvector in the same major vector. If it is not present it is rejected with the sense data X'086C 2100'. Refer to Table 9-39 in topic 2.9.3.9 for a complete list of the sense data associated with MS Capabilities major vector processing.

Table 9-30 illustrates the format of the MS Capabilities major vector received from a primary focal point.

Table 9-30. Contents of MS Capabilities Request (X'61') - Primary Focal Point

Structure Name	MV/SV/SF	Description of Contents
MS Capabilities	X'80F0'	
Focal Point Auth. Request	X'61'	
Primary Authorization Request	X'10'	
Back-level category	--	X'0000' Category = X'23F0F3F1' X'FFFF' Category &ne. X'23F0F3F1'
MS Caps recovery responsibility	--	0 Focal point will perform recovery 1 Entry point must perform recovery

Focal Point Identification	X'21'	Identifies the current focal point
MS Category	X'01'	Category name
Focal Point Identification Flags	X'02'	
Backup focal point	--	0 Process backup focal point SFs as usual 1 Keep current backup focal point, if any
Focal Point NETID	X'10'	Network ID of focal point
Focal Point NAU Name	X'11'	NAU name of focal point
Focal Point Appl. program ID	X'12'	MS appl. program name of focal point
Backup Focal Point NETID	X'20'	Network ID of backup focal point (note 2)
Backup Focal Point NAU Name	X'21'	NAU name of backup focal point (note 2)
Backup Focal Point Appl. program ID	X'22'	MS appl. program name of backup focal point (note 2)

Note:

1. Focal Point Identification (X'21') subvector always identifies the *current* focal point. For a request, the current focal point is the one sending MS Capabilities.

2. The subfields identifying a backup focal point are present in the request only if a backup is defined for the category.

Processing of MS Capabilities Recovery Responsibility:

If the MS Capabilities Recovery Responsibility indicator (byte 4 bit 0) in Primary Authorization Request (X'10') subfield of the X'61' subvector is set to 1 indicating that the entry point is responsible for recovery, the *implicit* focal point definition for the category is updated on non-volatile storage. This will enable "Retry Implicit Focal Point Mechanism" in topic 2.9.3.5.3 to recover the focal point to entry point relationship after a failure.

Processing of Backup Focal Point Identification Subfields:

The backup focal point is identified by the following subfields in Focal Point Identification (X'21') subvector:

- Backup Focal Point NETID (X'20')
- Backup Focal Point NAU Name (X'21')
- Backup Focal Point Application program ID (X'22')

The processing of these subfields is controlled by the setting of the Backup Focal Point flag (byte 2 bit 0) in the Focal Point Identification Flags (X'02') subfield:

0 Process as usual

The values in subfields X'20', X'21', and X'22', if present, will be accepted as the new backup focal point identification. If the subfields are present and contain null values, or if the subfields are not present, the current backup information, if any, is to be deleted.

1 Keep the same backup focal point name as you have now.

The X'20', X'21', and X'22' subfields are not present. The current backup focal point name, if any, is to be kept.

A "null" name is one with a length of 0 (i.e., the subfield consists of only a length and key) or one which has no significant characters (all blanks).

Processing of MS Capabilities Request From Back-Level Focal Point:

Focal point implementations which are not at the current level of the architecture do not include the Focal Point Identification (X'21') subvector with the Focal Point Authorization Request (X'61') subvector. An MS Capabilities major vector from such a focal point implementation can be recognized because the Primary Authorization Request (X'10') subfield in the Focal Point Authorization Request (X'61') subvector will contain the value X'0000' in bytes 2-3. If this is present, the following values are *assumed* in lieu of the Focal Point Identification (X'21') subvector information:

- Focal point network ID -- network ID of sender
- Focal point NAU -- NAU name of sender
- Focal point MS application program -- X'23F0F3F1' (ALERT_NETOP)

Table 9-31 illustrates the format of the MS Capabilities major vector received from a back-level primary focal point.

Table 9-31. Contents of MS Capabilities Request (X'61') - Primary Focal Point (Back-level)

Structure Name	MV/SV/SF	Description of Contents
MS Capabilities	X'80F0'	
Focal Point Auth. Request	X'61'	
Primary Authorization Request	X'10'	
Back-level category	--	X'0000' Category = X'23F0F3F1' X'FFFF' Category &ne. X'23F0F3F1'

The Focal Point Authorization Request (X'61') subvector request is then processed as if it had included the Focal Point Identification (X'21') subvector (it is not rejected with sense data X'086C 2100').

If the Focal Point Authorization Request (X'61') subvector did not originate from a back-level focal point, the absence of Focal Point Identification (X'21') subvector is an error and the request is rejected.

Determining the Response to the MS Capabilities Request:

Once the request has been parsed, this process determines how to respond to the request. It passes the results as shown below to the "Sending MS Capabilities Replies" process which will build and send the reply.

If the category is not supported it will reject the request passing:

- RESPONSE=reject
- REASON=X'01' (function not supported)
- A copy of the original request

If this category is supported it checks to see if there is focal point information in the MS Capabilities table for this category.

- If there is no focal point already in the MS Capabilities table for this category, it will accept the request passing:

  - RESPONSE=accept
  - A copy of the original request

  It then updates the table with the new focal point information from the request, including filling in the routing information as 'direct' since this node is a SOC node, and starts the "Notifying Application Programs of Focal Point Identity" in topic 2.9.3.4.3 and "Sending MS Capabilities Requests" in topic 2.9.3.5.3 processes.

- If there is focal point information in the table, it parses the request and determines whether to accept or reject it based on rank of the focal point in the request as compared to the rank of the focal point in the MS Capabilities table

  The priority (rank) which a node will accept and use a focal point are as follows (highest priority at the top of the list). This list indicates **how** the focal point authorization was obtained.

  Remember that *explicit* focal points are ones defined at the focal point, while *implicit* focal points are ones defined at the SOC node. Refer to "Properties of Focal Points" in topic 1.2.4.2 for a complete discussion of the topic.

Focal Point Rank List for SOC End Nodes

1. Explicit primary focal point
2. Implicit primary focal point
3. Backup focal point (see note 1)
4. Focal point identity by virtue of receipt of Focal Point Notification (X'E1') subvector (see note 2)

Notes:

1. The backup focal point (number 3 in the list) has the same rank whether it originally came from implicit definition or from a Focal Point Authorization Request (X'61') subvector primary request.

2. If an end node has a back-level network node server, it *assumes* that the network node server is its focal point for the category ALERT_NETOP. (Actually, the back-level network node server will forward Alerts received from end nodes to its own focal point.) In this case, the end nodes gives the back-level network node server the same priority as a focal point identified in a Focal Point Notification (X'E1') subvector.

° If the focal point in the MS Capabilities table has a higher rank than that of the focal point in the request, it will reject the request, passing:

- RESPONSE=reject
- REASON=X'02' (request from a focal point of lower rank)
- A copy of the original request

° If the focal point in the MS Capabilities table has an equal or lower rank than that of the focal point in the request, it will accept the request and revoke the services of the present focal point. It passes twice to "Sending MS Capabilities Replies."

The first is to revoke the services of the present focal point. It passes:

- RESPONSE=revoke
- REASON=X'60' (focal point receiving this request was replaced due to a request from another focal point)
- The MS Capabilities table information for the focal point being revoked (category, NETID.CPNAME, application program name)
- A copy of the original request

It then accepts this request, passing:

- RESPONSE=accept
- A copy of the original request

It then updates the MS Capabilities table with the new focal point information. It fills in the routing information as 'direct' since this node is a SOC node.

It then starts the "Notifying Application Programs of Focal Point Identity" in topic 2.9.3.4.3 and the "Sending MS Capabilities Requests" in topic 2.9.3.5.3 processes.

This process then terminates.

Sending MS Capabilities Replies:

This process is started by the Receiving MS Capabilities Requests process in response to the receipt of an MS Capabilities major vector which contains a Focal Point Authorization Request (X'61') subvector. It builds a Focal Point Authorization Reply (X'62') subvector and a Focal Point Identification (X'21') subvector from the information that it is given, completes the MS Capabilities (X'80F0') major vector and the MDS-MU and passes it across MS protocol boundary R to the MDS function set for sending.

The information it is given is the following:

° If the focal point authorization requested was accepted, it is passed:

- RESPONSE=accept
- A copy of the original request

This process will build the X'62' subvector with the X'10' subfield to

indicate acceptance, and get the rest of the information it needs to complete the MS Capabilities major vector from the original request. Table 9-32 illustrates the format of the MS Capabilities major vector when the request is accepted. This process reverses the origin and destination fields in the MDS header, uses the same unit-of-work correlator, and marks this a reply.

Table 9-32. Contents of MS Capabilities Reply (X'62') - Entry Point Accepts Request

Structure Name	MV/SV/SF	Description of Contents
MS Capabilities	X'80F0'	
Focal Point Auth. Reply	X'62'	
Authorization Request Accepted	X'10'	
Back-level category	--	X'0000' Category = X'23F0F3F1' X'FFFF' Category &ne. X'23F0F3F1'
Focal Point Identification	X'21'	Identifies the current focal point (note 1)
MS Category	X'01'	Category name
Focal Point Identification Flags	X'02'	All flags 0
Focal Point NETID	X'10'	Network ID of focal point
Focal Point NAU Name	X'11'	NAU name of focal point
Focal Point Appl. program ID	X'12'	MS appl. program name of focal point
Backup Focal Point NETID	X'20'	Network ID of backup focal point (note 2)
Backup Focal Point NAU Name	X'21'	NAU name of backup focal point (note 2)
Backup Focal Point Appl. prog. ID	X'22'	MS appl. program name of backup focal point (note 2)

Note:

1. Focal Point Identification (X'21') subvector always identifies the *current* focal point. For an accept reply, the current focal point is the one receiving the reply, so the X'21' SV in the reply is an echo of one in the request.

2. The subfields identifying a backup focal point are present in the reply only if they were present on the request.

° If the focal point authorization was rejected, it is passed:

- RESPONSE=reject
- REASON=X'xx', where xx is the reason code for the rejection
- A copy of the original request

This process will build the X'62' subvector with the X'20' subfield to indicate rejection, and get the rest of the information it needs to complete the MS Capabilities major vector from the original request. Table 9-33 illustrates the format of the MS Capabilities major vector when the request is rejected. This process reverses the origin and destination fields in the MDS header, uses the same unit-of-work correlator, and marks this a reply.

Table 9-33. Contents of MS Capabilities Reply (X'62') - Entry Point Rejects Request

Structure Name	MV/SV/SF	Description of Contents
MS Capabilities	X'80F0'	
Focal Point Auth. Reply	X'62'	
Authorization Rejected	X'20'	
Back-level category	--	X'0000' Category = X'23F0F3F1' X'FFFF' Category &ne. X'23F0F3F1'
Authorization status code	--	X'01' Function not supported X'02' Sender already has a focal point of higher rank X'03' Sender is an end node which

		does not support receipt of X'61'
Current Focal Point CP Name	X'40'	NETID.CPNAME of current focal point (note 2)
Current Focal Point Appl Name	X'41'	MS appl. program name of current focal point (note 2)
Focal Point Identification	X'21'	Identifies the current focal point (note 1)
MS Category	X'01'	Category name
Focal Point Identification Flags	X'02'	All flags 0
Focal Point NETID	X'10'	Network ID of focal point
Focal Point NAU Name	X'11'	NAU name of focal point
Focal Point Appl. program ID	X'12'	MS appl. program name of focal point

Note:

1. Focal Point Identification (X'21') subvector always identifies the *current* focal point. For a reject, the current focal point is the one to which the entry point still reports, *not* the one receiving the reply.

2. The Current Focal Point subfields are present only for reject reason code X'02', which indicates that the request came from a focal point of lower rank than the current focal point.

° If the focal point authorization process causes a prior focal point to be revoked, it is passed:

- RESPONSE=revoke
- REASON=X'xx', where xx is the reason code for the revocation
- The MS Capabilities table information for the focal point being revoked (category, NETID.CPNAME, application program name)
- A copy of the original request

This process will build the X'62' subvector with the X'30' subfield to indicate revocation. It takes focal point information from the original request to fill in the X'40' and X'41' subfields, (current focal point). Table 9-34 illustrates the format of the MS Capabilities major vector for revoking a focal point. The destination information is that which is passed to it, and the origin is the destination field from the original request. It needs to get a new unit-of-work correlator, and mark this as a request without reply.

Table 9-34. Contents of MS Capabilities Reply (X'62') - Revoke

Structure Name	**MV/SV/SF**	**Description of Contents**
MS Capabilities	X'80F0'	
Focal Point Auth. Reply	X'62'	
Authorization Revoked	X'30'	
Back-level category	--	X'0000' Category = X'23F0F3F1' X'FFFF' Category &ne. X'23F0F3F1'
Authorization status code	--	X'60' Authorization revoked (new FP) X'61' Authorization revoked (operator command)
Current Focal Point CP Name	X'40'	NETID.CPNAME of current focal point (note 2)
Current Focal Point Appl Name	X'41'	MS appl. program name of current focal point (note 2)
Focal Point Identification	X'21'	Identifies the current focal point
MS Category	X'01'	Category name
Focal Point Identification Flags	X'02'	All flags 0
Focal Point NETID	X'10'	Network ID of focal point
Focal Point NAU Name	X'11'	NAU name of focal point
Focal Point Appl. program ID	X'12'	MS appl. program name of focal point

Note:

1. Focal Point Identification (X'21') subvector always identifies the *current* focal point. For a revoke (with authorization status code X'60'), the current focal point is the one to which the entry point now reports, *not* the one receiving the revoke. For authorization status code X'61' (focal point revoked because of a command from an operator), the entry point does not have a new focal point, so the subfields of Focal Point Identification (X'21') subvector will be empty (but they must be present).

2. The Current Focal Point subfields are present only for authorization status code X'60', which indicates that the receiver has been revoked because another focal point has taken its place.

Processing MDS Error Message Information:

This process also receives MDS error message information via protocol boundary N (described in "Protocol Boundary N--MDS Error Message Information" in topic 2.9.1.10). MDS passes a copy of every MDS error message that it processes for which the origin application program name is X'23F0F1F0' (MDS_ROUTER) to MS_CAPS. This process determines if the MDS error message impacts any of the focal points in the MS Capabilities table.

The following exceptions are made:

- When the sense data in the MDS error message is X'08A9 0007' (MS application router re-initialization MDS transaction exception). In this case no MS_CAPS recovery action is required.

- When the sense data in the MDS error message is X'0000 0000'. This is passed by MDS to notify MS_CAPS of the normal UNBIND of the last SNASVCMG session to a session partner when there are no impacted MDS tranactions (i.e., no actual MDS error messages generated). This is not an error situation, since the SNASVCMG sessions may be deactivated by the LU when they are not in use to conserve resources. If SNASVCMG sessions are deactivated because of network connectivity failures, MDS will pass the non-zero UNBIND sense data, and MS_CAPS will act accordingly (following steps).

If this is not an exception case, the focal point names (NETID, NAU, and MS application program) in the MS Capabilities table are searched for every match on the Reported on location (NETID and NAU) from the signal:

If a match is found on location, the MS application program name from the signal is examined:

- If the reported on MS application program is X'23F0F1F1' (MS_CAPS), then all MS application programs in the reported on node, including the focal point application program in the MS Capabilities table, are impacted. The reported on application program is set to MS_CAPS when the MDS_ROUTER in this node reports on an SNASVCMG session outage.

- If the reported on MS application program is some other value, then it is compared to the focal point application program name in the MS Capabilities table entry.

If a match is found, this process does the following:

- If the process "Changing to a Backup Focal Point" in topic 2.9.3.5.3 exists on this node, it is passed the MDS error message information signal.

- If that process does not exist, this process updates the MS Capabilities table. The focal point node is removed from the table, and this process starts the process described in "Notifying Application Programs of Focal Point Identity" in topic 2.9.3.4.3, passing it the category for which the MS Capabilities table was just updated.

- The process for triggering retries for the implicit primary focal point, if this node has one, is started. This process, which is not defined by the architecture, will trigger periodic attempts to reacquire the implicit primary focal point.

This process then terminates.

The processing of focal point information by a SOC end node is summarized in finite-state machine format in "MS_CAPS FSM: Focal Point Status for SOC End Node."

For a description of FSM notation, see "FSM Notation" in topic E.1.

MS_CAPS FSM: Focal Point Status for SOC End Node

Function:	This finite-state machine describes the processing of a single row in the MS Capabilities table. The only processing described is that associated with the maintenance of the focal point information and the receipt and sending of MS Capabilities messages. Not included is the maintenance of local MS application program program status for each category.

Inputs:	1. Implicit focal point definitions read from nonvolatile storage (initialization). 2. Focal Point Authorization Request (X'61') subvector (primary) received 3. Entry Point Authorization Reply (X'64') subvector (accept) received 4. Entry Point Authorization Reply (X'64') subvector (reject) received 5. Local operator inputs new implicit information which takes priority over the current focal point, regardless of its rank. 6. Local operator requests that the current focal point be revoked without acquiring a new focal point. (It is optional for entry point implementations to support this action.) 7. MS Capabilities retry trigger An implementation-defined protocol machine determines that an attempt to sending MS Capabilities to establish an implicit focal point relationship. 8. "Protocol Boundary N--MDS Error Message Information" in topic 2.9.1.10 The report code (sense data) in the MDS error message information is examined: a. X'0000 0000' No error has occurred, just a normal session UNBIND. No further processing is done. b. X'08A9 0007' MDS application router reinitialized. The MDS_ROUTER at the focal point experienced some failure, but recovery has taken place. No entry point action is required. c. If the sense data is any other nonzero value, it indicates a communication failure with the focal point application program. The appropriate recovery action will be taken by MS_CAPS in the entry point for the affected category. If the reported on agent is MS_CAPS (X'23F0F1F1') in the focal point node, then all categories in the MS Capabilities table are examined for impacts. If the focal point for a category was on the indicated node, then error recovery is initiated. 9. Focal Point Notification (X'E1') subvector received from network node server

States:	01 Reset No focal point for category 02 Explicit -- active Focal Point Authorization Request (X'61') subvector has been received from focal point and Focal Point Authorization Reply (X'62') subvector (accept) has been sent. 03 Implicit -- pending active

```
|           |                                                          |
|           |     Entry Point Authorization Request (X'63') subvector  |
|           |     has been sent to the implicit focal point. Entry Point |
|           |     Authorization Reply (X'64') subvector not yet        |
|           |     received.                                            |
|           |                                                          |
|           | 04  Implicit -- active                                   |
|           |                                                          |
|           |     Entry Point Authorization Reply (X'64') subvector    |
|           |     (accept) has been received from implicit focal point. |
|           |                                                          |
|           | 05  Backup -- pending active                             |
|           |                                                          |
|           |     Entry Point Authorization Request (X'63') subvector  |
|           |     has been sent to the backup focal point. Entry Point |
|           |     Authorization Reply (X'64') subvector not yet        |
|           |     received.                                            |
|           |                                                          |
|           | 06  Backup -- active                                     |
|           |                                                          |
|           |     Entry Point Authorization Reply (X'64') subvector    |
|           |     (accept) has been received from backup focal point.  |
|           |                                                          |
|           | 07  Domain focal point -- active                         |
|           |                                                          |
|           |     The SOC end node is using the focal point it learned |
|           |     about from a Focal Point Notification (X'E1')        |
|           |     subvector received from its network node server.     |
|___________|__________________________________________________________|
|                                                                      |
|______________________________________________________________________|
```

	States						
Inputs	**Reset** 01	**Explicit active** 02	**Implicit pending active** 03	**Implicit active** 04	**Backup pending active** 05	**Backup active** 06	**Domain FP active** 07
Implicit names input from nonvolatile storage at node initialization	03f	/	/	/	/	/	/
SV X'61' (primary) received	02ad	02ad	02ad	02ad	02ad	02ad	02a
SV X'64' (accept) received	/	/	04id	/	06id	/	/
SV X'64' (reject) received - backup FP defined - domain FP defined	/	/	05g	/	07i	/	/
SV X'64' (reject) received - backup FP not defined - domain FP defined	/	/	07i	/	07i	/	/
SV X'64' (reject) received - backup FP not defined - domain FP not defined	/	/	01i	/	01i	/	/
Operator Input: New implicit FP	03f	03f	03f	03f	03f	03f	03f
Operator Input: Revoke current FP	01	01d	01d	01d	01d	01d	01
MS Capabilities retry trigger - implicit primary defined	03fj	/	03j	/	03fj	03fj	03fj
MS Capabilities retry trigger - implicit primary not defined - backup defined	05gj	/	03j	/	05j	06j	05g
MDS Error Msg Info (PB N) -Backup FP defined -Domain FP defined	/	05gh	05g	05gh	07i	07i	01i
MDS Error Msg Info (PB N) -Backup FP not defined -Domain FP defined	/	07ih	07i	07ih	/	/	01i
MDS Error Msg Info (PB N) -Backup FP not defined -Domain FP not defined	/	01ih	01i	01ih	/	/	/
SV X'E1' from network node server	07i	02e	03e	04e	05e	06e	07i

Output Code	Function
a	1. Focal Point Authorization Reply (X'62') subvector (accept) sent. 2. MS Capabilities recovery responsibility set to FP or EP (based on setting of MS Capabilities Recovery Responsibility indicator (byte 4 bit 0) in Focal Point Authorization Request (X'61') subvector) 3. Primary and backup FP information updated. 4. Implicit focal point definitions updated on nonvolatile storage if MS Capabilities recovery responsibility = EP. 5. Focal Point Notification (X'E1') subvector sent to local application programs.
b	1. Focal Point Authorization Reply (X'62') subvector (accept) sent. 2. MS Capabilities recovery responsibility set to FP. 3. Primary and backup FP information updated. 4. Focal Point Notification (X'E1') subvector sent to local application programs.
c	Focal Point Authorization Reply (X'62') subvector (reject) sent
d	Focal Point Authorization Reply (X'62') subvector (revoke) sent to previous active FP (if defined). The authorization status code used is: X'60' If a new FP has been acquired. X'61' If the revocation was requested by an operator.
e	Domain FP info updated
f	1. Entry Point Authorization Request (X'63') subvector (implicit) sent. 2. Primary FP information updated.
g	1. Entry Point Authorization Request (X'63') subvector (backup) sent 2. Backup FP information updated.
h	Retry mechanism (UPM) initiated for primary
i	1. FP information updated. 2. Focal Point Notification (X'E1') subvector sent to local application programs (may carry names with null values if FP is not active).
j	Retry mechanism (UPM) stopped

| 2.9.3.7 MS_CAPS Optional Subset 3 (Base Network Node Support)

Subtopics:

- 2.9.3.7.1 Functions Provided
- 2.9.3.7.2 Formats Supported
- 2.9.3.7.3 Implementation Requirements

| 2.9.3.7.1 Functions Provided

This is the MS Capabilities support needed for a base network node. This function set provides support for being a node in a sphere of control, and sending and receiving the MS Capabilities major vectors that are used to establish that relationship from the entry point side of the relationship. The Focal Point Authorization Request (X'61') subvector and Focal Point Authorization Reply (X'62') subvector processing are described in "MS_CAPS Optional Subset 2 (Be a Sphere-of-Control End Node)" in topic 2.9.3.6. In addition to that support, this optional subset include notifying client end nodes of the focal point identification.

| 2.9.3.7.2 Formats Supported

```
| Focal Point                        Network Node                        End Node
```

```
|   .                                  .                                   .
|   .   MS Caps[FP Auth. Request (X'61')]   .                             .
|   ____________________________________ÿ                                  .
|   .    MS Caps[FP Auth. Reply (X'62')]    .                             .
```

```
|   ____________________________________                                  .
|   .                                  .   MS Caps[FP Notification (X'E1')]    .
|   .                                  ____________________________________ÿ
|   .                                  .                                    .
```

| Figure 9-93. MS Capabilities Flows Supported by MS_CAPS Optional Subset 3

| This subset supports the receipt of the following management services
| major vector and subvectors in MDS-MUs:

| ° MS Capabilities (X'80F0') major vector
| ° Focal Point Authorization Request (X'61') subvector
| ° Focal Point Identification (X'21') subvector

| It also allows the sending of the following management services major
| vector and subvectors in MDS-MUs:

| ° MS Capabilities (X'80F0') major vector
| ° Focal Point Authorization Reply (X'62') subvector
| ° Focal Point Notification (X'E1') subvector
| ° Focal Point Identification (X'21') subvector

| The MS Capabilities major vector flows supported by this subset are
| illustrated in Figure 9-93.

| 2.9.3.7.3 Implementation Requirements

| The requirements for implementing the MS Capabilities optional subset 3
| are described by a model consisting of CP Management Services. "MS_CAPS
| Optional Subset 1 (Have a Backup or Implicit FP)" in topic 2.9.3.5 is a
| prerequisite to this subset.

| **A Subset of CP Management Services**

| ° In addition to the functions provided in MS_CAPS subsets prerequisite
| to this subset, and to the functions described in "MS_CAPS Optional
| Subset 2 (Be a Sphere-of-Control End Node)" in topic 2.9.3.6, this
| subset provides the following processes:

| - Sending MS Capabilities Requests

| -- Builds MDS-MUs containing a MS Capabilities major vector with
| the Focal Point Notification (X'E1') subvector (and
| accompanying subvectors) for sending to client end nodes.

| -- Passes these MDS-MUs to MDS for sending.

| ° This process provides additional function in the following processes:

| - Receiving CP Status Updates (Change in addition to function in
| "Receiving CP Status Updates" in topic 2.9.3.4.3)

| -- Processes CP Status messages received from MDS and adds or
| removes names of client end nodes in the MS Capabilities
| table.

| - Processing MDS Error Message Information (Change in addition to
| function in "Processing MDS Error Message Information" in
| topic 2.9.3.6.3)

| -- Notifies client end nodes when a focal point is lost.

| - Receiving MS Capabilities Requests (Changes in addition to
| function in "Receiving MS Capabilities Requests" in
| topic 2.9.3.6.3)

| Accepts MS Capabilities major vector from a default focal point.

| *Receiving MS Capabilities Requests*:

| In addition to the function described for a SOC end node in "Receiving MS
| Capabilities Requests" in topic 2.9.3.6.3, this optional subset must
| accept MS Capabilities major vector from a *default* focal point.
| Table 9-35 illustrates the format of the MS Capabilities major vector
| received from a default focal point.

Table 9-35. Contents of MS Capabilities Request (X'61') - Default Focal Point

Structure Name	MV/SV/SF	Description of Contents
MS Capabilities	X'80F0'	
Focal Point Auth. Request	X'61'	
Secondary Authorization Request	X'20'	
Back-level category	--	X'0000' Category = X'23F0F3F1' X'FFFF' Category &ne. X'23F0F3F1'
Focal Point Identification	X'21'	Identifies the current focal point (note 1)
MS Category	X'01'	Category name
Focal Point Identification Flags	X'02'	All flags 0
Focal Point NETID	X'10'	Network ID of focal point
Focal Point NAU Name	X'11'	NAU name of focal point
Focal Point Appl. program ID	X'12'	MS appl. program name of focal point
Backup Focal Point NETID	X'20'	Network ID of backup focal point (note 2)
Backup Focal Point NAU Name	X'21'	NAU name of backup focal point (note 2)
Backup Focal Point Appl. prog. ID	X'22'	MS appl. program name of backup focal point (note 2)

Note:

1. Focal Point Identification (X'21') subvector always identifies the *current* focal point. For a request, the current focal point is the one sending MS Capabilities.

2. The subfields identifying a backup focal point are present in the request only if a backup is defined for the category.

Table 9-36 illustrates the format of the MS Capabilities major vector received from a back-level default focal point.

Table 9-36. Contents of MS Capabilities Request (X'61') - Secondary Focal Point (Back-level)

Structure Name	MV/SV/SF	Description of Contents
MS Capabilities	X'80F0'	
Focal Point Auth. Request	X'61'	
Secondary Authorization Request	X'20'	
Back-level category	--	X'0000' Category = X'23F0F3F1' X'FFFF' Category &ne. X'23F0F3F1'

Supporting a default focal point means that MS_CAPS supports an expanded focal point rank list:

Focal Point Rank List for Network Nodes

1. Explicit primary
2. Implicit primary
3. Backup (implicit or explicit, see note 1)
4. Default primary
5. Backup (default, see note 2)
6. Pseudo focal point (if that option is supported)

Notes:

1. The backup focal point (number 3 in the list) has the same rank whether it originally came from implicit definition or from a Focal Point Authorization Request (X'61') subvector primary request.

2. A backup focal point (number 5 in the list) which is provided by a default focal point in a Focal Point Authorization Request (X'61') subvector secondary request has a lower rank than a backup which comes from implicit definition or from a Focal Point Authorization

Request (X'61') subvector primary request.

Sending MS Capabilities Requests:

This process is started by several processes.

- If it is started by the "Receiving CP Status Updates" process, it is passed the name of the new client end node. It then builds focal point information to that client end node.

- If it is started by the "Receiving MS Capabilities Requests" in topic 2.9.3.6.3 process, it is passed the category to which the new focal point information applies. It then builds focal point information to all client end nodes.

- If it is started by the "MDS Error Message Processing" in topic 2.9.3.4.3 process, it is passed the category to which the new focal point information applies. It then builds focal point information to all client end nodes.

The building and sending proceeds as follows. It builds an MDS-MU with a MS Capabilities major vector containing an Focal Point Notification (X'E1') subvector and a Focal Point Identification (X'21') subvector, getting its focal point information from the MS Capabilities table. If the MS Capabilities table for a category it is passed is empty, it builds empty focal point identification subfields. This will indicate that there currently is no focal point. It passes the completed MDS-MU across protocol boundary R (described in "Protocol Boundary R--Send MDS-MU" in topic 2.9.1.13) for sending.

Empty Focal Point Identification Subfields:

The loss of a focal point is reported by the creation of MS Capabilities (X'80F0') major vector with an Focal Point Notification (X'E1') subvector and a Focal Point Identification (X'21') subvector with the following subfield contents:

- MS Category (X'01') - category name
- Focal Point Identification Flags (X'02')
- Focal Point NETID (X'10') - null
- Focal Point NAU Name (X'11') - null
- Focal Point Application program ID (X'12') - null

"null" name is one with a length of 0 (i.e., the subfield consists of only a length and key) or one that has no significant characters (all space characters).

The MS Capabilities table maintained by a network node is illustrated in Table 9-37.

Table 9-37. MS Capabilities Table for Network Nodes

Category	Primary Focal Point					Backup Focal Point			Pseudo FP	Local Appls
	Stat.	Recov.	Rout.	Rank	Focal Point Name	Stat.	Rout.	Focal Point Name	Stat.	
X'23F0F3F1'	act.	FP	dir.	expl.	NET1 NODE1 X'23F0F3F1'	inact.	dir.	NET1 NODE2 X'23F0F3F1'	inact.	X'23F0F3F0' USERAP1
X'23F0F1F7'	act.	FP	dir.	expl.	NET1 NODE1 X'23F0F1F7'	inact.	dir.	NET1 NODE2 X'23F0F1F7'	inact.	X'23F0F1F6' USERAP4
USERCAT	act.	EP	dir.	impl.	NET1 NODE1 USERAP2	inact.	dir.		N/A	USERAP3

Note:

X'23F0F3F1' = ALERT_NETOP
X'23F0F3F0' = EP_ALERT
X'23F0F1F7' = OPERATIONS_MGMT_NETOP
X'23F0F1F6' = EP_OPERATIONS_MGMT

Table 9-38. Description of MS Capabilities Table Information. Each row of the MS Capabilities table contains the information described below.

Category

The name of the MS category associated with the focal points and application programs in this row. A row in the MS Capabilities table created when an MS application program program registers for a category not already listed in the table, or when a focal point relationship is established for a category not already in the table.

Primary Focal Point Information

- Status

  - Inactive
  - Active

- MS Capabilities recovery responsibility

  - FP (focal point will initiate recovery by sending MS Capabilities)
  - EP (entry point will initiate recovery by sending MS Capabilities)

  This value is based upon the setting of the MS Capabilities Recovery Responsibility indicator (byte 4 bit 0) in Focal Point Authorization Request (X'61') subvector received from an explicit focal point. It is set to EP for an implicit focal point.

- Routing

  - DEFAULT

    All traffic for this focal point will be routed using the *default* routing option of MULTIPLE_DOMAIN_SUPPORT. This value applies only to a domain focal point used by an end node.

  - DIRECT

    All traffic for this focal point will be routed using the *direct* routing option of MULTIPLE_DOMAIN_SUPPORT.

- Rank

  This field actually contains the focal point type:

  1. Explicit
  2. Implicit
  3. Default

  Rank refers to the preference given to a focal point of one type of a focal point of another type. Since network nodes support support *default* focal points, the focal point rank list is longer than that supported by SOC end nodes:

  1. Explicit primary

  2. Implicit primary

  3. Backup (implicit or explicit, see note)

  4. Default primary

  5. Backup (default, see note)

  6. Pseudo focal point (if that option is supported)

  Note:

  The backup focal point (number 3 in the list) has the same rank whether it originally came from implicit definition or from a Focal Point Authorization Request (X'61') subvector primary request.

  A backup focal point (number 5 in the list) which is provided by a default focal point in a Focal Point Authorization Request (X'61') subvector secondary request has a lower rank than a backup which comes from implicit definition or from a Focal Point Authorization Request (X'61') subvector primary request.

  The rank is used to determine whether or not the entry point will accept a particular Focal Point Authorization Request (X'61') subvector request. If the rank of the focal point sending the request is lower than the current focal point, the entry point will reject the request. If it is equal or higher in rank, the request will be accepted and the previous focal point revoked.

  If a local operator at the entry point overrides the current focal point with new implicit imformation, that new focal point is accepted regardless of the ranking list.

- Focal Point Name

  - NETID name
  - NAU name
  - MS application program name

Backup Focal Point Information

- Status

  - Inactive
  - Active

```
| | °   Routing                                                          |
|                                                                        |
| |     -   DIRECT                                                       |
|                                                                        |
| | °   Focal Point Name                                                 |
|                                                                        |
| |     -   NETID name                                                   |
| |     -   NAU name                                                     |
| |     -   MS application program name                                  |
|  ______________________________________________________________________|
| | Pseudo Focal Point Information                                       |
|                                                                        |
| | °   Status                                                           |
|                                                                        |
| |     -   Inactive                                                     |
| |     -   Active                                                       |
|  ______________________________________________________________________|
| | Local application programs                                           |
|                                                                        |
| | A list of local MS application programs that have registered with    |
| | MS_CAPS for focal point information about this category.             |
|  ______________________________________________________________________|
```

| *Receiving CP Status Updates*:

| In addition to the functions provided by "Receiving CP Status Updates" in topic 2.9.3.4.3, this process will parse CP status messages passed to it from the MDS component, and will keep track of client end nodes. It adds or deletes them in the MS Capabilities table as indicated. If it receives information about a new client end node (i.e., it adds one to the table) it starts the "Sending MS Capabilities Requests" process, passing it the end node name.

| *Processing MDS Error Message Information*:

| In addition to the processing described in "Processing MDS Error Message Information" in topic 2.9.3.6.3, the following processing is done.

| When a focal point is lost, in addition to notifying local application programs, the process "Sending MS Capabilities Requests" is started, and passed the category for which there was a change in focal point information so that client end nodes will be notified.

| The processing of focal point information by a network node is summarized in finite-state machine format in "MS_CAPS FSM: Focal Point Status for Network Node."

| *MS_CAPS FSM: Focal Point Status for Network Node*

```
  ____________________________________________________________________
 |____________ ______________________________________________________|
| | Function: | This finite-state machine describes the processing of a |
| |           | single row in the MS Capabilities table.  The only      |
| |           | processing described is that associated with the        |
| |           | maintenance of the focal point information and the receipt |
| |           | and sending of MS Capabilities messages.  Not included is |
| |           | the maintenance of local MS application program program |
| |           | status for each category and the maintenance of the list |
| |           | of client end nodes for this network node.              |
|  ___________|_____________________________________________________|
|  ___________________________________________________________________|
 |____________ ______________________________________________________|
| | Inputs:   | 1.  Implicit focal point definitions read from nonvolatile |
| |           |     storage (initialization).                           |
| |           |                                                         |
| |           | 2.  Focal Point Authorization Request (X'61') subvector |
| |           |     (primary) received                                  |
| |           |                                                         |
| |           | 3.  Focal Point Authorization Request (X'61') subvector |
| |           |     (secondary) received                                |
| |           |                                                         |
| |           | 4.  Entry Point Authorization Reply (X'64') subvector   |
| |           |     (accept) received                                   |
| |           |                                                         |
| |           | 5.  Entry Point Authorization Reply (X'64') subvector   |
| |           |     (reject) received                                   |
| |           |                                                         |
| |           | 6.  Local operator inputs new implicit information which |
| |           |     takes priority over the current focal point,        |
| |           |     regardless of its rank.                             |
| |           |                                                         |
| |           | 7.  Local operator requests that the current focal point |
| |           |     be revoked without acquiring a new focal point.  (It |
| |           |     is optional for entry point implementations to support |
| |           |     this action.)                                       |
|            |                                                         |
```

```
| |           | 8.  MS Capabilities retry trigger                          |
| |           |                                                            |
| |           |     An implementation-defined protocol machine determines  |
| |           |     that an attempt to sending MS Capabilities to          |
| |           |     establish an implicit focal point relationship.        |
| |           |                                                            |
| |           | 9.  "Protocol Boundary N--MDS Error Message Information"   |
| |           |     in topic 2.9.1.10                                      |
| |           |                                                            |
| |           |     The report code (sense data) in the MDS error message  |
| |           |     information is examined:                               |
| |           |                                                            |
| |           |     a.  X'0000 0000'                                       |
| |           |                                                            |
| |           |         No error has occurred, just a normal session       |
| |           |         UNBIND.  No further processing is done.            |
| |           |                                                            |
| |           |     b.  X'08A9 0007'                                       |
| |           |                                                            |
| |           |         MDS application router reinitialized.  The         |
| |           |         MDS_ROUTER at the focal point experienced some     |
| |           |         failure, but recovery has taken place. No entry    |
| |           |         point action is required.                          |
| |           |                                                            |
| |           |     c.  If the sense data is any other nonzero value, it   |
| |           |         indicates a communication failure with the focal   |
| |           |         point application program.  The appropriate       |
| |           |         recovery action will be taken by MS_CAPS in the    |
| |           |         entry point for the affected category.  If the     |
| |           |         reported on agent is MS_CAPS (X'23F0F1F1') in the  |
| |           |         focal point node, then all categories in the MS    |
| |           |         Capabilities table are examined for impacts.  If   |
| |           |         the focal point for a category was on the          |
| |           |         indicated node, then error recovery is initiated.  |
| |           |                                                            |
| |           | 10. ACTPU received from SSCP. (Pseudo focal point support  |
| |           |     only)                                                  |
| |           |                                                            |
| |           | 11. DACTPU received from SSCP. (Pseudo focal point support |
| |           |     only)                                                  |
|___________|____________________________________________________________|
|                                                                        |
|________________________________________________________________________|
|___________|____________________________________________________________|
| | States:   | 01  Reset                                                  |
| |           |                                                            |
| |           |     No focal point for category                            |
| |           |                                                            |
| |           | 02  Explicit -- active                                     |
| |           |                                                            |
| |           |     Focal Point Authorization Request (X'61') subvector    |
| |           |     has been received from focal point and Focal Point     |
| |           |     Authorization Reply (X'62') subvector (accept) has     |
| |           |     been sent.                                             |
| |           |                                                            |
| |           | 03  Implicit -- pending active                             |
| |           |                                                            |
| |           |     Entry Point Authorization Request (X'63') subvector    |
| |           |     has been sent to the implicit focal point. Entry Point |
| |           |     Authorization Reply (X'64') subvector not yet          |
| |           |     received.                                              |
| |           |                                                            |
| |           | 04  Implicit -- active                                     |
| |           |                                                            |
| |           |     Entry Point Authorization Reply (X'64') subvector      |
| |           |     (accept) has been received from implicit focal point.  |
| |           |                                                            |
| |           | 05  Backup -- pending active                               |
| |           |                                                            |
| |           |     Entry Point Authorization Request (X'63') subvector    |
| |           |     has been sent to the backup focal point. Entry Point   |
| |           |     Authorization Reply (X'64') subvector not yet          |
| |           |     received.                                              |
| |           |                                                            |
| |           | 06  Backup -- active                                       |
| |           |                                                            |
| |           |     Entry Point Authorization Reply (X'64') subvector      |
| |           |     (accept) has been received from backup focal point.    |
| |           |                                                            |
| |           | 07  Default -- active                                      |
| |           |                                                            |
| |           |     Focal Point Authorization Request (X'61') subvector    |
| |           |     (secondary) has been received from a default focal     |
| |           |     point and Focal Point Authorization Reply (X'62')      |
| |           |     subvector (accept) has been sent.                      |
| |           |                                                            |
| |           | 08  Pseudo focal point -- active                           |
| |           |                                                            |
| |           |     The network node server is using its SSCP as a focal   |
| |           |     point for the category. This state applies only to     |
| |           |     pseudo focal point support and then only for the       |
| |           |     category ALERT_NETOP.                                  |
|___________|____________________________________________________________|
|                                                                        |
|________________________________________________________________________|
```

	States							
Inputs	Reset 01	Explicit active 02	Implicit pending active 03	Implicit active 04	Backup pending active 05	Backup active 06	Default active 07	Pseudo FP active 08
Implicit names input from nonvolatile storage at node initialization	03f	/	/	/	/	/	/	/
SV X'61' (primary) received	02ad	02ad	02ad	02ad	02ad	02ad	02ad	02a
SV X'61' (secondary) received - backup FP known from implicit definition or from SV X'61' (primar)	07b	02c	03c	04c	05c	06c	07bd	07b
SV X'61' (secondary) received - backup FP known from previous SV X'61' (secondary)	07b	02c	03c	04c	07bd	07bd	07bd	07b
SV X'64' (accept) received	/	/	04id	/	06id	/	/	/
SV X'64' (reject) received - backup FP defined - domain FP defined	/	/	05g	/	08i	/	/	/
SV X'64' (reject) received - backup FP not defined - domain FP defined	/	/	08i	/	08i	/	/	/
SV X'64' (reject) received - backup FP not defined - domain FP not defined	/	/	01i	/	01i	/	/	/
Operator Input: New implicit FP	03f	03f	03f	03f	03f	03f	03f	03f
Operator Input: Revoke current FP	01	01d	01d	01d	01d	01d	01d	01
MS Capabilities retry trigger - implicit primary defined	03fj	/	03j	/	03fj	03fj	03fj	03fj
MS Capabilities retry trigger - implicit primary not defined - backup defined	05gj	/	03j	/	05j	06j	05g	05g
MDS Error Msg Info (PB N) -Backup FP defined -Domain FP defined	/	05gh	05g	05gh	08i	08i	08i	01i
MDS Error Msg Info (PB N) -Backup FP not defined -Domain FP defined	/	08ih	08i	08ih	/	/	08i	01i
MDS Error Msg Info (PB N) -Backup FP not defined -Domain FP not defined	/	01ih	01i	01ih	/	/	01i	/
ACTPU received from SSCP	08i	02e	03e	04e	05e	06e	07e	08i
DACTPU received from SSCP	01i	02e	03e	04e	05e	06e	07e	01i

Output Code	Function
a	1. Focal Point Authorization Reply (X'62') subvector (accept) sent. 2. MS Capabilities recovery responsibility set to FP or EP (based on setting of MS Capabilities Recovery Responsibility indicator (byte 4 bit 0) in Focal Point Authorization Request (X'61') subvector) 3. Primary and backup FP information updated. 4. Implicit focal point definitions updated on nonvolatile storage if MS Capabilities recovery responsibility = EP. 5. Focal Point Notification (X'E1') subvector sent to local application programs. 6. Focal Point Notification (X'E1') subvector sent to client end nodes.
b	1. Focal Point Authorization Reply (X'62') subvector (accept) sent. 2. MS Capabilities recovery responsibility set to FP. 3. Primary and backup FP information updated. 4. Focal Point Notification (X'E1') subvector sent to local application programs. 5. Focal Point Notification (X'E1') subvector sent to client end nodes.

c	Focal Point Authorization Reply (X'62') subvector (reject) sent
d	Focal Point Authorization Reply (X'62') subvector (revoke) sent to previous FP. The authorization status code used is: X'60' If a new FP has been acquired. X'61' If the revocation was requested by an operator.
e	Domain FP info updated
f	1. Entry Point Authorization Request (X'63') subvector (implicit) sent. 2. Primary FP information updated.
g	1. Entry Point Authorization Request (X'63') subvector (backup) sent 2. Backup FP information updated.
h	Retry mechanism (UPM) initiated for primary
i	1. FP information updated. 2. Focal Point Notification (X'E1') subvector sent to local application programs (may carry names with null values if FP is not active). 3. Focal Point Notification (X'E1') subvector sent to client end nodes (may carry names with null values if FP is not active).
j	Retry mechanism (UPM) stopped

| 2.9.3.8 MS_CAPS Optional Subset 4 (Have a Subarea Focal Point)

Subtopics:

- 2.9.3.8.1 Functions Provided
- 2.9.3.8.2 Formats Supported
- 2.9.3.8.3 Implementation Requirements

| 2.9.3.8.1 Functions Provided

This subset provides the ability for a network node to support having a subarea focal point by letting the network node act as a pseudo focal point on behalf of its domain. It will forward data it receives as a pseudo focal point on an SSCP-PU session to a subarea focal point, after converting the CP-MSUs to NMVTs. This optional subset is valid only for the category ALERT_NETOP.

| 2.9.3.8.2 Formats Supported

```
Pseudo Focal                        Network Node
Point (SSCP)                      (acting as T2.0)
```

```
.                                   .                                   .
.                ACTPU              .                                   .
_____________________________________ÿ                                   .
.                +rsp               .                                   .
_____________________________________                                   .
.                                   .   MS Caps[FP Notification (X'E1')]  .
.                                   _____________________________________ÿ
.               DACTPU              .                                   .
_____________________________________ÿ                                   .
.                +rsp               .                                   .
_____________________________________                                   .
.                                   .                                   .
```

Figure 9-94. MS Capabilities Flow Supported by MS_CAPS Optional Subset 4

This subset supports the sending of the following management services major vector and subvectors in MDS-MUs:

- MS Capabilities (X'80F0') major vector
- Focal Point Notification (X'E1') subvector
- Focal Point Identification (X'21') subvector

| Figure 9-94 illustrates the concept of this subset.

| 2.9.3.8.3 Implementation Requirements

| The requirements for implementation of this subset are a subset of CP
| management services.

| **A Subset of CP Management Services**

| ° In addition to the functions provided in MS Capabilities subsets
| prerequisite to this subset, this subset provides the following
| processes:

| - Pseudo Focal Point Support

| -- Provides the function necessary for this network node to act
| as a pseudo focal point for its domain, in order to collect
| all the Alerts for forwarding to a subarea focal point.

| *Pseudo Focal Point Support*:

| This process is started when this node receives an ACTPU or a DACTPU,
| about its SSCP-PU session. When it receives an ACTPU it checks to see if
| it has a focal point entry for the ALERT_NETOP category. If it does not,
| then it fills in its own NETID.CPNAME for the ALERT_NETOP category, with
| the focal point type of "pseudo." The application program name is the name
| of the process in this node that does the CP-MSU/NMVT conversion.

| When this process receives a DACTPU, it checks the MS Capabilities table.
| If the focal point for the ALERT_NETOP category is of the type "pseudo,"
| it removes the entry from the MS Capabilities table for this category.
| This process must know the name of the process where the conversion is
| done, and it must be able to obtain knowledge of the existence of an
| SSCP-PU session.

| After filling in the table with this information, it starts the processes
| described in "Notifying Application Programs of Focal Point Identity" in
| topic 2.9.3.4.3 and "Sending MS Capabilities Requests" in topic 2.9.3.7.3
| to notify the application programs and the client end nodes where to send
| Alerts for forwarding (in the case of a received ACTPU) or that there is
| no longer a focal point (in the case of a received DACTPU).

2.9.3.9 Summary of MS Capabilities Parsing

Table 9-39 summarizes the parsing of the MS Capabilities requests and replies by MS_CAPS. The "Order of Checking" column indicates the required order for performing the checking of a received MS Capabilities. This table includes all format checks for the base and all the optional subsets, so not all checks are applicable to every implementation. The "Sense Data" column indicates the sense code used to report each format error. The "Condition" column summarizes the format check. The notes at the end of the table indicate any special considerations, as well as the mechanism to be used for reporting each error.

Table 9-39. Sense Data Returned by MS_Capabilities Parser

Order of Checking	Sense Data	Condition	Notes
1	X'080C 0013'	MULTIPLE_DOMAIN_SUPPORT message type not supported	1
2	X'080C 0012'	Application program GDS variable in MDS-MU not recognized	1
3	X'080C 0005'	Major vector key not supported	2
4	X'080C 0006'	Control subvector missing or not supported (X'61', X'62', X'63', X'64' or X'E1').	2
5	X'086F nn05'	The sum of the subfield lengths in the subvector is incompatible with the length of the subvector.	2,3
6	X'086F 0002'	The sum of the subvectors lengths in the major vector is incompatible	2

		with the length of the major vector.	
7	X'080C 0161'	Focal Point Authorization Request (X'61') subvector missing function subfield (X'10', X'20', X'30').	2
8	X'080C 0162'	Focal Point Authorization Reply (X'62') subvector missing function subfield (X'10', X'20', X'30').	2
9	X'086D 6240'	Focal Point Authorization Reply (X'62') subvector missing Current FP CP_name subfield (X'40') when required for the Authorization Request Rejected (X'20') subfield or the Authorization Revoked (X'30') subfield.	2
10	X'086D 6241'	Focal Point Authorization Reply (X'62') subvector missing Current FP application program subfield (X'41') when required for the Authorization Request Rejected (X'20') subfield or the Authorization Revoked (X'30') subfield.	2,4
11	X'080C 0163'	Entry Point Authorization Request (X'63') subvector missing the function subfield (X'10', X'30').	2
12	X'080C 0164'	Entry Point Authorization Reply (X'64') subvector missing the function subfield (X'10', X'20').	2
13	X'086C 2100'	Required Focal Point Identification (X'21') subvector not present. Already found either X'61', X'63' or X'E1' subvector.	2,4
14	X'086F 2105'	The sum of the subfield lengths in the Focal Point Identification (X'21') subvector incompatible with the length of the subvector.	2,4
15	X'086F 2106'	One or more of the subfields in Focal Point Identification (X'21') subvector has an invalid length field.	2, 4
16	X'086D 2101'	Focal Point Identification (X'21') subvector is missing the MS_Category subfield (X'01').	2,4
17	X'086D 2102'	Focal Point Identification (X'21') subvector is missing the Flags subfield (X'02').	2,4
18	X'086D 2110'	Focal Point Identification (X'21') subvector is missing the FP NETID subfield (X'10').	2, 4
19	X'086D 2111'	Focal Point Identification (X'21') subvector is missing the FP NAU name subfield (X'11').	2, 4
20	X'086D 2112'	Focal Point Identification (X'21') subvector is missing the FP application program name subfield (X'12').	2, 4
21	X'086D 2120'	Focal Point Identification (X'21') subvector is missing the Backup FP Netid subfield (X'20') when either the Backup FP NAU Name (X'21') subfield or the Backup FP Application program Name (X'22') subfield was found.	2, 4
22	X'086D 2121'	Focal Point Identification (X'21') subvector is missing the Backup FP NAU name subfield (X'21') when the Backup FP NETID (X'20') subfield is present.	2, 4
23	X'086D 2122'	Focal Point Identification (X'21') subvector is missing the Backup FP application program name (X'22') subfield when the Backup FP NETID (X'20') subfield and the Backup FP NAU Name (X'21') subfield are present.	2, 4
24	X'1005 0010'	Backup focal point flag is set to 1 **and** a backup focal point name is included.	2,4

Note:

1. The sense data is sent in the Sense Data (X'7D') subvector of the

```
|     SNA Condition Report (X'1532') GDS variable, which is sent as an   |
|     MDS error message.  A generic Alert is also required when any of   |
|     these errors is detected.                                          |
|                                                                        |
|                                                                        |
| 2.  The sense data is sent in the Sense Data (X'7D') subvector of the  |
|     SNA Condition Report (X'1532') GDS variable, which is placed in a  |
|     CP-MSU.  The MDS message type is "reply" if the received MS        |
|     Capabilities is a "request with reply" (this should be the case    |
|     for a received Focal Point Authorization Request (X'61') subvector |
|     or Entry Point Authorization Request (X'63') subvector).  The MDS  |
|     message type is "request without reply" if the received MS         |
|     Capabilities had that same message type (this should be the case   |
|     for a received Focal Point Authorization Reply (X'62') subvector   |
|     (revoke) or Focal Point Notification (X'E1') subvector). See        |
|     "Building Negative Replies for MDS-MUs" in topic 2.9.2.9.4 for     |
|     more information.                                                  |
|                                                                        |
|                                                                        |
| 3.  The only applicable values which may be substituted for nn in this |
|     sense code are X'61', X'62', X'63', and X'64'.                     |
|                                                                        |
|                                                                        |
| 4.  This check is not applicable to an MS Capabilities received from a |
|     back-level node.                                                   |
|                                                                        |
|                                                                        |
|________________________________________________________________________|
```

2.9.3.10 Application Program Registration with MS_CAPS

EP_XXXX application programs requiring service from XXXX_NETOP application programs in focal points must register themselves with MS_CAPS when they first become active.

An application program registers itself with MS_CAPS via internal MS protocol boundary T (Register MS Category). The details of this protocol boundary are described in "Protocol Boundary T--Register MS Category" in topic 2.9.1.15. The application program registering itself passes to MS_CAPS via this protocol boundary its own name and the name of the MS *category* for which it seeks to receive focal point information. (4) MS_CAPS always honors this request, since there is no prohibition against either (1) a single application program's registering for more than one MS category, or (2) two or more application programs' registering for the same MS category.

After it has registered itself with MS_CAPS, an application program will receive focal point information via internal MS protocol boundary V (Current Focal Point). The details of this protocol boundary are described in "Protocol Boundary V--Current Focal Point" in topic 2.9.1.17. Once it has gotten the information via protocol boundary V, an application program remembers its focal point's NAU and application program names until either (1) the application program receives new information from MS_CAPS via protocol boundary V, or (2) the application program is re-initialized.

(4) This protocol boundary is thus different from protocol boundary O, by which an application program passes its name to MDS. This difference reflects a difference in the information that MDS need to know to do their jobs. To route MDS-MUs to an application program, MDS need only know the application program's name, but not its management services category. To route focal point information to an application program, MS_CAPS needs to know the application program's management services category as well as its name.

2.9.4 FILE_SERVICES_SUPPORT Function Set

```
°°°
```

```
°                        |   |                                                 °
°°°°°°°°°°°°°°°°°°°°°°°°°|°°°|°°°°°°°°°°°°°°°°°°°
```

```
| subset 1 |
|__________|_____________________________________________________
|                                                                |
|             FILE_SERVICES_SUPPORT base subset                  |
|                                                                |
|________________________________________________________________|
```

Figure 9-96. Base and Optional Subsets of FILE_SERVICES_SUPPORT Function Set

---

2.9.4.4 FILE_SERVICES_SUPPORT Base Subset

Subtopics:

- 2.9.4.4.1 Functions Provided
- 2.9.4.4.2 Formats Supported
- 2.9.4.4.3 Implementation Requirements

2.9.4.4.1 Functions Provided

The FILE_SERVICES_SUPPORT base subset provides the ability to route management services requests and bulk data between function set groups located at separate LUs. This routing is accomplished over LU-LU sessions used by SNA/Distribution Services (SNA/DS).

2.9.4.4.2 Formats Supported

This base subset provides:

° The capability to send and receive CP-MSUs, SNA/FS agent objects, and SNA/FS files (bulk data) over LU-LU sessions.

° Support for both processing and building SNA/FS agent objects, containing one of the following:

- TRANSFER_TO_REQUESTER command

- REPORT_FS_ACTION command

  where the SNA/FS server instruction in the server object is one of the following:

  -- CREATE/LOAD

  -- CREATE/LOAD_OR_REPLACE

- REPORTING_FS_ACTION report

and support for processing, but not building:

- REPORT_FS_ACTION command

  where the SNA/FS server instruction in the server object is:

  -- DELETE

2.9.4.4.3 Implementation Requirements

The requirements for implementing the FILE_SERVICES_SUPPORT base subset are described by a model consisting of a subset of SNA/Distribution Services and SNA/Management Services.

A Subset of SNA/Distribution Services (SNA/DS):

° Interacts with MS via protocol boundary 7 as follows:

- Notifies MS that a management services MU and bulk data have arrived on a conversation

- Receives a management services MU and bulk data from another SNA/DS node

- Sends an MU and bulk data to the specified LU on a conversation when requested by MS

A Subset of SNA/Management Services:

° Provides the following processes:

- Sending requests and data

  -- Receives CP-MSUs and SNA/FS parameters from specialized management services function set groups on protocol boundary F

  -- Sends SNA/FS agent objects and bulk data to other nodes, sending them on a conversation via SNA/DS

  -- Passes back SNA/DS exception conditions if the agent object and bulk data could not be sent

- Receiving requests and data

  -- Receives CP-MSUs and bulk data from other nodes on an SNA/DS conversation and routes the data to the appropriate XXXX_NETOP or EP_XXXX specialized function set group on protocol boundary G

  -- Receives SNA/FS agent objects and bulk data from other nodes on an SNA/DS conversation and processes them

- Processing SNA/FS Agent Objects

  -- Parses a SNA/FS agent object and honors the request contained within it, which is of one of the following kinds:

    ° To send bulk data to another node, or to build and send an SNA/FS report if there are exceptions in attempting to do so, or

    ° To build and send an SNA/FS report indicating either success or failure in attempting to store a file at this node.

Sending Requests and Data:

This process is started by one of the following:

° A specialized management services function set group that has CP-MSU and bulk data destined for a remote LU. The process is started via internal MS protocol boundary F and is passed a CP-MSU, an agent unit-of-work correlator, SNA/FS server object parameters, and a Network-qualified LU name.

° The process described in "Processing SNA/FS Agent Objects."

° The network operator, if the Network Operator Support optional subset is implemented.

After being started, this process issues SEND_DISTRIBUTION to pass the agent object, along with SNA/FS parameters, to SNA/Distribution Services.

Note: Refer to the SNA/DS and SNA/FS references listed in "Prerequisite Publications" in topic PREFACE.4 for a complete description of the SNA/DS verbs and included SNA/FS parameters.

SEND_DISTRIBUTION is prepared as follows:

- DISTRIBUTION_ID specifies the Network-qualified LU name of this LU
- AGENT_CORREL specifies the correlation value
- DESTINATION specifies the Network-qualified destination LU name passed via protocol boundary F
- DEST_AGENT specifies SNA/Management Services (X'23F0F0F0')
- AGENT_OBJECT contains either the CP-MSU or the SNA/FS agent object
- SERVICE_PARMS specifies
  - priority=DATA_12 or lower
  - protection=YES
  - capacity=16 MEG
  - security=YES OR NO
  - accept_delay=INDEFINITE
- REPORTING_REQUESTED specifies
  - exception_report_req=YES
- INTEGRITY specifies HIGH
- REPORT-TO_DSU specifies the Network-qualified LU name of the node that initiated this unit of work
- SERVER specifies SNA/File Services (X'24F0F0F0')
- SPECIFIC_SERVER_INFO specifies the SNA/FS parameters

The return code from the execution of the verb along with any SNA/DS exception information is passed back to the process that started this process.

This process then terminates.

Receiving Requests and Data:

This process is started by SNA/DS in this node when it receives data for SNA/MS (as specified by the agent TP name in the MU). After being started, it issues RECEIVE_DISTRIBUTION to receive the data sent to it by a remote LU The following returned values are used by this process:

- DISTRIBUTION_ID specifies the Network-qualified LU name of the LU that the MU was received from
- AGENT_CORREL specifies the Agent Unit-of-work Correlator (X'1549') GDS variable sent by the remote LU
- ORIGIN specifies the Network-qualified origin LU name to be passed via protocol boundary G
- AGENT_OBJECT (if specified) specifies one of the following:
  - A CP-MSU (X'1212') GDS variable carrying a major vector, or
  - An FS command
- REPORTING_REQUESTED specifies
  - exception_report_req=YES
- REPORT-TO_DSU specifies the Network-qualified LU name of the originator of the unit of work

- DISTRIBUTION_TIME specifies the time at which the distribution originated

- SPECIFIC_SERVER_INFO specifies the SNA/FS parameters

- RETURN_CODE specifies results of RECEIVE_DISTRIBUTION verb execution.

If the agent object contains a CP-MSU, this process then passes it along with other required parameters to the appropriate specialized management services function set via internal MS protocol boundary G and terminates.

Otherwise (if the agent object does not contain a CP-MSU), this process starts the process described in "Processing SNA/FS Agent Objects."

Processing SNA/FS Agent Objects:

This process is started when the process described in "Receiving Requests and Data" receives an FS command. After being started, this process parses the agent object.

Refer to Table 9-40.

Table 9-40. Choosing Agent Object Contents and Server Parameters from the Command Received

SNA/FS command received	SNA/FS agent object contents to send	SNA/FS server instructions to send (E=encoder D=decoder S=source T=target)
TRANSFER_ TO_ REQUESTER	None (agent object is omitted)	S: (FETCH, ABEND, ONLY_ IF_ EXCEPTIONS) T: (CREATE/LOAD_ OR_ REPLACE, ABEND, DETAILED, STORING) *for focal point targets* *or* T: (CREATE/LOAD_ OR_ REPLACE, ABEND, DETAILED, EXECUTING) *for entry point targets*
REPORT_ FS_ ACTION	REPORTING_ FS_ ACTION	E: (ENCODE_ ONLY, ABEND, ONLY_ IF_ EXCEPTIONS) D: (DECODE_ ONLY, ABEND, ONLY_ IF_ EXCEPTIONS)

Notes:

1. In the case of TRANSFER_TO_REQUESTER, the file name is a Fetched File Name if SNA/FS action is successful, a To-Be-Fetched File Name if it is not.

2. In the REPORTING_FS_ACTION report, the file name is a Stored Name or a Deleted Name if SNA/FS action is successful, a To-Be-Stored Name or To-Be-Deleted Name if it is not.

If the agent object contains a TRANSFER_TO_REQUESTER command, this process uses the specified FS parameters required to fetch the data object, and issues SEND_DISTRIBUTION to transfer the file to the requesting node, using the same parameters as described in "Sending Requests and Data," but omitting AGENT_OBJECT.

If the agent object contains a REPORT_FS_ACTION command, this process encodes FS parameters required to report on the status of the stored data object, builds an FS agent object containing a REPORTING_FS_ACTION command, and then issues SEND_DISTRIBUTION.

If this process is unable to parse the agent object, or in any other exception conditions, it builds an agent object containing an SNA Condition Report (X'1532') GDS variable and issues SEND_DISTRIBUTION.

This process then terminates.

2.9.4.5 FILE_SERVICES_SUPPORT Optional Subset 1 (Network Operator Support)

Subtopics:

- 2.9.4.5.1 Functions Provided
- 2.9.4.5.2 Verbs Supported
- 2.9.4.5.3 Implementation Requirements

2.9.4.5.1 Functions Provided

Optional Subset 1 (Network Operator Support) provides the capability to interact with the network operator at the node to receive request verbs and pass back reply verbs. Refer to Appendix B, "Management Services Protocol Boundary Verbs" in topic B.0 for a detailed description of these verbs.

2.9.4.5.2 Verbs Supported

This optional subset provides support for receiving and processing the following verbs from the network operator:

° SEND

° RETRIEVE

and issuing the following verbs to the network operator:

° REPLY_TO_SEND

° REPLY_TO_RETRIEVE

° NOTIFICATION_OF_ARRIVAL

2.9.4.5.3 Implementation Requirements

The additional requirements for implementing this optional subset are described by a model consisting of the following:

A Subset of SNA/Management Services:

° Provides the following processes:

- Sending requests and data

  -- Receives SNA/FS parameters from the network operator, and sends them after the SNA/FS agent object is built to other nodes on a conversation via SNA/DS

  -- Passes back SNA/DS exception conditions if the agent object and bulk data could not be sent

- Receiving data

  -- Receives SNA/FS agent object reports and passes them to the network operator after converting them to the reply verb format.

- Processing SNA/FS Agent Objects

  -- Builds a SNA/FS agent object from a network operator verb

Sending Requests and Data:

The process described in "Sending Requests and Data" in topic 2.9.4.4.3 is enhanced to support being started by:

° The network operator.

Then, the process described in "Processing SNA/FS Agent Objects" in topic 2.9.4.4.3 (enhanced as described below) is started to build the required SNA/FS agent object from the verb. Otherwise, processing is the same.

Receiving Data:

The process described in "Receiving Requests and Data" in topic 2.9.4.4.3 is enhanced to convert SNA/FS reports and notifications of the arrival of files into reply verbs. The appropriate reply verb is chosen in an obvious manner from the request with which it is correlated using the Agent Unit-of-Work correlator:

- An outstanding SEND request will result in receipt of a REPORTING_FS_ACTION report, and the REPLY_TO_SEND reply verb is built.

- An outstanding RETRIEVE request will result in receipt of a file but no agent object, and the REPLY_TO_RETRIEVE reply verb is built.

This process then passes the reply verb to the network operator and terminates.

Processing SNA/FS Agent Objects:

The process described in "Processing SNA/FS Agent Objects" in topic 2.9.4.4.3 is enhanced:

- To build SNA/FS agent objects from the verb issued by the network operator.

- To pass REPORTING_FS_ACTION reports back to the process described in "Receiving Requests and Data" in topic 2.9.4.4.3.

This process uses the following table to determine which kind of agent object to build, and which server instructions to include in the SNA/FS parameters. After building this data, it returns control to the process that started it and terminates.

Table 9-41. Choosing Command and Server Instructions from the Verb

Verb issued by network operator	**SNA/FS Command**	**SNA/FS Server Instructions (E=encoder D=decoder S=source T=target)**
RETRIEVE	TRANSFER_ TO_ REQUESTER	E: (ENCODE_ ONLY, ABEND, ONLY_ IF_ EXCEPTIONS) D: (DECODE_ ONLY, ABEND, ONLY_ IF_ EXCEPTIONS) S: (FETCH, ABEND, ONLY_ IF_ EXCEPTIONS) T: (CREATE/LOAD_ OR_ REPLACE, ABEND, DETAILED, STORING) *for focal point targets* *or* T: (CREATE/LOAD_ OR_ REPLACE, ABEND, DETAILED, EXECUTING) *for entry point targets*
SEND (DESTRUCTION ALLOWED)	REPORT_ FS_ ACTION	S: (FETCH, ABEND, ONLY_ IF_ EXCEPTIONS) T: (CREATE/LOAD_ OR_ REPLACE, ABEND, DETAILED, EXECUTING) *for entry point targets when source is a focal point* *or* T: (CREATE/LOAD_ OR_ REPLACE, ABEND, DETAILED, STORING) *otherwise*
SEND (DESTRUCTION NO)	REPORT_ FS_ ACTION	S: (FETCH, ABEND, ONLY_ IF_ EXCEPTIONS) T: (CREATE/LOAD, BACKOUT, DETAILED, EXECUTING) *for entry point targets when source is a focal point* *or* T: (CREATE/LOAD, BACKOUT, DETAILED,

```
|                     |                     | STORING) otherwise       |
|_____________________|_____________________|__________________________|
```

Notes:

1. For REPORT_FS_ACTION,

   TARGET_AGENT_REPORTING_ACTION = DETAILED.

2. For TRANSFER_TO_REQUESTER,

   SOURCE_AGENT_REPORTING_ACTION = ONLY_IF_EXCEPTIONS and
   TARGET_AGENT_REPORTING_ACTION = ONLY_IF_EXCEPTIONS.

3. The REPORT_TO location is omitted, since the REPORT_TO party is the requester, or target agent.

2.9.5 SEND_DATA_SSCP_PU Function Set

```
  ................          ................
  .              .          .              .
  . EP_ALERT     .          . EP_XXXX¹     .
  . function set .          . function set .
  . group        .          . group        .
  ................          ................
      "    "   |                  |
      |    |   |________ _________|
      |    |            |
      |(C) |(B)         |(A)
°°°°°°|°°°°|°°°°°°°°°°°°|°°°°°°°°°°°°
```

```
        --  EP_ALERT
        --  EP_RTM
        --  EP_QPI
        --  EP_COMMON_OPERATIONS_SERVICES


        The input data consists of a complete NMVT built by the requesting
        function set group.  The details of this protocol boundary are
        described in "Protocol Boundary A--Send NMVT" in topic 2.9.1.3.


    -   Internal MS protocol boundary C (Delayed Alert Processing)


        If EP_ALERT optional subset 2 (Delayed Alert) is implemented, the
        SEND_DATA_SSCP_PU function set group requests the EP_ALERT
        function set group to process an NMVT to determine whether it can
        be transmitted at this time, classified as a delayed Alert to be
        transmitted at a later time, or logged if it cannot be transmitted
        at all.  After issuing such a request, this function set group
        will receive input from the EP_ALERT function set group indicating
        whether the NMVT can be transmitted.  The details of this protocol
        boundary are described in "Protocol Boundary C--Delayed Alert
        Processing" in topic 2.9.1.5.


  °  Output:


    -   Internal MS protocol boundary B (Held Alert Processing)


        If EP_ALERT optional subset 3 (Held Alert) is implemented, the
        SEND_DATA_SSCP_PU function set group requests the EP_ALERT
        function set group to do the following:


        --  Queue an Alert NMVT to be sent at a later time


        --  Queue a Resolution NMVT to be sent at a later time


        --  Log a non-Alert or non-Resolution NMVT that cannot be sent at
            this time, or an Alert or Resolution NMVT that cannot be
            queued


        --  Remove an Alert from the held Alert queue and pass it to the
            SEND_DATA_SSCP_PU function set group to be sent on the SSCP-PU
            session.


        --  Remove a Resolution NMVT from the held Alert queue and pass it
            to the SEND_DATA_SSCP_PU function set group to be sent on the
            SSCP-PU session.


        The details of this protocol boundary are described in "Protocol
        Boundary B--Held Alert Processing" in topic 2.9.1.4.


    -   Internal MS protocol boundary C (Delayed Alert Processing)


        If EP_ALERT optional subset 2 (Delayed Alert) is implemented, the
        SEND_DATA_SSCP_PU function set group requests the EP_ALERT
        function set group to process an NMVT to determine whether it can
        be transmitted at this time, classified as a delayed Alert to be
        transmitted at a later time, or logged if it cannot be transmitted
        at all.  The details of this protocol boundary are described in
        "Protocol Boundary C--Delayed Alert Processing" in topic 2.9.1.5.
```

2.9.5.2 Prerequisite Function Sets

See "Role Requirements for Management Services Components" in topic 2.8.1 for information on the relationships between this function set and other function sets.

2.9.5.3 Overview of Subsets

```
 _____________________________________
|                                     |
|   no optional subsets available for |
|    SEND_DATA_SSCP_PU function set   |
|                                     |
|_____________________________________|
|                                     |
|    SEND_DATA_SSCP_PU base subset    |
|                                     |
|_____________________________________|
```

Figure 9-98. Base and Optional Subsets of SEND_DATA_SSCP_PU Function Set

2.9.5.4 SEND_DATA_SSCP_PU Base Subset

Subtopics:

- 2.9.5.4.1 Functions Provided
- 2.9.5.4.2 Formats Supported
- 2.9.5.4.3 Implementation Requirements

2.9.5.4.1 Functions Provided

PUMS communicates with CPMS on an SSCP-PU session by means of the NMVT. A brief description and example flow for each of these RUs is presented in Chapters 3, 4, 5, and 7. SEND_DATA_SSCP_PU provides the following support for sending the NMVT RU:

° Causes an NMVT to be sent on the SSCP-PU session

° Deals with the +RSP to an NMVT from CPMS

° Assists the EP_ALERT function set group in the processing of delayed Alerts, held Alerts, and held Resolution major vectors.

2.9.5.4.2 Formats Supported

This function set group supports the sending of NMVTs, and the receipt of responses to NMVTs.

2.9.5.4.3 Implementation Requirements

The requirements for implementing the SEND_DATA_SSCP_PU base subset are described by a model consisting of a subset of PU configuration services, the half-session and PU management services.

A Subset of the Half-Session:

° Interacts with PUMS via PUMS protocol boundary 4 as follows:

- Notifies PUMS that an NMVT response RU has arrived on an SSCP-PU session

- Receives an NMVT response RU from an SSCP-PU session when requested by PUMS

- Sends an NMVT RU on a specified SSCP-PU session when requested by PUMS

A Subset of PU Management Services:

° Provides the following processes:

- Sending NMVTs

  -- Sends an NMVT on the SSCP-PU session when requested by other function set groups via protocol boundary A

  -- Assists the EP_ALERT function set group in the processing of

delayed Alerts, held Alerts, and held Resolution major vectors.

- Receiving NMVT responses

  -- Receives an NMVT response when notified by the half-session

  -- Assists the EP_ALERT function set group in the processing of held Alerts and held Resolution major vectors

Sending NMVTs:

This process receives NMVTs from other function set groups and causes them to be sent to CPMS.

This process is started by the noted processes in the following function set groups:

° EP_ALERT ("Sending Alerts" in topic 2.10.1.4.3)

° EP_RTM ("Sending RTM Data" in topic 2.10.2.4.4)

° EP_QPI ("Sending QPI data" in topic 2.10.3.4.3)

° EP_COMMON_OPERATIONS_SERVICES ("Sending Common Operations Services Data" in topic 2.10.5.4.4)

When one of these function set groups starts this process, it passes a complete NMVT to it via protocol boundary A. Refer to "Protocol Boundary A--Send NMVT" in topic 2.9.1.3 for details. After it is started, the process proceeds as follows:

It checks for the existence of the delayed Alert control block, and, if it is present, starts the process described in "Processing Delayed Alerts" in topic 2.10.1.6.3, passing it a pointer to the NMVT to be transmitted via internal MS protocol boundary C item 1. See "Protocol Boundary C--Delayed Alert Processing" in topic 2.9.1.5. The delayed Alert control block is present if optional subset 1 (Delayed Alert) of the EP_ALERT function set is implemented. If the delayed Alert control block is empty, and if the current NMVT is not itself a delayed Alert, then this process can send this NMVT to CPMS. This is indicated by returned item 1 from the Delayed Alert process. See "Protocol Boundary C--Delayed Alert Processing" in topic 2.9.1.5. If the returned value is non-zero, the NMVT cannot be transmitted and this process ends.

If the PRID value is non-zero (i.e., if the NMVT was solicited) this process requests the half-session to send the NMVT that it has constructed to the control point address passed to this process in item 2, i.e., the control point that solicited it.

If the PRID value is X'000', (indicating unsolicited data), then this process requests the half-session to send the NMVT it has constructed to CPMS.

Figure 10-4 in topic 2.10.1.7.4 shows the structure of the held Alert control block. If the half-session was unable to send an NMVT, and if the held Alert control block is present (if optional subset 3 of the EP_ALERT function set is implemented), this process starts the process described in "Sending Held Alerts" in topic 2.10.1.7.4 via internal MS protocol boundary B. A pointer to the completed NMVT is passed in item 1. See "Protocol Boundary B--Held Alert Processing" in topic 2.9.1.4.

After this process has sent each NMVT it has received, passed it (when possible) to the Delayed Alert or Send Held Alert process, to be saved as a delayed or held Alert, or had it logged locally, it terminates.

Receiving NMVT Responses:

This process receives responses to an NMVT from a control point and (if optional subset 3 of the EP_ALERT function set is implemented) starts the process described in "Sending Held Alerts" in topic 2.10.1.7.4 if any Alerts and/or Resolution major vectors have been held.

When a response to an NMVT arrives on the SSCP-PU session, the half-session starts this process which receives the response.

Figure 10-4 in topic 2.10.1.7.4 shows the structure of the held Alert control block. If the held Alert control block is present (if optional

subset 3 of the EP_ALERT function set is implemented), and if HEAD_POINTER is non-zero, this process starts the process described in "Sending Held Alerts" in topic 2.10.1.7.4 via internal MS protocol boundary B. Zero is specified in item 1 of the *Data Content* in the protocol boundary description. See "Protocol Boundary B--Held Alert Processing" in topic 2.9.1.4. Note that Resolution major vectors may also be held in the held Alert control block. For this reason, internal MS protocol boundary B (and the process it initiates) may be used for both Alerts and Resolution major vectors.

This process then terminates.

2.9.6 RECEIVE_REQUEST_SSCP_PU Function Set

```
                        ................
                        .              .
                        . EP_XXXX¹     .
                        . function set .
                        . group        .
                        .              .
                        ................
                           "      |
                           |      |
                           |      |
                           |(D)   |(E)
°°°°°°°°°°°°°°°°°°°°°°°°°°°|°°°°°°|°°°°°°°°°°°°°°°°°°°
```

details of this protocol boundary are described in "Protocol Boundary D--NMVT Received" in topic 2.9.1.6.

2.9.6.2 Prerequisite Function Sets

See "Role Requirements for Management Services Components" in topic 2.8.1 for information on the relationships between this function set and other function sets.

2.9.6.3 Overview of Subsets

```
 _____________________________________
|                                     |
|   no optional subsets available for |
|  RECEIVE_REQUEST_SSCP_PU  function set |
|                                     |
|_____________________________________|
|                                     |
|  RECEIVE_REQUEST_SSCP_PU  base subset |
|                                     |
|_____________________________________|
```

Figure 9-100. Base and Optional Subsets of RECEIVE_REQUEST_SSCP_PU Function Set

---

2.9.6.4 RECEIVE_REQUEST_SSCP_PU Base Subset

Subtopics:

- 2.9.6.4.1 Functions Provided
- 2.9.6.4.2 Formats Supported
- 2.9.6.4.3 Implementation Requirements

2.9.6.4.1 Functions Provided

RECEIVE_REQUEST_SSCP_PU base subset provides the basic capability to receive NMVTs, perform basic parsing of the NMVT, and pass the NMVT to the appropriate function set group or send a negative response if errors are detected while parsing. It is a prerequisite for any function set supporting an NMVT request.

2.9.6.4.2 Formats Supported

The RECEIVE_REQUEST_SSCP_PU function set group provides the ability to receive NMVTs and do the processing detailed below. It requires other function set groups to complete NMVT processing.

2.9.6.4.3 Implementation Requirements

The requirements for implementing the RECEIVE_REQUEST_SSCP_PU base subset are described by a model consisting of subsets of the half-session and PU management services.

A Subset of the Half-Session:

° Interacts with PUMS via PUMS protocol boundary 4

- Notifies PUMS that an NMVT RU has arrived on an SSCP-PU session

- Receives an NMVT RU from an SSCP-PU session when requested by PUMS

- Sends an NMVT response RU on an SSCP-PU session when requested by PUMS

A Subset of PU Management Services:

° Provides the following processes:

- Receiving NMVTs

  -- When notified by the half-session, receives an NMVT, performs preliminary parsing, and does the following:

  ° If no errors were detected in parsing, passes the NMVT to the appropriate function set group (via protocol boundary D)

  ° If errors were detected in parsing, starts the process described in "Sending NMVT Responses" (via protocol boundary E) to send a negative response

- Sending NMVT responses

  -- Constructs an NMVT response RU and sends it on an SSCP-PU session when requested by other function set groups (via protocol boundary E)

Receiving NMVTs:

When a request NMVT arrives on the SSCP-PU session, the following sequence of events takes place (refer to "Parsing of NMVTs" in topic 2.10.7.7 for details of common NMVT parsing):

1. The half-session manager examines the NS header to determine where within the PU to route the request.

2. Since the header is X'41038D', indicating that the RU is an NMVT, the half-session manager does one of the following:

   ° If PUMS in the node does not support NMVTs, the half-session manager sends a -RSP (1003 0001) to indicate that NMVTs are not supported.

   ° If PUMS is not prepared to process an NMVT, the PU resources manager sends a -RSP (0815 0003) to indicate that another management services request is currently being processed and the request cannot be accepted at this time.

   ° If PUMS in the node supports NMVTs, and if it is currently prepared to process one, the half-session manager starts this process.

3. Having been started by the half-session manager, this process receives the NMVT. It is passed the length of the NMVT, as well as an indication of what control point sent the NMVT. Optionally, this process locates the major vector length in the NMVT. (As indicated in Table 10-47 in topic 2.10.7.7, checking for an invalid major vector length is an option for an implementation of PUMS. The PUMS model presented here will include all optional checks, for completeness.) If the major vector length is incompatible with the RU length in the transmission header, this process starts the process described in "Sending NMVT Responses" (RECEIVE_REQUEST_SSCP_PU function set), passing to it the sense data value X'086F 0001'. That process sends a -RSP containing this sense data value to CPMS.

4. If the major vector length is compatible with the RU length in the transmission header, this process locates the major vector key in the NMVT.

   ° If the major vector key is X'8080', and the EP_RTM function set has been implemented, then this process starts the process described in "Receiving RTM Requests" in topic 2.10.2.4.4, passing the NMVT pointer and the address of the sending control point to it.

   ° If the major vector key is X'8090', and the EP_QPI function set has been implemented, then this process starts the process

described in "Receiving QPI requests" in topic 2.10.3.4.3, passing the NMVT pointer and the address of the sending control point to it.

° If the major vector key is X'8061', X'8062', X'8063', or X'8064', and the EP_COMMON_OPERATIONS_SERVICES function set has been implemented, then this process starts the process described in "Receiving Common Operations Services Requests" in topic 2.10.5.4.4, passing the NMVT pointer and the address of the sending control point to it.

See "Protocol Boundary D--NMVT Received" in topic 2.9.1.6 for details of the protocol boundary used in the preceding cases.

° If the major vector key specifies a function not supported by this instance of PUMS, then this process starts the process described in "Sending NMVT Responses," passing to it the sense data value X'080C 0005' and the address of the sending control point. That process will send a -RSP containing this value to CPMS in the control point that sent the request.

5. The process started by this process executes, then it returns control to this process, which terminates.

Sending NMVT Responses:

This process is started by the noted processes of the following function set groups:

° RECEIVE_REQUEST_SSCP_PU ("Receiving NMVTs")

° EP_RTM ("Receiving RTM Requests" in topic 2.10.2.4.4)

° EP_QPI ("Receiving QPI requests" in topic 2.10.3.4.3)

° EP_COMMON_OPERATIONS_SERVICES ("Receiving Common Operations Services Requests" in topic 2.10.5.4.4)

See "Protocol Boundary E--Send NMVT Response" in topic 2.9.1.7 for a description of the protocol boundary by which this process was started.

After it has sent a response, this process returns control to the process that started it.

2.10 Chapter 10. Specialized Management Services Function Sets for Entry Points

Subtopics:

- 2.10.1 EP_ALERT Function Set
- 2.10.2 EP_RTM Function Set
- 2.10.3 EP_QPI Function Set
- 2.10.4 EP_CHANGE_MGMT Function Set
- 2.10.5 EP_COMMON_OPERATIONS_SERVICES Function Set
- 2.10.6 EP_OPERATIONS_MGMT Function Set
- 2.10.7 Common EP_XXXX Functions
- 2.10.8 EP_XXXX Parsing of Individual Management Services Major Vectors

---

2.10.1 EP_ALERT Function Set

```
                                  _____________________   .
                                 | served application   | .
                                 |     program 2        |.
                              _____________________    |
                             | served application   |__|
                             |     program 1        |
                             |                      |
                             |______________________|
                                        |
                                        |(A-5)
°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°|°°°°°°°°°°°°°°°°°°°
```

```
°        _________________          |                                          °
°       | configuration   |         |                                          °
°       | services        |         |                                          °
°       |_________________|         |                                          °
°                | (6)               |          __________      ___________    °
°                                    |         | physical  |     |    LU     |__ °
°                                    |       | resources |  (2) |  _____   |  |  °
°  ____________________________________        | manager   |    ___|__| LMS |  |  |  °
°  |                                 |  (1) |  _____   |   |  |_____|  |  |  °
°  |                                 |_____|__| LMS |  |   |   |___________|  |  °
°  |                                 |      |  |_____|  |   |     |___________|  °
°  |                                 |      |__________|   |                    °
°  |                                 |                     |   __________      °
°  |                                 |                     |  |  DLC      |__  °
°  |                                 |_____________________|  |  manager  |  |  °
°  |                                 |                       (3) |  _____   |  |  °
°  |                          MS     |_________________________|__| LMS |  |  |  °
°  |_________________________________|                          |_____|  |  |  °
°      |      "    "      |    "    |    "    |                 |___________|  |  °
°      |      |    |      |    |    |    |    |                   |___________|  °
°      |      |    |      |    |    |    |    |                EP_ALERT function set group  °
°      |      |    |      |    |    |    |    |                                          °
°°°°°°°°|°°°°°°|°°°°|°°°°°|°°°|°°°|°°°|°°°|°°°°°°°°
```

If optional subset 3 (Held Alert for PUMS) is implemented, the EP_ALERT function set group receives requests from the SEND_DATA_SSCP_PU function set group to do the following:

-- Queue an problem notification NMVT to be sent at a later time

-- Log a non-problem notification NMVT, or a problem notification NMVT that cannot be queued

-- Remove an problem notification from the held queue and pass it to the SEND_DATA_SSCP_PU function set group to be sent on the SSCP-PU session

The details of this protocol boundary are described in "Protocol Boundary B--Held Alert Processing" in topic 2.9.1.4.

- Internal MS protocol boundary C (Delayed Problem Notification Processing)

  If optional subset 2 (Delayed Alert) is implemented, the EP_ALERT function set group receives requests from the SEND_DATA_SSCP_PU function set group to process an NMVT to determine whether it can be transmitted at this time, classified as a delayed problem notification to be transmitted at a later time, or logged if it cannot be transmitted at all. The details of this protocol boundary are described in "Protocol Boundary C--Delayed Alert Processing" in topic 2.9.1.5.

- Internal MS protocol boundary Q (MDS-MU Received)

  The EP_ALERT function set group receives requests from the MULTIPLE_DOMAIN_SUPPORT function set group to process incoming MDS-MUs. The only MDS-MU received by EP_ALERT is an MDS error message reporting that an problem notification was not successfully delivered to the focal point. The details of this protocol boundary are described in "Protocol Boundary Q--MDS-MU Received" in topic 2.9.1.12.

- Internal MS protocol boundary V (Current Focal Point)

  The identity of an application program's current focal point is passed across protocol boundary V from the MS_CAPS function set group. The application program must have previously registered itself with MS_CAPS via protocol boundary T. The details of this protocol boundary are described in "Protocol Boundary V--Current Focal Point" in topic 2.9.1.17.

  The process by which an application program registers itself with MS_CAPS is described in "Application Program Registration with MS_CAPS" in topic 2.9.3.10.

- MS application program protocol boundary A-5 (Operations Management Support)

  If "EP_ALERT Optional Subset 14 (Operations Management Support)" in topic 2.10.1.18 is implemented, the EP_ALERT function set group receives CP-MSUs containing one Routing and Targeting Instructions (X'154D') GDS variable (RTI) in addition to an Alert and/or Resolution major vector. The details of this protocol boundary are described in "EP_ALERT Optional Subset 14 (Operations Management Support)" in topic 2.10.1.18.

° Output:

- Internal MS protocol boundary A (Send NMVT)

  The EP_ALERT function set group requests the SEND_DATA_SSCP_PU function set group to send an NMVT RU on the SSCP-PU session with its controlling CP. The output data consists of a complete NMVT built by this function set group. The details of this protocol boundary are described in "Protocol Boundary A--Send NMVT" in topic 2.9.1.3.

- Internal MS protocol boundary C (Delayed Problem Notification Processing)

  If EP_ALERT optional subset 2 (Delayed Alert) is implemented, the EP_ALERT function set group receives requests from the SEND_DATA_SSCP_PU function set group to process an NMVT to determine whether it can be transmitted at this time, classified as a delayed problem notification to be transmitted at a later time, or logged if it cannot be transmitted at all. After receiving such a request, this function set group will produce

output for the SEND_DATA_SSCP_PU function set group indicating whether the NMVT can be transmitted. The details of this protocol boundary are described in "Protocol Boundary C--Delayed Alert Processing" in topic 2.9.1.5.

- Internal MS protocol boundary O (Register Application Program Name)

  An application program utilizing the MDS transport registers itself with the MULTIPLE_DOMAIN_SUPPORT function set group via this protocol boundary. The details of this protocol boundary are described in "Protocol Boundary O--Register Application Program Name" in topic 2.9.1.11.

  The process by which an application program registers itself with MULTIPLE_DOMAIN_SUPPORT is described in "Registering with MDS" in topic 2.9.2.9.1.

- Internal MS protocol boundary R (Send MDS-MU)

  The EP_ALERT function set group requests the MULTIPLE_DOMAIN_SUPPORT function set group to route problem notifications in MDS-MU format to the appropriate destination. The details of this protocol boundary are described in "Protocol Boundary R--Send MDS-MU" in topic 2.9.1.13.

- Internal MS protocol boundary T (Register MS Category)

  In the process of initialization, management services application programs register themselves with the MS_CAPS function set group via protocol boundary T if they want focal point information. The application program passes its name and the category for which it is registering. The details of this protocol boundary are described in "Protocol Boundary T--Register MS Category" in topic 2.9.1.15.

  The process by which an application program registers itself with MS_CAPS is described in "Application Program Registration with MS_CAPS" in topic 2.9.3.10.

2.10.1.2 Prerequisite Function Sets

See "Role Requirements for Management Services Components" in topic 2.8.1 for information on the relationships between this function set and other function sets.

2.10.1.3 Overview of Subsets

```
 ____________________________________________________________________________
|  Optional Subsets 1 through 14.  With the exception of subsets 3 and 12; and 3 and 14,      |
|  which are mutually exclusive, all other subsets are mutually independent;                 |
|  the only prerequisite for each is the EP_ALERT base subset.                                |
|                                                                                             |
|   ° Optional Subset  1:  Problem Diagnosis Data                                             |
|   ° Optional Subset  2:  Delayed Alert                                                      |
|   ° Optional Subset  3:  Held Alert for PUMS                                                |
|   ° Optional Subset  4:  Operator_Initiated Alert                                           |
|   ° Optional Subset  5:  Qualified Message Data                                             |
|   ° Optional Subset  6:  Self_Defining Text Message Subvector                               |
|   ° Optional Subset  7:  LAN Alert                                                          |
|   ° Optional Subset  8:  SDLC/LAN LLC Alert                                                 |
|   ° Optional Subset  9:  X.21 Alert                                                         |
|   ° Optional Subset 10:  Hybrid Alert                                                       |
|   ° Optional Subset 11:  X.25 Alert                                                         |
|   ° Optional Subset 12:  Held Alert for CPMS                                                |
|   ° Optional Subset 13:  Resolution Notification Support                                    |
|   ° Optional Subset 14:  Operations Management Support                                      |
|                                                                                             |
|____________________________________________________________________________________________|
| EP_ALERT base subset:                                                                       |
|   ° Sending Alerts                                                                          |
|   ° Support of all subvectors and subfields, as required for problem determination,        |
|     except those listed in the note below                                                   |
|   ° When EP_ALERT is in an APPN network node or end node:                                   |
|       - registration with MULTIPLE_DOMAIN_SUPPORT                                          |
|       - registration with MS_CAPS                                                           |
|                                                                                             |
|____________________________________________________________________________________________|
```

Note: Support for the following major vectors/subvectors/subfields is not part of the EP_ALERT base subset; it falls into the indicated optional subsets:

```
Qualified Message Data (X'01') subfield.............Optional Subset  5
Self_Defining Text Message (X'31') subvector.........Optional Subset  6
Incident Identification (X'4A') subvector...........Optional Subset 13
LAN LCS Data (X'51') subvector......................Optional Subset  7
LCS Configuration Data (X'52') subvector............Optional Subset  8, 9 or 11
SDLC Link Station Data (X'8C') subvector............Optional Subset  8
Text Message (X'00') subvector......................Optional Subset 10
Hierarchy Name List (X'03') subvector...............Optional Subset 10
Basic Alert (X'91') subvector.......................Optional Subset 10
Detail Qualifier (EBCDIC) (X'A0') subvector..........Optional Subset 10
Detail Qualifier (hexadecimal) (X'A1') subvector.....Optional Subset 10
Resolution (X'0002') major vector...................Optional Subset 13
```

Figure 10-2. Base and Optional Subsets of EP_ALERT Function Set

---

2.10.1.4 EP_ALERT Base Subset

Subtopics:

- 2.10.1.4.1 Functions Provided
- 2.10.1.4.2 Formats Supported
- 2.10.1.4.3 Implementation Requirements

2.10.1.4.1 Functions Provided

An Alert is an unsolicited notification of an actual or impending loss of availability of a system resource to an end user. This base subset of EP_ALERT provides the following:

° Detects an Alert condition for any resource controlled by this node

° Constructs an NMVT containing an Alert (X'0000') major vector, or an MDS-MU containing one or more Alert (X'0000') major vectors

° Passes the NMVT to the SEND_DATA_SSCP_PU function set group, to be sent unsolicited, to all control points that control the resource to which the Alert applies

° Passes the MDS-MU to the MULTIPLE_DOMAIN_SUPPORT function set group, to be sent unsolicited to this node's current Alert focal point

Every implementation of PUMS or CPMS is required to send Alerts. (1) Support of certain *types* of Alerts, and of certain types of data on Alerts, is optional. Also optional is the saving of Alerts when the SSCP-PU session is unavailable and sending them when the session with the SSCP is re-established. Refer to the optional subsets that follow for a detailed description of the Alert options available to implementations.

Alert data falls into two broad categories: data indicating the nature of the problem being reported, and data identifying the component that has the problem. Some portions of this data are textual; these are displayed to the network operator just as they are received. Others are code points that index particular strings of stored text; these strings of text are displayed in a predetermined manner, upon receipt of the code points that index them.

Refer to "Problem Determination, Diagnosis, and Resolution via Alerts and Resolution Notifications" in topic 1.3.4 for additional details on the function provided by this base subset.

---

(1) More precisely, every implementation is required to be *capable of* sending Alerts, if this function has been enabled. Implementations are permitted to support *selective generation* of their Alerts, i.e., support an interface that allows some or all of their Alerts to be turned off by operator command. The granularity with which Alerts can be selected for enabling is left up to each implementation of EP_ALERT.

2.10.1.4.2 Formats Supported

The following subvectors are constructed by this function set group and included in the Alert major vector. The list here includes only those subvectors supported in the base subset of EP_ALERT; additional subvectors and the Resolution major vector are discussed below, in connection with the optional subsets of EP_ALERT.

- A Date/Time (X'01') or Relative Time (X'42') subvector, depending on the node's time-stamping capabilities

- An SNA Address List (X'04') subvector if the origin of the origin of the Alert condition is an SNA-addressable entity other than the sending PU.

  Note: This subvector is constructed only for an NMVT Alert. Resource identification for CP-MSU Alerts is always done in the Hierarchy/Resource List (X'05') subvector.

- A Hierarchy/Resource List (X'05') subvector logically identifying the origin of the Alert condition. In some cases the subvector also identifies one or more additional resources, e.g., session partners, logically associated with this origin.

  This subvector is constructed for an NMVT Alert if the origin of the Alert condition is not located in the products implementing the PU, and cannot be represented in the SNA Address List subvector.

  This subvector is also constructed if the origin of the Alert condition is a component of the hardware product implementing the PU, and requires operator attention to resolve the problem, e.g., an integrated disk drive that requires a diskette to be changed.

  This subvector is always constructed for a CP-MSU Alert.

- A Product Set ID subvector identifying the PU or CP sending the Alert.

- A second Product Set ID (X'10') subvector physically identifying the origin of the Alert condition, if the origin of the Alert condition is a hardware or software product and is not one of the products implementing the PU

- An MSU Correlation (X'47') subvector if the Alert sender is including a token for correlating the Alert with another MSU (e.g., an Alert from a second Alert sender) reporting on the same resource as the Alert.

- A Supporting Data Correlation (X'48') subvector if this function set group has preserved supporting data, e.g., a storage dump, to which the Alert must be correlated

- A Generic Alert Data (X'92') subvector

- A Probable Causes (X'93') subvector

The X'92' and X'93' subvectors are included in every Alert. The next five subvectors (X'94'-X'98') are included when appropriate. Support for these subvectors is required for an implementation of PUMS, even though not all of them are included in every Alert. The architecture does not specify *what* problem determination data an Alert sender is required to capture for the different Alert conditions it can detect. What it does require is the following: If an Alert sender has captured some problem determination data, and if this data is of a type that the Alert subvectors can transport, then the sender must include the data in the Alert it sends to report the problem. The architecture does not allow an implementation of PUMS to fail to send problem determination data that it has captured simply because it does not choose to support the subvector in which that data would flow.

- A User Causes (X'94') subvector

- An Install Causes (X'95') subvector

- A Failure Causes (X'96') subvector

- A Cause Undetermined (X'97') subvector

- A Detailed Data (X'98') subvector

2.10.1.4.3 Implementation Requirements

The requirements for implementing the EP_ALERT base subset are described by a model consisting of subsets of a number of LMSs inside and outside the node and a subset of management services.

Subsets of LMSs:

° Interact with MS via numbered protocol boundaries as follows:

- Detect a loss or impending loss of availability of the resources that they are managing, and report this loss of availability to MS such that an Alert major vector can be created.

Table 10-1 lists the LMSs that interact with MS in this way, and the numbered protocol boundary that each uses.

Table 10-1. LMSs That Initiate Alerts

Component Containing the LMS	PB Number
physical resources manager	1
LU	2
DLC manager	3
path control manager	9
CP session services	11
CP configuration services	12
CP address space manager	15
CP session connector manager	18
CP topology and routing services	19
CP directory services	20
CP management services	_

See Figure 1-5 in topic 1.1.2.4 and Figure 1-6 in topic 1.1.2.5 for an overview of these protocol boundaries.

A Subset of Management Services:

° Provides the following process:

- Sending Alerts

-- Constructs a complete NMVT containing an Alert (X'0000') major vector, or a complete MDS-MU containing one or more Alert (X'0000') major vectors. The NMVT is passed to the SEND_DATA_SSCP_PU function set group via protocol boundary A, or the MDS-MU is passed to the MULTIPLE_DOMAIN_SUPPORT function set group via protocol boundary R.

Sending Alerts:

This process is started by one of the LMSs when it has detected one or more Alert conditions. When this process is started, it receives data from the LMS describing each Alert condition. It uses this data to construct a complete Alert (X'0000') major vector for each Alert condition. Each Alert major vector constructed is then enveloped in a separate NMVT or CP-MSU, or multiple Alert major vectors are enveloped in a CP-MSU. If a CP-MSU is used, it is enveloped in an MDS-MU. This process then starts the process described in "Sending NMVTs" in topic 2.9.5.4.3 to send the NMVT to the controlling CPMS, or starts the process described in "Processing Send MDS-MU" in topic 2.9.2.4.8 to send the MDS-MU to the Alert focal point.

EP_ALERT in a node implementing PUMS will build an NMVT. For CPMS implementations, however EP_ALERT must make a choice of whether to build and send an NMVT or a CP-MSU. This choice is based on the following considerations:

1. Is "SEND_DATA_SSCP_PU Function Set" in topic 2.9.5 *NOT* implemented?

If so (i.e., SSCP-PU is *not* supported), CP-MSU will be used in every case.

2. Is "EP_ALERT Optional Subset 12 (Held Alert for CPMS)" in topic 2.10.1.16 implemented?

   If so, CP-MSU will be used in every case.

3. Is "EP_ALERT Optional Subset 14 (Operations Management Support)" in topic 2.10.1.18 implemented?

   If so, CP-MSU will be used in every case. The Alert (and/or Resolution major vector if optional subset 13 is implemented) and Routing and Targeting Instructions (X'154D') GDS variable will be sent to EP_ALERT already packaged in a CP-MSU.

4. Does the entry point have a focal point available?

   If so, CP-MSU will be used regardless of the function set implemented.

5. Otherwise, does the entry point has an active SSCP-PU session?

   If so, NMVT will be used.

Subvectors are constructed for each Alert as follows:

° Date/Time (X'01') or Relative Time (X'42') subvector: Depending on the node's capabilities, this process constructs one of two time stamps. See "Building the Date/Time (X'01') and Relative Time (X'42') Subvectors" in topic 2.10.7.1 for details.

° SNA Address List (X'04') subvector: If the required data is provided by the LMS, this process constructs an SNA Address List subvector. The LMS provides this data if the origin of the Alert condition is an SNA-addressable entity other than the PU. See "Building the SNA Address List (X'04') Subvector" in topic 2.10.7.2 for details.

  Note: This subvector is only constructed for Alerts sent as NMVTs.

° Hierarchy/Resource List (X'05') subvector: If the required data is provided by the LMS, this process constructs a Hierarchy/Resource List subvector. The LMS provides this data to logically identify (by name and resource type) the origin of the Alert condition for the following cases:

  - The origin of the Alert condition is not located in the products implementing the PU, and cannot be represented in the SNA Address List subvector.

  - The origin of the Alert condition is a component of the hardware product implementing the PU, and requires operator attention to resolve the problem, e.g., an integrated disk drive that requires a diskette to be changed.

  For certain Alerts, the LMS also identifies (again by name and resource type) one or more resources logically associated with the origin of the Alert condition. These resource identifications are also incorporated into the Hierarchy/Resource List subvector.

° Product Set ID (X'10') subvector: One instance of this subvector, identifying the PU sending the NMVT, or the CP sending the MDS-MU, is is always constructed by this process. If the required data is provided by the LMS, this process constructs a second Product Set ID subvector as well, to identify the component for which the Alert was generated. See "Building the Product Set ID (X'10') Subvector" in topic 2.10.7.3 for a description of how the Product Set ID subvector is constructed.

  Note: The LMS provides data for construction of a Product Set ID subvector to physically identify the origin of the Alert condition, if the origin of the Alert condition is a hardware or software product different from the products implementing the PU or CP.

  If data to identify a hardware product is not available to the LMS, the following values are passed:

  - Machine type: EBCDIC 0's (X'F0's)

  - Other hardware ID fields: binary 0's

Note: It is only in this instance of the product set ID, identifying an origin of an Alert condition different from the Alert sender, that these 0 values are allowed. The instance of the product set ID that identifies the Alert sender must have nonzero values in the subvector.

° Self-Defining Text Message (X'31') subvector: *If optional subset 6 (Text Message) was implemented*, and if the LMS provides one or more text messages, this process constructs a Self-Defining Text Message subvector containing the message text and code points identifying the entity sending the message, the coded character set in which the message is encoded, and the national language in which the message is written for each subvector. All of this information is supplied by the LMS.

Refer to optional subset 6 for additional details.

° MSU Correlation (X'47') subvector: If a correlation token is provided by the LMS, this process builds an MSU Correlation subvector to transport the token.

Note: The LMS provides a correlation token to make it possible for the Alert receiver to correlate the Alert containing it with another MSU, e.g., an Alert from a second Alert sender, reporting on the same resource.

° Supporting Data Correlation (X'48') subvector: If correlation data is provided by the LMS, this process builds a Supporting Data Correlation subvector to uniquely identify data associated with the problem reported by this Alert.

Note: The LMS provides correlation data if a dump, trace, or log data has been saved at the Alert sender, and is required to resolve the Alert condition.

° Incident Identification (X'4A') subvector: *If this optional subvector in optional subset 13 (Resolution Notification Support) was implemented*, the LMS provides a unique token (an incident identifier) to identify the Alert condition that has occurred. This process then builds an Incident Identification subvector containing the unique token.

Note: The LMS provides a unique token when it notifies MS that a particular Alert condition has occurred, and provides the same unique token when it notifies MS that the Alert condition has been resolved. This makes it possible for the Alert/Resolution receiver to correlate related Alert and Resolution major vectors.

° LAN Link Connection Subsystem Data (X'51') subvector: *If optional subset 7 (LAN Alert) was implemented*, this process constructs this subvector, including in it data supplied by the DLC manager LMS.

Refer to optional subset 7 for additional details.

° LCS Configuration Data (X'52') subvector: *If optional subset 8 (SDLC/LAN LLC Alert), optional subset 9 (X.21 Alert) or optional subset 11 (X.25 Alert) was implemented*, this process constructs this subvector, including in it data supplied by the DLC manager LMS.

Refer to optional subsets 8, 9 and 11 for additional details.

° SDLC Link Station Data (X'8C') subvector: *If optional subset 8 (SDLC/LAN LLC Alert) was implemented*, this process constructs this subvector, including in it data supplied by the DLC manager LMS.

Refer to optional subset 8 for additional details.

° Generic Alert Data (X'92') subvector: This process always constructs this subvector, as follows:

- The initiation indicator flag is set based on data supplied by the LMS.

- If optional subset 2 (Delayed Alert) is not implemented, the held Alert indicator and delayed Alert indicator flags are always set to 0 by this process; if optional subset 2 is implemented, these two flags are set based on information passed to this process by the LMS. See "EP_ALERT Optional Subset 2 (Delayed Alert)" in topic 2.10.1.6 and "EP_ALERT Optional Subset 3 (Held Alert for PUMS)" in topic 2.10.1.7 for details of the conditions under which these indicators are set to 1.

- The Alert Description Code is filled in, based on a value supplied by the LMS.

- The Alert ID Number is filled in; see "Identification of Unique Alerts" in topic 2.10.1.19.3 for a description of this number and how it is calculated.

° Probable Causes (X'93') subvector: This process always constructs this subvector, from data supplied by the LMS.

° User Causes (X'94'), Install Causes (X'95'), Failure Causes (X'96'), and Cause Undetermined (X'97') subvectors: This process constructs one or more of these subvectors, from data supplied by the LMS.

° Detailed Data (X'98') subvector: If the LMS supplies data for a Detailed Data subvector, this process constructs an instance of it.

° Text Message (X'00'), Hierarchy Name List (X'03'), Basic Alert (X'91'), Detail Qualifier (EBCDIC) (X'A0'), and Detail Qualifier (hexadecimal) (X'A1') subvectors: *If optional subset 10 (Hybrid Alert) was implemented*, this process always constructs the Basic Alert subvector, and may construct some or all of the other subvectors listed.

Refer to optional subset 10 for additional details.

After it has completed building subvectors for an Alert condition, this process places the subvectors in an Alert (X'0000') major vector. See "Building a Management Services Major Vector" in topic 2.10.7.4 for details of this step.

After it completes the Alert major vector, this process envelops it within an NMVT or a CP-MSU. The decision of which envelope to use for the Alert major vector is based upon a number of factors. Refer to "Sending Alerts" for a description of how the choice is made.

Suppose that CP-MSUs are chosen and multiple Alert conditions have been detected. This process will build an Alert major vector for each Alert condition. As each Alert major vector is constructed, it could be enveloped within a separate CP-MSU, or multiple Alert major vectors could be grouped in a single CP-MSU. Each CP-MSU constructed must be enveloped within a separate MDS-MU. If NMVTs are chosen, only one Alert major vector is permitted within an NMVT.

See "Building an NMVT" in topic 2.10.7.5 for the details of how a process builds an NMVT. See "Building an MDS-MU" in topic 2.10.7.6 for the details of how a process builds an MDS-MU.

After it has completed building an NMVT, this process uses internal MS protocol boundary A to start the process described in "Sending NMVTs" in topic 2.9.5.4.3 (SEND_DATA_SSCP_PU function set) to send the NMVT to CPMS."Protocol Boundary A--Send NMVT" in topic 2.9.1.3 shows the items provided when starting the "Sending NMVTs" process. This process places the following values in these items:

Item 1: A pointer to the NMVT that it has built

Item 2: "NONE" (This data is unsolicited.)

If it has constructed an MDS-MU rather than an NMVT, this process uses internal MS protocol boundary R to start the process described in "Processing Send MDS-MU" in topic 2.9.2.4.8 (MULTIPLE_DOMAIN_SUPPORT function set) to send the MDS-MU to the Alert focal point. "Protocol Boundary R--Send MDS-MU" in topic 2.9.1.13 shows the items provided when starting the "Processing Send MDS-MU" process. This process places the following values in these items:

Item 1: A pointer to the completed MDS-MU

Item 2: "DEFAULT" or "DIRECT" (as indicated by MS_CAPS in "Protocol Boundary V--Current Focal Point" in topic 2.9.1.17)

After this process has started the "Sending NMVTs" or "Processing Send MDS-MU" in topic 2.9.2.4.8 process, it terminates.

2.10.1.5 EP_ALERT Optional Subset 1 (Problem Diagnosis Data)

Subtopics:

- 2.10.1.5.1 Functions Provided
- 2.10.1.5.2 Formats Supported

- 2.10.1.5.3 Implementation Requirements

2.10.1.5.1 Functions Provided

```
EP_ALERT optional subset 1 (Problem Diagnosis Data) provides for the
inclusion of problem diagnosis data in Alerts.  One or more Detailed Data
Extended (X'85') or Detailed Data (X'82') subfields (2) may be used in an
Alert to carry problem determination data (e.g., port number), and problem
diagnosis data, such as an error or malfunction code, that points to a
particular element within a product.  This optional subset is
characterized by use of already supported subfields to provide an
additional function, rather than by support of an additional subvector or
subfield.  As was the case with Problem Determination data, the Alert
architecture says nothing about what problem diagnosis data an implementer
of this subset is required to provide.
```

---

(2) For more information on these subfields see "The Detailed Data Extended (X'85') and Detailed Data (X'82') Common Subfields" in topic 2.10.1.19.5.

2.10.1.5.2 Formats Supported

```
This optional subset involves support for no additional formats.
```

2.10.1.5.3 Implementation Requirements

```
The requirements for implementing the EP_ALERT optional subset 1 (Problem
Diagnosis Data) are described by a model consisting of subsets of one or
more of the LMSs that initiate Alerts.  These LMSs provide the ability to
include problem diagnosis data in the Alert data that they pass to PUMS.
```

2.10.1.6 EP_ALERT Optional Subset 2 (Delayed Alert)

```
This optional subset is not supported for type 2.1 nodes.
```

Subtopics:

- 2.10.1.6.1 Functions Provided
- 2.10.1.6.2 Formats Supported
- 2.10.1.6.3 Implementation Requirements

2.10.1.6.1 Functions Provided

```
EP_ALERT optional subset 2 (Delayed Alert) provides for the following:

°   Detecting the loss of the session between the PU and its controlling
    CP

°   Constructing an Alert (with the Delayed Alert and Held Alert flags
    set) indicating the reason the session was lost, and holding that
    Alert until the session is re-established

°   Logging all NMVTs that would have been sent unsolicited while the
    session with the PU's controlling CP has been unavailable, or passing
    them to EP_ALERT optional subset 3 (Held Alert for PUMS) if it has
    been implemented

°   Detecting that the session between the PU and its controlling CP has
    been re-established, and sending the delayed Alert that has been held
    while the session has been unavailable
```

2.10.1.6.2 Formats Supported

The EP_ALERT optional subset 1 supports the setting of the delayed Alert flag in the Generic Alert Data (X'92') subvector.

2.10.1.6.3 Implementation Requirements

The requirements for implementing the EP_ALERT optional subset 2 (Delayed Alert) are described by a model consisting of a subset of PU configuration services, the physical resources manager LMS and DLC manager LMS, and PU management services.

A Subset of PU Configuration Services:

° Interacts with PUMS via MS protocol boundary 6 as follows:

- Notifies PUMS (the process described in "Sending Delayed Alerts") whenever the PU's session with its SSCP has been established

Subsets of the Physical Resources Manager LMS and DLC Manager LMS:

° Interact with PUMS via, respectively, MS protocol boundaries 1 and 3 as follows:

- Determines that the problem being reported to PUMS has caused the SSCP-PU session to be lost, and indicates this fact to PUMS. When it receives such an indication, the Sending Alerts process within PUMS sets to 1 both the Delayed Alert and Held Alert flags in the Generic Alert Data (X'92') subvector. It is entirely the responsibility of the LMS to determine whether an Alert condition has affected the SSCP-PU session. Examples of how an LMS might know this to be the case are (1) an Alert condition that results in the loss of a node's only link, and (2) an Alert condition that results in the loss of all the sessions out of a node. If the LMS cannot determine whether an Alert condition has affected the SSCP-PU session (e.g., the LMS might know that the condition has resulted in the loss of a link, but have no knowledge whether this was the link supporting the SSCP-PU session), then it passes no indication to PUMS, and the Delayed Alert and Held Alert flags are set to 0.

A Subset of PU Management Services:

° Provides the following processes:

- Processing delayed alerts

  Processes an NMVT as follows when requested by the SEND_DATA_SSCP_PU function set group (protocol boundary C):

  -- Stores it as a delayed Alert if it qualifies

  -- Passes it to EP_ALERT optional subset 3 (Held Alert) if it has been implemented, and if the NMVT cannot be transmitted at this time and cannot be stored as a delayed Alert

  -- Passes it back to the "Sending NMVTs" process on "Sending NMVTs" in topic 2.9.5.4.3 to be sent on the SSCP-PU session if it can be sent at this time.

° Sending delayed Alerts

  Does the following when requested by PU configuration services (via protocol boundary 21):

- Removes a delayed Alert from the delayed Alert control block and passes it to the SEND_DATA_SSCP_PU function set group to be sent on the SSCP-PU session

Processing Delayed Alerts:

(See Figure 10-5 in topic 2.10.1.7.4 and the accompanying text for a complete description of delayed and held Alert processing.

This process is started by the process described in "Sending NMVTs" in topic 2.9.5.4.3 and is passed a completed NMVT. See "Protocol Boundary C--Delayed Alert Processing" in topic 2.9.1.5 for the details of protocol boundary C (Delayed Alert processing).

This process examines the delayed Alert control block (see Figure 10-3) to determine whether it can send the NMVT. Either this control block is empty, or it contains a pointer to a single NMVT that carries a delayed Alert.

° If there is a pointer in the delayed Alert control block (the control block is not empty), this process checks for the existence of the held Alert control block.

  - If the held alert control block is present (if optional subset 3 of the EP_ALERT function set is implemented), this process starts the process described in "Sending Held Alerts" in topic 2.10.1.7.4, passing a pointer to the completed NMVT in item 1. (See "Protocol Boundary B--Held Alert Processing" in topic 2.9.1.4.)

  - If the held Alert control block does not exist, this process starts an undefined process to log the NMVT.

  This process then returns to the process "Sending NMVTs" in topic 2.9.5.4.3 with a non-zero value in returned item 1 indicating that the NMVT cannot be sent at this time on the SSCP-PU session. (See "Protocol Boundary C--Delayed Alert Processing" in topic 2.9.1.5.)

° If the delayed Alert control block is empty, this process examines the current NMVT to determine whether it should be sent or placed in the delayed Alert control block. If this NMVT contains an Alert (X'0000') major vector, and if the delayed Alert flag in the Generic Alert Data (X'92') subvector is set to 1, then the NMVT is stored until the SSCP-PU session on which it is to be sent becomes active. This process places a pointer to the NMVT containing a delayed Alert in the delayed Alert control block, shown in Figure 10-3.

  - If a pointer to the NMVT was placed in the delayed Alert control block, this process returns to the process "Sending NMVTs" in topic 2.9.5.4.3 with a non-zero value in returned item 1 indicating that the NMVT cannot be sent at this time on the SSCP-PU session. (See "Protocol Boundary C--Delayed Alert Processing" in topic 2.9.1.5.)

  - If a pointer to the NMVT was not placed in the delayed Alert control block, this process returns to the process "Sending NMVTs" in topic 2.9.5.4.3 with a 0 value in returned item 1 and expects SEND_DATA_SSCP_PU to send the NMVT on the SSCP-PU session. (See "Protocol Boundary C--Delayed Alert Processing" in topic 2.9.1.5.)

```
 _________________________________
|  pointer to an NMVT containing  |
|  a delayed Alert                |
|_________________________________|
```

Figure 10-3. Delayed Alert Control Block

Sending Delayed Alerts:

This process sends a delayed Alert, if one is contained in the Delayed Alert Control Block, to the PU's SSCP when that session becomes active.

Whenever an SSCP-PU session to a node becomes active, the PU session manager in the node notifies the appropriate process in PU configuration services. The process in PU configuration services then causes this process to be started.

When started, this process examines the delayed Alert control block. If no pointer to an NMVT is present in this control block, then this process terminates. If a pointer to an NMVT (containing a delayed Alert) is there, then this process removes it, leaving the control block empty. It then starts "Sending NMVTs" in topic 2.9.5.4.3 (SEND_DATA_SSCP_PU function set) to send the NMVT. "Protocol Boundary A--Send NMVT" in topic 2.9.1.3 shows the items provided when starting the "Sending NMVTs" process. This process places a pointer to the NMVT to be sent, into item 1, starts the "Sending NMVTs" process, and then terminates.

2.10.1.7 EP_ALERT Optional Subset 3 (Held Alert for PUMS)

Subtopics:

- 2.10.1.7.1 Functions Provided
- 2.10.1.7.2 Formats Supported
- 2.10.1.7.3 Electives
- 2.10.1.7.4 Implementation Requirements

2.10.1.7.1 Functions Provided

EP_ALERT optional subset 3 (Held Alert for PUMS) provides the capability to hold Alerts that occur while the session to the control point is out, and to send the Alerts, with an indication that they have been held, after the session is re-established.

2.10.1.7.2 Formats Supported

This optional subset supports the setting of the held Alert bit to 1 in the Flag byte of the Generic Alert Data (X'92') subvector.

2.10.1.7.3 Electives

The only elective available to this subset is the choice of the number of Alerts to be held. This number can be chosen to be any value acceptable to the implementation. The implementation holds all Alerts that occur until this value is exceeded, at which time new Alerts are logged, and not held.

2.10.1.7.4 Implementation Requirements

The requirements for implementing the EP_ALERT optional subset 3 (Held Alert for PUMS) are described by a model consisting of a subset of PU management services.

A Subset of PU Management Services:

° Sending held Alerts

- Removes a held Alert from the queue (held Alert control block) and passes it to the SEND_DATA_SSCP_PU function set group (via internal MS protocol boundary K) to be sent on the SSCP-PU session (It does this after being started by the SEND_DATA_SSCP_PU function set group via PUMS protocol boundary B.)

- Places an Alert on the held Alert queue (held Alert control block) when started via MS protocol boundary L by the process described in "Processing Delayed Alerts" in topic 2.10.1.6.3, or the process described in "Sending NMVTs" in topic 2.9.5.4.3 (SEND_DATA_SSCP_PU function set). Prior to placing an Alert on the held Alert queue, this process sets the Held Alert and Delayed Alert flags correctly.

Sending Held Alerts:

This process is started to either remove an Alert from the held Alert queue, or to place an Alert on the held Alert queue or log it if queueing is not possible. See "Protocol Boundary B--Held Alert Processing" in topic 2.9.1.4 for the details of protocol boundary B (Held Alert processing).

This process examines the value passed in item 1 to determine what it is being requested to do. If item 1 is 0, it goes to the *Dequeuing Entry*

Point. If item 1 is non-zero, it goes to the *Queuing Entry Point*. Figure 10-4 shows the held Alert control block.

- HEAD_POINTER is a pointer to the first Alert on the queue

- FOOT_POINTER is a pointer to the last Alert on the queue

- COUNT is the number of Alerts in the queue. The maximum number allowed is determined by the elective selected by the implementation.

```
 _____________ _____________ _______
| HEAD_POINTER | FOOT_POINTER | COUNT |
|_____________|_____________|_______|
```

Figure 10-4. Held Alert Control Block

Queuing Entry Point

If the NMVT pointed to by item 1 contains an Alert (X'0000') major vector, this process updates the Delayed and Held Alert flags as follows, to reflect the fact that the Alert has been held. (See Figure 10-5 for a pictorial representation.)

- If both flags are set to 1, and if the initiating process was Sending Delayed Alerts (EP_ALERT optional subset 2), then this process resets the Delayed Alert Flag to 0. An Alert marked as delayed is passed from Sending Delayed Alerts to Sending Held Alerts only if there is already a delayed Alert in the delayed Alert control block; in this case the Delayed Alert flag is no longer appropriate for the current Alert since it was the condition reported by the Alert in the delayed Alert control block, not the condition reported by the current Alert, that prevented the current Alert from being sent to CPMS.

- If both flags are set to 1, and if the initiating process was Sending NMVTs (SEND_DATA_SSCP_PU base subset), then this process leaves the flags unchanged. An Alert marked as delayed is passed from Sending NMVTs to Sending Held Alerts only if Sending NMVTs was unsuccessful in sending a delayed Alert passed to it by Sending Delayed Alerts. In this case the Delayed Alert flag is still appropriate, since the Alert provides information on an Alert condition that (1) caused the SSCP-PU session to be lost, and (2) was, at least temporarily, recovered from.

- If both flags are set to 0, then regardless of the initiating process, this process changes the Held Alert flag to 1. This change indicates that the Alert was held, i.e., was not sent to CPMS immediately upon its creation.

- If the Delayed Alert flag is set to 0 and the Held Alert flag is set to 1, then this process leaves them unchanged. This Alert is one that was previously held, passed to Sending NMVTs for transmission to CPMS, then passed back to Sending Held because it could not be sent. When it finally is sent to CPMS, it should still be marked as a held Alert.

If the NMVT pointed to by item 1 does not contain an Alert (X'0000') major vector, or if the COUNT in the held Alert control block equals the maximum value allowed by the implementation, this process starts an undefined process to log the NMVT and then terminates.

If the NMVT pointed to by item 1 is to be queued, its chain pointer is set to 0. This process then checks the value of FOOT_POINTER. If FOOT_POINTER is zero, the value from item 1 is placed into HEAD_POINTER and FOOT_POINTER. If FOOT_POINTER is nonzero, this process places the value from item 1 into CHAIN_POINTER of the NMVT pointed to by FOOT_POINTER, and overlays FOOT_POINTER with the value from item 1. This process then overlays COUNT with COUNT+1, and terminates.

Dequeuing Entry Point

This process examines CHAIN_POINTER of the NMVT pointed to by HEAD_POINTER. If it points to another NMVT, HEAD_POINTER is overlaid with CHAIN_POINTER, if not, HEAD_POINTER and FOOT_POINTER are set to 0. This process then sets COUNT to COUNT-1, overlays the first three bytes of the NMVT with X'41038D', and starts the process described in "Sending NMVTs" in topic 2.9.5.4.3 (SEND_DATA_SSCP_PU function set) to send the NMVT. "Protocol Boundary A--Send NMVT" in topic 2.9.1.3 shows the parameters specified when starting the "Sending NMVTs" in topic 2.9.5.4.3 process. This process places a pointer to the NMVT to be sent, into item 1. After this process has started the process "Sending NMVTs" in topic 2.9.5.4.3, it terminates.

```
          ____________
         | half-session |
         |_________ ____|
              "    |
              |    |
              |    |
°°°°°°°°°°°°|°°°°|°°°°°°°°°°°°
```

- *Arrow 8*: If the half-session is unable to send the Alert, it indicates this failure to Sending NMVTs.

- *Arrow 9*: When it receives such an indication, Sending NMVTs passes to Sending Held Alerts a pointer to the Alert that could not be sent, via protocol boundary B.

- *Arrow 10*: If an Alert was passed to it to be held, Sending Held Alerts looks to see if there is room for it in the held Alert control block. If there is room, it queues the Alert there.

- *Arrow 11*: If the held Alert control block is full, Sending Held Alerts passed the Alert to an undefined process for local logging.

The preceding discussion covers the creation and sending of Alerts, as well as the saving of them as delayed and held. The processes shown in Figure 10-5 also cooperate to send delayed and held Alerts when communication on the SSCP-PU session is restored:

- *Arrows 4 and 5*: After the SSCP-PU session has been restored, PU Configuration Services apprises Sending Delayed Alerts of this fact. Sending Delayed Alerts checks to see if there is currently an Alert in the delayed Alert control block. If there is, it passes a pointer to this Alert to Sending NMVTs, via protocol boundary A.

- *Arrow 7*: Sending NMVTs passes this Alert to the half-session.

- *Arrows 8 and 9*: If the half-session is again unable to send the Alert, it indicates this fact to Sending NMVTs and, as described above, Sending NMVTs passes a pointer to the Alert to Sending Held Alerts, so that it may be stored in the held Alert control block.

- *Arrows 8 and 9*: If the delayed Alert is successfully transmitted to CPMS, then Sending NMVTs receives a response (positive or negative) from the half-session. At this point it determines whether there is a held Alert control block (i.e., whether EP_ALERT optional subset 3 has been implemented) and, if so, whether this control block currently contains any held Alerts. If there are Alerts currently being held, then Sending NMVTs passes to Sending Held Alerts, via protocol boundary B, an indication that it should remove an Alert from the held Alert control block.

- *Arrows 10 and 12*: Upon receipt of this indication, Sending Held Alerts removes an Alert from the held Alert control block and passes a pointer to it to Sending NMVTs, via protocol boundary A.

- *Arrows 7, 8, and 9*: Sending NMVTs attempts to send this Alerts. If it fails, the Alert is returned to Sending Held Alerts for insertion once again into the held Alert control block.

Sending NMVTs checks for the presence of held Alerts whenever it receives *any* response from CPMS. Thus even if the Held Alert optional subset is implemented without the Delayed Alert optional subset, the process of clearing the held Alert control block is initiated as soon as any management services data is successfully transmitted to CPMS.

Table 10-2. Setting of the Held and Delayed Alert Indicators by Different Alert Senders

Alert sender:	**H=0,D=0**	**H=0,D=1**	**H=1,D=0**	**H=1,D=1**
Supports neither held Alerts nor delayed Alerts	Always sent	Never sent	Never sent	Never sent
Supports held Alerts, does not support delayed Alerts	On Alerts sent immediately	Never sent	On all held Alerts	Never sent
Supports delayed Alerts, does not support held Alerts	On Alerts sent immediately	Never sent	Never sent	On all delayed Alerts
Supports	On Alerts sent	Never	On all held	On all delayed

both held Alerts and delayed Alerts	immediately	sent	Alerts except delayed Alerts	Alerts

Table 10-2 summarizes the setting of the Delayed Alert and Held Alert flags by Alert senders implementing both, one of, or neither of the Delayed Alert and Held Alert optional subsets. Two entries in this table require special comment:

- An Alert sender supporting held Alerts but not supporting delayed Alerts sets H=1,D=0 on *all* held Alerts that it sends. Some of these Alerts may very well report conditions that have caused a loss of communications with CPMS, and so qualify as delayed Alerts, but this Alert sender does not distinguish them from other held Alerts.

- An Alert sender supporting delayed Alerts but not supporting held Alerts sets H=1,D=1 on all delayed Alerts that it sends. Thus even though this sender does not support the sending of held Alerts in general, it nevertheless sets the Held Alert *indicator* on its delayed Alerts.

2.10.1.8 EP_ALERT Optional Subset 4 (Operator-Initiated Alert)

Subtopics:

- 2.10.1.8.1 Functions Provided
- 2.10.1.8.2 Formats Supported
- 2.10.1.8.3 Implementation Requirements

2.10.1.8.1 Functions Provided

EP_ALERT optional subset 4 (Operator-Initiated Alert) provides a mechanism to allow a human operator to initiate the reporting of an Alert condition. This is used to report problems that are not detected automatically by the system. Examples of the types of problems that may only be detected by humans are: poor print or display quality; a machine making an unusual noise; smoke or the smell of something burning coming from a machine. Often the interface provided to the operator will include an area for the operator to enter text describing the Alert condition; if this is the
| case, the text will flow in one or more Self-Defining Text Message (X'31')
| subvectors, and optional subset 6 will also be supported. It is possible, however, to support operator-initiated Alerts without supporting user-entered text on these Alerts, by, perhaps, providing the operator with a menu for choosing among the various code points defined by the architecture to describe the Alert condition. This is why support of the Self-Defining Text Message subvector falls into a different optional subset from that into which support of operator-initiated Alerts falls.

2.10.1.8.2 Formats Supported

The EP_ALERT optional subset 4 supports the setting of the initiation indicator bit to 1 in the Flag byte of the Generic Alert Data (X'92') subvector.

2.10.1.8.3 Implementation Requirements

The requirements for implementing the EP_ALERT optional subset 4 (Operator-Initiated Alert) are described by a model consisting of a subset of the LU LMS. The LMS provides a mechanism to allow a human operator to report Alert conditions to PUMS. These Alerts are reported to PUMS by providing an indication that the Alert was initiated by an operator. This indication is provided at internal MS protocol boundary 12 along with the data required by the base subset.

The LMS is responsible for ensuring that all required data is provided, just as though the LU itself had detected the problem.

2.10.1.9 EP_ALERT Optional Subset 5 (Qualified Message Data)

Subtopics:

- 2.10.1.9.1 Functions Provided
- 2.10.1.9.2 Formats Supported
- 2.10.1.9.3 Implementation Requirements

2.10.1.9.1 Functions Provided

EP_ALERT optional subset 5 (Qualified Message Data) provides for an alternative method of indexing text messages at an Alert receiver. It is especially tailored for the case in which the Alert sender and the Alert receiver are implemented by different instances of the same product, since in such a case (1) the Alert sender can send to the Alert receiver, via the Qualified Message Data subfield, the same index, and the same qualifier data, that it uses to construct the message that notifies its local operator of the existence of the Alert condition, and (2) the Alert receiver will already have the required messages available, since they are needed for notifying its own operator of Alert conditions that it detects locally.

2.10.1.9.2 Formats Supported

This optional subset supports the use of the Qualified Message Data (X'01') subfield within the Detailed Data (X'98') subvector.

2.10.1.9.3 Implementation Requirements

The requirements for implementing the EP_ALERT optional subset 5 (Qualified Message Data) are described by a model consisting of a subset of the physical resources manager LMS, LU LMS, and DLC LMS. These LMSs provide the ability to specify an index identifying a particular message stored at the Alert receiver, as well as one or more variable length text strings to be inserted into the message text by the Alert receiver.

See "The Qualified Message Data (X'01') Subfield" in topic 2.10.1.19.9 for details.

2.10.1.10 EP_ALERT Optional Subset 6 (Text Message)

Subtopics:

- 2.10.1.10.1 Functions Provided
- 2.10.1.10.2 Formats Supported
- 2.10.1.10.3 Implementation Requirements

2.10.1.10.1 Functions Provided

EP_ALERT optional subset 6 (Text Message) provides the capability to send
| in an Alert one or more language-dependent strings, each of text of up to
| 236 characters. In addition to the text string itself, the Alert
| identifies the coded character set in which each string is encoded, its
national language, and the origin of the text string (e.g., operator, application program).

2.10.1.10.2 Formats Supported

This optional subset supports the use of the Self-Defining Text Message (X'31') subvector.

2.10.1.10.3 Implementation Requirements

The requirements for implementing the EP_ALERT optional subset 6 (Text Message) are described by a model consisting of a subset of the physical resources manager LMS, LU LMS, and DLC LMS. These LMSs provide the ability to add to an Alert a text message and indications of its coded character set, its national language, and the nature of its origin. See "Text in Alerts" in topic 2.10.1.19.2 for details.

2.10.1.11 EP_ALERT Optional Subset 7 (LAN Alert)

Subtopics:

- 2.10.1.11.1 Functions Provided
- 2.10.1.11.2 Formats Supported
- 2.10.1.11.3 Implementation Requirements

2.10.1.11.1 Functions Provided

EP_ALERT optional subset 7 (LAN Alert) provides the capability to send Alerts for errors detected at the MAC layer of a token-ring, CSMA/CD, or bridged LAN. See "Alerts for Local Area Networks" in topic A.1 for a list of the Alerts defined for this optional subset.

2.10.1.11.2 Formats Supported

This optional subset supports the use of the LAN Link Connection Subsystem Data (X'51') subvector.

2.10.1.11.3 Implementation Requirements

The requirements for implementing the EP_ALERT optional subset 7 (LAN Alert) are described by a model consisting of a subset of the DLC manager LMS. This LMS provides the ability to send the appropriate Alerts from "Alerts for Local Area Networks" in topic A.1, depending on the type and LAN role of the node in which the DLC manager resides.

2.10.1.12 EP_ALERT Optional Subset 8 (SDLC/LAN LLC Alert)

Subtopics:

- 2.10.1.12.1 Functions Provided
- 2.10.1.12.2 Formats Supported
- 2.10.1.12.3 Implementation Requirements

2.10.1.12.1 Functions Provided

EP_ALERT optional subset 8 (SDLC/LAN LLC Alert) provides the capability to send Alerts for problems detected on SDLC and LAN LLC logical connections. See "SDLC/LAN LLC Alerts" in topic A.2 for a list of the Alerts defined for this optional subset.

2.10.1.12.2 Formats Supported

This optional subset supports the use of the Link Connection Subsystem Configuration Data (X'52') and SDLC Link Station Data (X'8C') subvectors.

2.10.1.12.3 Implementation Requirements

The requirements for implementing the EP_ALERT optional subset 8 (SDLC/LAN LLC Alert) are described by a model consisting of a subset of the DLC manager LMS. This LMS provides the ability to send all of the Alerts from one or both of the sections "LAN LLC Alerts" in topic A.2.1 and "SDLC Alerts" in topic A.2.2, depending on the nature of the link connections in question.

2.10.1.13 EP_ALERT Optional Subset 9 (X.21 Alert)

Subtopics:

- 2.10.1.13.1 Functions Provided
- 2.10.1.13.2 Formats Supported
- 2.10.1.13.3 Implementation Requirements

2.10.1.13.1 Functions Provided

EP_ALERT optional subset 9 (X.21 Alert) provides the capability to send Alerts for problems detected on X.21 (including X.21 Short Hold Mode) link connections. See "X.21 and X.21 Short-Hold Mode Alerts" in topic A.3.1 for a list of the Alerts defined for this optional subset.

2.10.1.13.2 Formats Supported

This optional subset supports the use of the Link Connection Subsystem Configuration Data (X'52') subvector.

2.10.1.13.3 Implementation Requirements

The requirements for implementing the EP_ALERT optional subset 9 (X.21 Alert) are described by a model consisting of a subset of the DLC manager LMS. This LMS provides the ability to send all of the Alerts from "X.21 and X.21 Short-Hold Mode Alerts" in topic A.3.1.

2.10.1.14 EP_ALERT Optional Subset 10 (Hybrid Alert)

This optional subset is not supported for type 2.1 nodes.

Subtopics:

- 2.10.1.14.1 Functions Provided
- 2.10.1.14.2 Formats Supported
- 2.10.1.14.3 Implementation Requirements

2.10.1.14.1 Functions Provided

EP_ALERT optional subset 10 (Hybrid Alert) provides the capability to send a single Alert record that can be processed by both new and old implementations of CPMS. ("New" versus "old" Alert receivers are distinguished by the subvectors they require in an Alert: an old Alert receiver requires the Basic Alert (X'91') subvector, while a new Alert receiver requires the Generic Alert Data (X'92') subvector.) Since the parsing rules for Management Services major vectors specify that all unrecognized subvectors are ignored, the older implementations will process only the older subvectors listed below. Thus a hybrid Alert is, for these implementations, indistinguishable from an Alert containing only the older subvectors.

A new implementation, on the other hand, processes the new subvectors, described in the base subset and the other optional subsets. Either it does not recognize these older subvectors at all (and thereby ignores them), or it is coded to "prefer" the new subvectors to the old ones. To simplify this second option, a single constraint is introduced on the relative positioning of the old and new subvectors; see "Implementation

Requirements" in topic 2.10.1.14.3 below.

2.10.1.14.2 Formats Supported

This optional subset supports the use of the following subvectors:

- Text Message (X'00')
- Hierarchy Name List (X'03')
- Basic Alert (X'91')
- Detail Qualifier (EBCDIC) (X'A0')
- Detail Qualifier (Hexadecimal) (X'A1')

2.10.1.14.3 Implementation Requirements

The requirements for implementing the EP_ALERT optional subset 10 (Hybrid Alert) are described by a model consisting of a subset of the physical resources manager LMS, LU LMS, DLC LMS, and PU management services. These components create the referenced subvectors in precisely the way documented in earlier editions of this book. The subvectors are then included in the Alert major vector with the subvectors created by the base subset and the other optional subsets. If the Hierarchy Name List (X'03') subvector is included in the Alert major vector, it must appear *after* the Hierarchy/Resource List (X'05') subvector. Otherwise there are no constraints on the relative ordering of the subvectors created by this subset and those created by the base subset and the other optional subsets.

2.10.1.15 EP_ALERT Optional Subset 11 (X.25 Alert)

Subtopics:

- 2.10.1.15.1 Functions Provided
- 2.10.1.15.2 Formats Supported
- 2.10.1.15.3 Implementation Requirements

2.10.1.15.1 Functions Provided

EP_ALERT optional subset 11 (X.25 Alert) provides the capability to send Alerts for problems detected on X.25 link connections. See "Alerts for X.25 Link Connections" in topic A.4 for a list of the Alerts defined for this optional subset.

2.10.1.15.2 Formats Supported

This optional subset supports the use of the Link Connection Subsystem Configuration Data (X'52') subvector.

2.10.1.15.3 Implementation Requirements

The requirements for implementing the EP_ALERT optional subset 11 (X.25 Alert) are described by a model consisting of a subset of the DLC manager LMS. This LMS provides the ability to send all of the Alerts from "Alerts for X.25 Link Connections" in topic A.4.

2.10.1.16 EP_ALERT Optional Subset 12 (Held Alert for CPMS)

Subtopics:

- 2.10.1.16.1 Functions Provided
- 2.10.1.16.2 Formats Supported
- 2.10.1.16.3 Electives
- 2.10.1.16.4 Implementation Requirements

2.10.1.16.1 Functions Provided

EP_ALERT optional subset 12 (Held Alert for CPMS) provides the capability to hold Alerts that occur when an Alert focal point is not available, and to send the Alerts, with an indication that they have been held, when a focal point is once again available.

2.10.1.16.2 Formats Supported

This optional subset supports the setting of the held Alert bit to 1 in the Flag byte of the Generic Alert Data (X'92') subvector.

2.10.1.16.3 Electives

The only elective available to this subset is the choice of the number of Alerts to be held. This number can be chosen to be any value acceptable to the implementation. This value represents the maximum number of elements which can be placed in the held Alert queue. When the maximum number is exceeded, adding new alerts to the queue causes the oldest elements to be deleted. Thus, if the number of Alerts which can be held is "n," when Alert "n+1" is enqueued, Alert "1" will be deleted from the queue. The last "n" Alerts, rather than the first "n" Alerts, are held.

2.10.1.16.4 Implementation Requirements

The requirements for implementing the EP_ALERT optional subset 12 (Held Alert for CPMS) are described by a model consisting of a subset of CP management services.

A Subset of CP Management Services

```
      _____________________
     |                     |
     |MULTIPLE_DOMAIN_SUPPORT|
     |  function set group   |
     |_____________________|
        "       |       "
          |       |       |
°°°°°°°°°|°°°°°°°|°°°°°°°|°°°°°°°°°
```

|_____|

Figure 10-6. Overview of Delayed Alert Processing in a Control Point

Figure 10-6 illustrates the components involved in the implementation of held Alert for a control point. In the figure, the letters next to arrows identify the protocol boundaries involved.

The basic premise of held Alerts for CPMS is that MULTIPLE_DOMAIN_SUPPORT will provide notification, via an MDS error message, if an MDS-MU cannot be delivered. Each Alert is cached in the held Alert queue before sending. If an MDS error message is returned for an Alert, the cached copy is marked as HELD, and will be re-sent when a focal point is once again available.

Much of the following information on this subset assumes, for simplicity, that one Alert is present in a CP-MSU. However, since multiple Alerts are permitted in a CP-MSU (and thus an MDS-MU), each statement below that references *the* Alert in a CP-MSU or MDS-MU applies to *each* Alert in the CP-MSU or MDS-MU.

Sending Alerts

- Maintains awareness of current Alert focal point (as indicated by MS_CAPS via "Protocol Boundary V--Current Focal Point" in topic 2.9.1.17).

- Builds MDS-MUs for Alert CP-MSUs and sends them to the current focal point via MDS (using "Protocol Boundary R--Send MDS-MU" in topic 2.9.1.13).

- Causes each Alert sent to be logged in a historical database.

- Passes each MDS-MU to Held Alert Processing so that it can be added to the held Alert queue.

Held Alert Processing

- Adds each Alert sent to the held Alert queue.

  This queue is a cache for Alerts which are *potentially* held Alerts. The MDS-MU is kept so that subsequent MDS error messages can be correlated to the correct Alerts. The size of the queue is an implementation option, and is not specified by the MS architecture. If the queue can contain "n" elements, these represent the last "n" Alerts sent.

- When an MDS error message is received from MDS (via "Protocol Boundary Q--MDS-MU Received" in topic 2.9.1.12), indicating that an MDS-MU could not be delivered, the impacted Alert is found in the held Alert queue, and is marked as HELD. If a match is not found, no action is taken (the indicated Alert presumably is no longer cached in the held Alert queue). No other Alerts are marked, since MDS error messages will be forthcoming for any Alerts sent subsequent to the indicated one (MDS creates MDS error messages for every undeliverable MDS-MU).

- When MS_CAPS has a new focal point, "Protocol Boundary V--Current Focal Point," is passed to this process with the focal point name. All undelivered Alerts in the held Alert queue are processed:

  1. The MDS-MU is removed from the queue.

  2. The Generic Alert (X'92') subvector is located in the Alert (X'0000') major vector in the CP-MSU portion of the MDS-MU. The Held Alert Flag is set to 1 and the Delayed Alert Flag is set to 0 (Delayed Alerts are defined only for PUMS implementations). See Table 10-2 in topic 2.10.1.7.4 for a summary of the Held and Delayed Alert flags.

  3. An MDS-MU is built for the modified Alert, with the new focal point as the destination location name. The origin location name and Agent Unit-of-Work Correlator (X'1549') GDS variable are copied from the original MDS-MU.

  4. The new MDS-MU is added to the held Alert queue.

  5. The MDS-MU is passed to MDS ("Protocol Boundary R--Send MDS-MU")

for routing to the focal point.

```
 ____________________________________________________________________
|                                                                    |
|     LMS      Database     Held Alert       Sending      MS_CAPS     MD|
|              Support      Processing        Alerts                    |
| ___________________________________________________________________|
|      .          .            .               .            .          .|
|      .          .            .               . Current FP .          .|
| 1    .          .            .               o___________o          .|
|      .          .   (Alert data)              .            .          .|
| 2    o_____________________________________ÿo            .          .|
|      .          .    (add to Alert log)       |            .          .|
| 3    .          o_____________________________|            .          .|
|      .         (add to queue of recent Alerts)|            .          .|
| 4    .          .            o_______________|            .          .|
|      .          .            .               | Send MDS-MU to FP     .|
| 5    .          .            .               |_______________________ÿo|
|      .          .            .               .     MDS Error Message .|
| 6    .          .            .               .           o_________o|
|      .          .            .               .     MDS Error Message ||
| 7    .          .            .               o_________________________||
|      .          .            .MDS Error Msg  |            .          .|
| 8    .          .            o_______________|            .          .|
|      .          .   (Alert data)              .            .          .|
| 9    o_____________________________________ÿo            .          .|
|      .          .    (add to Alert log)       |            .          .|
| 10   .          o_____________________________|            .          .|
|      .          .          (add to held queue)|            .          .|
| 11   .          .            o_______________|            .          .|
|      .          .            .               . Current FP .          .|
| 12   .          .            .               o___________o          .|
|      .          .            .  Current FP   |            .          .|
| 13   .          .            o_______________|            .          .|
|      .          .            |         Send MDS-MU to new FP         .|
|      .          .            |_____________________________________ÿo|
|      .          .            |         Send MDS-MU to new FP         .|
|      .          .            |_____________________________________ÿo|
|      .          .   (Alert data)              .            .          .|
| 14   o_____________________________________ÿo            .          .|
|      .          .    (add to Alert log)       |            .          .|
|      .          o_____________________________|            .          .|
|      .          .          (add to held queue)|            .          .|
|      .          .            o_______________|            .          .|
|      .          .            .               | Send MDS-MU to FP     .|
|      .          .            .               |_______________________ÿo|
|      .          .            .               .            .          .|
|                                                                        |
|________________________________________________________________________|
```

Figure 10-7. Held Alert Processing with Multiple-Domain Support

Figure 10-7 illustrates how held Alert processing is implemented by CPMS. The numbers in the figure correspond to the numbered steps which follow.

1. MS_CAPS provides the name of the focal point in "Protocol Boundary V--Current Focal Point." This will be the destination of all Alerts processed by Sending Alerts.

2. An LMS passes Alert data to Sending Alerts, which formats an Alert major vector. The Alert is encapsulated in a CP-MSU, and an MDS-MU is built with the focal point as the destination.

3. The Alert is recorded in the historical log.

4. The MDS-MU is passed to Held Alert Processing to be added to the held Alert queue.

5. The MDS-MU is passed to MDS for routing to the focal point ("Protocol Boundary R--Send MDS-MU").

6. An MDS error message is received by MDS indicating that an MDS-MU could not be delivered to the destination. Information about the MDS error message is passed to MS_CAPS so that any recovery, if appropriate, can be started. In this case, since the destination which could not be reached is the Alert focal point, MS_CAPS will attempt to contact the backup focal point, if that option is supported and one is known.

7. The MDS error message is delivered (over "Protocol Boundary Q--MDS-MU Received"). Sending Alerts will now refrain from attempting to send any more Alerts until MS_CAPS indicates that a focal point is available.

8. The MDS error message is passed to Held Alert Processing, which searches the held Alert queue for a previously sent Alert with a matching Agent Unit-of-Work Correlator (X'1549') GDS variable. If a match is found, the entry is marked HELD. If a match is not found, no action is taken (the indicated Alert presumably is no longer cached in the held Alert queue).

Since every Alert sent but not delivered will cause an MDS error message to be returned, no other Alerts in the queue are marked.

9. An LMS in the node detects another Alert condition and passes Alert data to Sending Alerts. The appropriate Alert major vector is constructed. The Alert is encapsulated in a CP-MSU, and an MDS-MU is built without a destination.

10. The Alert is recorded in the historical log.

11. The MDS-MU is passed to Held Alert Processing to be added to the held Alert queue. An indication is passed that the Alert is to be marked as HELD. There is no attempt to send the Alert.

12. Eventually, a new focal point is acquired and MS_CAPS passes the appropriate information via "Protocol Boundary V--Current Focal Point." The focal point is remembered by Sending Alerts for all subsequent Alerts.

13. The focal point name is passed to Held Alert Processing, which sends each HELD Alert in the queue to the new focal point. The Held Alert flag in the Generic Alert (X'92') subvector is set to 1 in each Alert, and a new MDS-MU is built with the new focal point as the destination location. The Agent Unit-of-Work Correlator (X'1549') GDS variable and origin location name is copied from the original MDS-MU. This is appropriate since the unit of work correlator in this case represents the creation of the Alert, and not the sending of the Alert. The old queue entry is deleted and the new MDS-MU is added to the end of the queue.

    In Figure 10-7, two Alerts were held: the first, which could not be delivered to the previous focal point, and which caused an MDS error message; and the second, which was created after the focal point was considered lost. Each is sent.

14. Another LMS passes Alert data to Sending Alerts. An Alert is built, logged, enqueued, and sent just as in the initial case.

2.10.1.17 EP_ALERT Optional Subset 13 (Resolution Notification Support)

Subtopics:

- 2.10.1.17.1 Functions Provided
- 2.10.1.17.2 Formats Supported
- 2.10.1.17.3 Implementation Requirements

2.10.1.17.1 Functions Provided

This subset provides for the unsolicited notification of the correction of a error condition. It provides the following:

° Detection of resolution of an error condition for a resource controlled by this node

° Constructs an NMVT containing a Resolution (X'0002') major vector, or an MDS-MU containing one or more Resolution (X'0002') major vectors

° Passes the NMVT to the SEND_DATA_SSCP_PU function set group, to be sent unsolicited, to all control points that control the resource to which the resolution condition applies

° Passes the MDS-MU to the MULTIPLE_DOMAIN_SUPPORT function set group, to be sent unsolicited to this node's current Alert focal point

Support of certain *types* of Resolution notifications, and of certain types of data within them, is optional. Also optional is the saving of resolution notifications when the SSCP-PU session is unavailable and sending them when the session with the SSCP is re-established. When this subset is implemented, all of the optional support provided by optional subsets 3 through 6 and optional subset 12 is to be extended to Resolution major vectors in a like manner.

As with Alerts, resolution data falls into two broad categories: data indicating the nature of the correction to the problem, and data identifying the component that was corrected. Some portions of this data are textual; these are displayed to the network operator just as they are received. Others are code points that index particular strings of stored

text; these strings of text are displayed in a predetermined manner, upon receipt of the code points that index them.

The Resolution major vector shares many of the same characteristics with the Alert major vector, including shared subvectors and subfields that are identified using the same key (but that perform different, although related, roles). For example, the Resolution Data (X'92') subvector describes the problem that was resolved, whereas the Generic Alert Data (X'92') subvector describes a problem that has occurred. The Actual Causes (X'93') subvector occurs in the Resolution major vector, whereas the Probable Causes (X'93') subvector occurs in the Alert. In the Alert, the "Causes" subvectors identify possible causes for the error condition; in the Resolution major vector the "Causes" subvectors identify the *actual* causes of the problem.

The same code point tables are shared between Resolution subvectors/subfields and Alert subvectors/subfields with the same key. For example, the Actual Causes and Probable Causes subvectors use the same code points. However, the Actual Actions (X'86') subfield is used in the Resolution major vector instead of the Recommended Actions (X'81') subfield (which is used in Alerts). The code point tables used for the Actual Actions and Recommended Actions subfields are different (as are the subfield keys).

This subset can provide correlation between an Alert major vector describing a problem, and the Resolution major vector indicating that the problem was resolved. This correlation is provided via a unique token identifying the problem. When supported, this unique token is carried in the Incident Identification (X'4A') subvector of both the Alert and Resolution major vectors.

2.10.1.17.2 Formats Supported

This optional subset supports the use of the following subvector in the Alert major vector.

- An optional Incident Identification (X'4A') subvector transporting a unique token (an incident identifier) for the Alert condition that has occurred.

This optional subset supports the use of the following subvectors in the Resolution major vector.

- A Date/Time (X'01') or Relative Time (X'42') subvector, depending on the node's time-stamping capabilities.

- An SNA Address List (X'04') subvector if the corrected resource was an SNA-addressable entity other than the sending PU.

  Note: This subvector is constructed only for Resolution major vectors that flow in an NMVT. Resource identification for Resolution major vectors that are transported in CP-MSUs is always done in the Hierarchy/Resource List (X'05') subvector.

- A Hierarchy/Resource List (X'05') subvector logically identifying the resource that was corrected.

  This subvector is constructed for an NMVT resolution notification if the corrected resource was not located in the products implementing the PU, and cannot be represented in the SNA Address List subvector.

  This subvector is also constructed if the corrected resource was a component of the hardware product implementing the PU.

  This subvector is always constructed for a CP-MSU resolution notification.

- A Product Set ID subvector identifying the PU or CP sending the resolution notification.

- A second Product Set ID (X'10') subvector physically identifying the corrected resource, if the resource was a hardware or software product and is not one of the products implementing the PU

- An optional Incident Identification (X'4A') subvector transporting a unique token (an incident identifier) for the Alert condition that was resolved. It is possible that multiple Alerts could have been sent for seemingly different problems. Yet it may later be discovered that one particular problem caused a group of Alerts to be sent. In this case, an incident identifier for each resolved Alert condition may be included in the X'4A' subvector. If the incident identifiers for all the resolved conditions do not fit within a single X'4A' subvector,

then multiple X'4A' subvectors may be included in the Resolution major vector.

- A Resolution Data (X'92') subvector describing the problem that was resolved. This subvector uses the same code points as the Generic Alert Data (X'92') Alert subvector.

- An Actual Causes (X'93') subvector identifying the actual causes of the resolved problem. This subvector uses the same code points as the Probable Causes (X'93') Alert subvector.

The X'92' and X'93' subvectors are included in *every* Resolution notification. The following subvectors are included *when appropriate* to report more detail about the correction that was made.

- An Actual User Causes (X'94') subvector identifying the actual user causes of the resolved problem and the actual actions taken to correct the problem. This subvector uses the same "causes" code points as the User Causes (X'94') Alert subvector.

- An Actual Install Causes (X'95') subvector identifying the actual install causes of the resolved problem and the actual actions taken to correct the problem. This subvector uses the same "causes" code points as the Install Causes (X'95') Alert subvector.

- An Actual Failure Causes (X'96') subvector identifying the actual failure causes of the resolved problem and the actual actions taken to correct the problem. This subvector uses the same "causes" code points as the Failure Causes (X'96') Alert subvector.

- An Actual Cause Undetermined (X'97') subvector identifying the actual actions taken to correct the problem when no actual user, install, or failure causes for the problem could be determined.

The Actual Actions (X'86') Resolution common subfield is used within the X'94'-X'97' Resolution subvectors to identify the actual actions performed instead of the Recommended Actions (X'81') subfield, which is used in Alerts. The code points defined for the Actual Actions subfield are *different* from those defined for the Recommended Actions subfield.

Depending on the code points used, the X'82'-X'85' Resolution common subfields may be included within the X'94'-X'97' Resolution subvectors. The X'82X'-X'85' Resolution common subfields are identical in structure to the X'82'-X'85' Alert common subfields, and are thus used similarly.

- A Detailed Data (X'98') subvector transporting product specific detailed data about the corrected problem may also be included in the Resolution major vector when appropriate.

2.10.1.17.3 Implementation Requirements

The requirements for implementing optional subset 13 are the same as the requirements for implementing the base subset for EP_ALERT. The difference is that the model supports exchange of Resolution data and records across the same protocol boundaries.

As with Alerts, CP-MSUs constructed by EP_ALERT may contain multiple Resolution major vectors. A CP-MSU constructed by EP_ALERT may also contain one or more Alert major vectors as well as one or more Resolution major vectors. If an Alert and a Resolution major vector with the same incident identifier are both present in the same CP-MSU, then the Alert major vector always *precedes* the Resolution major vector.

CP-MSUs that are forwarded to EP_ALERT by second-level application programs cannot, however, contain multiple Alert major vectors or multiple Resolution major vectors. For more information see "EP_ALERT Optional Subset 14 (Operations Management Support)" in topic 2.10.1.18.

Relationship with Other Optional Subsets:

The functions provided by the following optional subsets are extended to support resolution notifications. The changes required to support resolution notifications are included for each listed subset.

- Optional Subset 3 (Held Alert for PUMS), when extended, provides held processing for resolution notifications (in addition to Alerts). This extension supports the setting of the Held Resolution Notification flag to 1 in the Flags field of the Resolution Data (X'92') subvector.

Delayed resolution notifications are not supported (there is no delayed flag in the Resolution Data flags field). Thus the delayed processing discussion in optional subset 3 does not apply to resolution notifications.

- Optional Subset 4 (Operator-Initiated Alert), when extended, provides a mechanism to allow a human operator to initiate the reporting of a resolution notification (in addition to an Alert condition). This extension supports the setting of the Initiation Indicator flag to 1 in the Flags field of the Resolution Data (X'92') subvector.

- Optional Subset 5 (Qualified Message Data), when extended, provides for an alternative method of indexing text messages at a resolution notification receiver.

- Optional Subset 6 (Text Message), when extended, provides support for one or more Self-Defining Text Message (X'31') subvectors in a resolution notification.

- Optional Subset 12 (Held Alert for CPMS), when extended, provides held processing for resolution notifications (in addition to Alerts). This extension supports the setting of the Held Resolution notification flag to 1 in the Flags field of the Resolution Data (X'92') subvector.

Note that Optional Subset 14 (Operations Management Support) is not included in the list above since it describes a specific method for processing the Resolution major vector. The remaining optional subsets are not supported for resolution notifications.

2.10.1.18 EP_ALERT Optional Subset 14 (Operations Management Support)

Subtopics:

- 2.10.1.18.1 Functions Provided
- 2.10.1.18.2 Formats Supported
- 2.10.1.18.3 Implementation Requirements

2.10.1.18.1 Functions Provided

This subset is only for CPMS implementations sending CP-MSU Alerts. The following function-set application programs are required if this optional subset is implemented:

- EP_OPERATIONS_MGMT

- An operations management focal-point application program

Implementation of this subset allows operations management commands to be initiated as a direct result of an Alert or Resolution condition, thereby providing problem bypass and recovery capability in the network.

EP_ALERT optional subset 14 provides the capability for second-level application programs to forward to EP_ALERT a CP-MSU containing one Routing and Targeting Instructions (X'154D') GDS variable in addition to one of the following:

1. One Alert major vector

2. One Resolution major vector

3. One Alert major vector and one Resolution major vector. The Alert and the Resolution major vector must have the same incident identifier. The Alert major vector must precede the Resolution major vector.

The Routing and Targeting Instructions (RTI) GDS variable must precede the Alert and/or Resolution major vector in the CP-MSU.

Only one Alert and/or Resolution major vector is permitted so that a one-to-one correspondence is maintained between each Alert/Resolution condition and the operations management command that is generated by an application program the operations management focal point serves. Below is an explanation of this process.

A CP-MSU (one of the types discussed above) is forwarded to EP_ALERT, across application-program protocol boundary A-5, from a second-level

application program served by EP_ALERT.

EP_ALERT envelops the CP-MSU in an MDS-MU and sends it to the Alert focal point. The Alert focal point then forwards the MDS-MU to the operations management focal point in the same node. The operations management focal point then forwards the entire MDS-MU to an application program it serves, based on the application program names in the RTI. If no destination application name information is present in the RTI, then the MDS-MU is forwarded to the application program that has registered for Alert and/or Resolution major vector MDS-MUs sent from the application program named in the origin application name in the RTI. Such registration is implementation-specific. Refer to the format descriptions below for information concerning application program names in the RTI.

The served application may initiate problem recovery procedures by building an operations management command and sending it to the application program that sent the Alert and/or Resolution major vector (with an RTI) to EP_ALERT originally. See "Operations Management Function Set" in topic 1.7.3 for a description of the Operations Management commands.

2.10.1.18.2 Formats Supported

The EP_ALERT function set does not interpret any of the RTI subvectors. Their description is included here to only explain the usage of the RTI and its relevance to the interface between the problem management and operations management architectures.

The RTI (X'154D') GDS variable is composed of two subvectors:

- Name List (X'06')
- Correlator Control (X'0D')

Only the Name List subvector is used in this optional subset.

The Name List subvector may contain the following subfields:

- Origin Application Name (X'60'): The application program name that passed the CP-MSU to EP_ALERT. If the Cascaded Resource Name List subfield (see below) is present, it identifies the application program that actually passed the CP-MSU to EP_ALERT. But the Origin Application Name subfield still identifies the application program that initiated the process. The application program named in this subfield will be placed in the Destination Application Name subfield (see below) in the RTI of the operations management command that results from the Alert and/or Resolution major vector being forwarded. This subfield is required to be present within the Name List subvector. It is the only required subfield.

- Origin Instance Identifier (X'80'): If multiple instances of the application program named in the Origin Application Name subfield exist, then the instance that sent the Alert and/or Resolution major vector may be identified here. The application program named in this subfield will be placed in the Destination Instance Identifier subfield (see below) in the RTI of the operations management command that results from the Alert and/or Resolution major vector being forwarded.

- Destination Application Name (X'50'): The application program, served by the operations management focal point, that is to receive the forwarded Alert and/or Resolution major vector. It is responsible for initiating the construction of the operations management command to be sent. The application program named in this subfield will be placed in the Origin Application Name subfield in the RTI of the operations management command that results from the Alert and/or Resolution major vector being forwarded.

- Destination Instance Identifier (X'70'): If multiple instances of the application program named in the Destination Application Name subfield exist, then the instance that is the target may be identified here. The application program named in this subfield will be placed in the Origin Instance Identifier subfield in the RTI of the operations management command that results from the Alert and/or Resolution major vector being forwarded.

- Cascaded Resource Name List (X'10'): The application program named in the Origin Application Name subfield in the RTI of the CP-MSU sent to EP_ALERT may not be able to communicate directly with EP_ALERT. It may need to forward the Alert and/or Resolution major vector along a path of one or more hierarchically related application programs. The Cascaded Resource Name List subfield is used for this purpose. The first name in the list identifies the application program that actually passed the Alert and/or Resolution CP-MSU across protocol

boundary A-5 to EP_ALERT. The last name in the list identifies the application program to which the application program named in the Origin Application Name subfield first sent the Alert and/or Resolution major vector. This subfield provides path information related only to the application programs served by EP_ALERT. This subfield is retained for the RTI of the operations management command; but, the order of the application program names it lists is reversed. This is done because the "return path" of the command will be an exact reversal of the path the original Alert and/or Resolution major vector took to get to EP_ALERT.

2.10.1.18.3 Implementation Requirements

A Subset of Management Services:

° Receiving a CP-MSU

- Receives, across MS application program protocol boundary A-5, a CP-MSU containing an Alert and/or Resolution major vector and an RTI.

° Sending the MDS-MU

- Envelops the CP-MSU in an MDS-MU and then starts the process described in "Processing Send MDS-MU" in topic 2.9.2.4.8 to send the MDS-MU to the Alert focal point.

2.10.1.19 Details of the Alert Encodings

This section contains detailed discussions of several topics related to the encoding of the Alert subvectors. Refer to the Alert subvectors in *SNA Formats*, as you read this section.

Subtopics:

- 2.10.1.19.1 Default/Replacement Code Points
- 2.10.1.19.2 Text in Alerts
- 2.10.1.19.3 Identification of Unique Alerts
- 2.10.1.19.4 Reserved Ranges of Alert Code Points
- 2.10.1.19.5 The Detailed Data Extended (X'85') and Detailed Data (X'82') Common Subfields
- 2.10.1.19.6 Product Set ID Indexing
- 2.10.1.19.7 Resource List Indexing
- 2.10.1.19.8 Associating Code Points with X'82', X'83', X'84', and X'85' Subfields
- 2.10.1.19.9 The Qualified Message Data (X'01') Subfield
- 2.10.1.19.10 Hierarchy Information in Alerts
- 2.10.1.19.11 Correlation in Alerts
- 2.10.1.19.12 Examples of Physical and Logical Identification of the Origin of an Alert Condition

---

2.10.1.19.1 Default/Replacement Code Points

The following types of code points in a generic Alert share the same structure and semantics:

° Alert Description Code, in the X'92' subvector

° Probable Causes, in the X'93' subvector

° User Causes, in the X'94' subvector

° Install Causes, in the X'95' subvector

° Failure Causes, in the X'96' subvector

° Recommended Actions, in the X'81' subfield

These code points are termed *default/replacement* code points; their structure is represented in Figure 10-8.

```
 _____________________ _________________________
|        Byte 1       |          Byte 2         |
```

```
| (indexes default text) | (indexes replacement text) |
|_______________________|____________________________|
```

Figure 10-8. Format for Default/Replacement Alert Code Points

Each code point is two bytes long, with the second byte serving to qualify the information carried in the first byte. A code point with a value of X'00' in the second byte is termed a *default* code point. It corresponds to a high-level description of an Alert condition, cause or recommended action. A default code point is typically followed by one or more *replacement* code points, that have the same first byte but a non-zero second byte. These code points correspond to more specific descriptions, that fall under the high-level description indexed by the default code point. Default/replacement code points are illustrated in Table 10-3. The specific descriptions "RUN CONSOLE TEST," etc. can be seen to fall under the more general description "RUN APPROPRIATE TEST."

Table 10-3. Default and Replacement Text for Recommended Action Code Points

Code Point	Default Text	Replacement Text
X'0400'	RUN APPROPRIATE TEST	RUN APPROPRIATE TEST
X'0401'	" "	RUN CONSOLE TEST
X'0402'	" "	RUN CONSOLE LINK TEST
X'0403'	" "	RUN MODEM TESTS

Default/replacement code points allow for certain flexibility in displays at an Alert receiver. When it receives a replacement code point, an Alert receiver is allowed to display either the text for this code point itself or the text for the default code point above it. Upon receipt of a Recommended Action of X'0401', for example, an Alert receiver may display either "RUN CONSOLE TEST" or the more general "RUN APPROPRIATE TEST." There are two reasons why an Alert receiver might choose to display default text when it receives a replacement code point:

- If a new replacement code point has been added to the architecture and is being sent by an Alert sender, but the receiving product has not yet had its tables updated, then it can still communicate some information by displaying the default text. Suppose, for example, that a new Recommended Action code point X'0404', "RUN TEXT-X TEST," has been added to the architecture. If an Alert receiver receives this code point before it has updated its tables to include the new text, it can still display the default text "RUN APPROPRIATE TEST." Later, after its table is updated, it will be able to display the more informative "RUN TEST-X TEST," but in the interim it has been able to extract useful information from the code point and display this information to a network operator without having had to synchronize its table update with the release of the new sending product.

- An Alert receiver running on a small system might elect to display *only* default text, in order to keep the size of its tables small. Or it might elect to display only default text for some code points, perhaps User, Install, and Failure Causes, while maintaining the full tables of default and replacement text for Alert Description, Probable Causes and Recommended Actions. In any case, default/replacement code points allow an Alert sender to send identical Alerts to Alert receivers of different capabilities.

An additional advantage of default/replacement code points follows simply from the fact that they are code points: An Alert receiver can provide national language support for them, by providing different tables of text for the different languages. It is even possible for a receiver to provide full tables of default and replacement text in certain languages, but to provide shorter tables of default text only for others.

Default/replacement code points for User, Install, and Failure Causes, and for Recommended Actions, carry an additional piece of information in the way they are encoded. Since these code points provide for the insertion of strings of qualifier text, flowing in the Detailed Data Extended (X'85') or Detailed Data (X'82') subfield, for the insertion of product data, identified in the Product Set ID Index (X'83') subfield, or for the insertion of resource identification data, identified in the Resource List Index (X'84') subfield, into the stored text that the code points index, the code points must themselves specify implicitly how many of these subfields are to be associated with them by the receiver processing the Alert. See "Associating Code Points with X'82', X'83', X'84', and X'85' Subfields" in topic 2.10.1.19.8 for further details.

2.10.1.19.2 Text in Alerts

Textual data is carried in Alerts in three places:

- In the Detailed Data Extended (X'85') and Detailed Data (X'82') common subfields

- In the Self-Defining Text Message (X'31') common subvector

- In other management services common subvectors:

  - Hierarchy/Resource List (X'05')

  - Product Set ID (X'10')

Textual data from all of these sources may be combined by an Alert receiver to create a single display.

To specify fully how a field containing textual data is to be interpreted, three pieces of information about the field are needed:

- The *character set* used in it.

- The *code page* according to which the data in it is encoded.

These first two pieces of information jointly define the *Coded Graphic Character Set* for the field in question.

- The *symbol strings* allowed in it.

Thus the following data might be specified for a particular field:

- It can contain any upper-case letters and/or numerals. (This by itself says nothing about the bit patterns that represent the characters.)

- It is encoded according to code page xxx of some standard. (This correlates characters with bit patterns, but the only limit it places on the character set is the full set of characters represented on code page xxx.)

Taken together, the first two pieces of information say that the field contains upper-case letters and numerals, encoded as they are on code page xxx. This defines the Coded Graphic Character Set for the field.

- The first character in any instance of the field must be either a numeral or an 'X'. (This characterizes the strings that can appear in the field, rather than the individual characters.)

In the Alert architecture, the Detailed Data Extended (X'85') and Detailed Data (X'82') subfields contain a 1-byte Data Encoding field indicating how an Alert receiver is to display the detailed data. The following values of the Data Encoding field are defined for the detailed data subfields:

X'00': Hexadecimal. The detailed data is treated as a hexadecimal number. If, for example, the detailed data is X'0107', then "0107" is displayed.

X'01': Binary. The detailed data is treated as an unsigned integer, for which the decimal equivalent is displayed. In this case the detailed data X'0107' results in the display "263."

X'10': Hexadecimal code point. The detailed data is a hexadecimal code point, to be treated as an index to a text string stored at the Alert receiver. The text string is displayed on the Alert receiver screen in place of the code point comprising the detailed data.

X'11': Coded graphic character set 00640-00500 plus three additional characters. This coded graphic character set is documented in the *SNA Formats*. It contains 26 upper and lower case letters, 10 numerals, and 19 special characters. Data Encoding X'11' refers to the union of this coded graphic character set with the following three characters and hexadecimal values:

"$" - dollar sign: X'5B'

"#" - number sign: X'7B'

"@" - at sign: X'7C'

coded graphic character set 00640-00500 is displayable on the vast majority of EBCDIC terminals, and capable of being entered from

the vast majority of EBCDIC keyboards. The three additional characters are included for migration; they are present in the old SNA character set type A, and thus may appear in such SNA designators as LU or CP name.

The Self-Defining Text Message (X'31') subvector potentially supports *any* coded graphic character set, so the mechanism for identifying the coded graphic character set in the X'31' subvector must be a general one. Thus this subvector carries a 4-byte field, with 2 bytes identifying a character set and two bytes identifying a code page.

The Self-Defining Text Message subvector also carries an indication of the national language of the text message being transported. With this information, an Alert receiver is able to route messages to different operators based on the set of languages that each operator understands.

2.10.1.19.3 Identification of Unique Alerts

One requirement on the generic Alert architecture is that a mechanism be provided for identifying individual Alerts, so that customers can alter the presentation for selected Alerts. This requirement is met via the *Unique Alert Identifier*, which, as Figure 10-9 indicates, consists of two parts, the Product ID and the Alert ID Number.

```
 Unique Alert Identifier:
 _____________________________ ____________________________________
|          Product ID         |             Alert ID Number        |
|(from the Alert sender's PSID)|(from the Generic Alert Data (X'92') SV)|
|_____________________________|____________________________________|
```

Figure 10-9. The Unique Alert Identifier

The Alert ID Number, which flows as bytes 7-10 of the Generic Alert Data (X'92') subvector, serves to differentiate the Alerts sent by one Alert sender. Also, since an Alert ID Number serves to summarize the contents of the fields in an Alert that describe the Alert condition being reported, an Alert ID Number may be associated with a particular Alert condition without regard to the product sending the Alert. This latter feature of Alert ID Numbers is especially significant in an environment that features "common building blocks," i.e., units of hardware, microcode, or software that are incorporated into different products. An Alert condition detected and reported by such a common building block will result in Alerts with identical Alert ID Numbers being sent by each of the different products incorporating the common building block: the Alert ID Number will make it possible for an operator (or application program) at an Alert receiver to identify these Alerts from different products as ones that are in fact reporting different instances of the same Alert condition.

```
                 ____________                          __________
                |Create input|                        |32_bit CRC|
 Alert MV _____ÿ|to algorithm|____ÿAlgorithm input_____ÿ|algorithm |____ÿAlert ID Number
                |____________|                        |__________|
```

Figure 10-10. Two-Stage Procedure for Generating an Alert ID Number

Figure 10-10 shows the two-stage procedure by which the Alert ID Number is generated for a particular Alert. Briefly, the procedure involves first extracting from the Alert major vector, in a specified order, the contents of the Alert that are significant for defining the Alert ID Number, and then passing these contents through a specified 32 bit cyclic redundancy check (CRC) algorithm to generate a 4 byte Alert ID Number. This number then flows in the Alert major vector from the Alert sender to the Alert receiver.

The input to the CRC algorithm consists of the following code points from the Alert major vector, beginning at the left of the input:

1. The Alert Type

2. The Alert Description Code

3. All Probable Causes code points, in order

4. The delimiter X'FFFF'

This delimiter serves in three places to separate groups of causes code points from each other. Without it there would be no way to tell whether a particular code point was, e.g., the last Probable Cause or the first User Cause.

5. All User Causes code points, in order, if any are present

6. The delimiter X'FFFF'

7. All Install Causes code points, in order, if any are present

8. The delimiter X'FFFF'

9. All Failure Causes code points, in order, if any are present

Note that the Recommended Action code points in an Alert do not figure in the computation of its Alert ID Number. These code points are not included because they do not necessarily relate to the Alert condition per se; they may, for example, vary depending on whether the Alert sending node is attended or unattended, whether it is locally or remotely attached to the Alert receiver, or whether the product implementing it is serviced by the customer or by the vendor.

```
<<<x sup 32> I(x)> + <<x sup k> L(x)>> over <G(x)>
          = Q(x) + <R(x)> over <G(x)>
```

where:

function L(x)

sum from i=0 to 31 of x sup i

function G(x)

sum from i of x sup i
for i = 32,26,23,22,16,12,11,10,8,7,5,4,2,1,0

function I(x)
The polynomial represented by the input to the CRC algorithm

k
number of bits in the input polynomial
function I(x)

The Alert ID number is the complement of the remainder polynomial
function R(x)
(sometimes represented as
Alert%%ID = <function R(x)> bar
). The reader should remember that all arithmetic is modulo 2, and that the degree of the remainder polynomial,
function R(x)
, is less than 32.

The CRC formula in topic 2.10.1.19.3 shows the equations defining the algorithm that actually generates the 32 bit Alert ID Number. The input to the algorithm represents the coefficients of the terms of the polynomial
function I(x)
, in order of *decreasing* powers of
x
. For example, the first bit of the Alert Type code represents the coefficient of the
x sup k
term of
function I(x)
, the second bit of the Alert Type code represents the coefficient of the
x sup k-1
term, and so on. The CRC formula in topic 2.10.1.19.3 summarizes the operations performed on the input to the algorithm:

Operation	*Usual Implementation*
Multiply function I(x) by x sup 32 :	Shift the input 32 places to the left.

Add to this <x sup k><function L(x)> :	Complement the first 32 bits of the input.
Divide this by function G(x) :	Perform polynomial division.
Complement the remainder polynomial:	(self-explanatory)

The result of these operations is the Alert ID Number.

The 32 bit CRC algorithm documented here is identical to that specified in the ISO International Standards 8802/3 (CSMA/CD) and 8802/5 (Token Ring Access Method) for implementing CRCs on local area networks.

An Alert sending product has two alternatives for determining the Alert ID Numbers to be included the Alerts that it sends:

- The product may implement both of the stages shown in Figure 10-10 in its code; such senders will generate their Alert ID Numbers dynamically, as each Alert is being prepared for transmission.

- The product's developers may manually catalogue the relevant code points for each of the Alerts that the product will send (i.e., they may perform the first stage from Figure 10-10 manually). These code points then serve as input to an offline implementation of the CRC formula in topic 2.10.1.19.3; the resulting Alert ID Numbers are then stored in a table, to be retrieved and inserted in each Alert as it is sent.

When it receives an Alert, an Alert receiver creates the Unique Alert Identifier for it by combining the Alert ID Number with the identity of the Alert sender. Thus an Alert might be identified as "9988 Alert 15F3 2684," or "8899 Alert 3392 483B," where 9988 and 8899 denote fictitious IBM machine types. The single product to be identified as *the* Alert sender for the purposes of creating this unique identifier is always that identified in the first Product Identifier (X'11') subvector within the first (i.e., Alert sender's) Product Set ID.

An Alert receiver can make the Unique Alert Identifier available to a user application program before it creates its own display for the Alert. If the user has written an alternative presentation for the Alert, the user application program can pass this display to the Alert receiver together with an indication that the normal display is not to be created for this Alert. Otherwise the user application program indicates to the Alert receiver that the usual display is to be created.

2.10.1.19.4 Reserved Ranges of Alert Code Points

Within each set of Alert code points, a range has been explicitly reserved for non-IBM use. These are all code points of the form X'Exxx' in the following sets

- Alert Description
- Probable Cause
- User Cause
- Install Cause
- Failure Cause
- Recommended Action
- Detailed Data Extended (X'85') Data IDs

as well as all Detailed Data (X'82') Data ID code points of the form X'Ex'. These code points will never be sent by any IBM Alert sender, and they will never be defined in the management services Alert architecture. They are reserved for the use of OEMs, VARs, customer-written application programs, and the like.

Neither Systems Network Architecture nor IBM makes any attempt to regulate the use of code points within this range by different non-IBM implementations.

2.10.1.19.5 The Detailed Data Extended (X'85') and Detailed Data (X'82') Common Subfields

The X'85' and X'82' subfields both provide a similar function, i.e. each instance of a X'85' or X'82' subfield corresponds to a single unit of display (i.e., a single body of information that is displayed as a unit) at the Alert receiver. Since it provides additional function (namely additional Data ID encodings), the X'85' subfield should be used instead

of the X'82' subfield when possible. Note that the X'85' and X'82' subfields are mutually exclusive in a subvector.

To specify a single unit of display four separate pieces of data are required:

° *Product ID Code:* This field instructs the Alert receiver as to what product data, if any, is to be included in the unit of display. It contains the same encoding structure as the Product Set ID Index (X'83') subfield. See Table 10-4 in topic 2.10.1.19.6 for a description of this structure.

° *Data ID:* This field contains a code indicating the type of data present in the subfield. These codes index text, identifying the type of the data, to be included in the display. The Data IDs for the X'85' and X'82' subfields are two bytes and one byte in length respectively. The text strings for the first 256 Data IDs are equivalent for both subfields; i.e., a X'82' Data ID of the form X'xx' and a X'85' Data ID of the form X'00xx' both represent the same text string for each X'00' ° xx ° X'FF'.

° *Data Encoding:* This field contains a code indicating how the data in the subfield is actually encoded, and thus how it is to be displayed. See "Text in Alerts" in topic 2.10.1.19.2 for further details.

° *Detailed Data:* This field contains the detailed data itself, encoded as the value in the Data Encoding field specifies.

Detailed Data Extended (X'85') Subfield:

```
 _______ _______ _______ _______ _________ _______ __________
| X'09' | X'85' | X'91' | X'00' | X'0001' | X'00' | X'1023'  |
|_______|_______|_______|_______|_________|_______|__________|
  .       .       .       .       .         .       .
  .       .       .       .       .         .       .
  .       .       .       .       .         .       .
  .       .       .       .       .         .       .....  Detailed Data
  .       .       .       .       .         .............  Data Encoding
  .       .       .       .       .......................  Data ID
  .       .       .       ............................... Reserved
  .       .       ....................................... Product ID Code
  .       ............................................... Key
  ..................................................... Length
```

Unit of Display:

```
 _____________________________________
|     ACF/FICT ABEND CODE 1023        |
|_____________________________________|
```

Figure 10-11. Example of the Detailed Data Extended (X'85') Subfield and the Corresponding Unit of Display

Figure 10-11 shows an example of the Detailed Data Extended subfield and a possible unit of display corresponding to it. The unit of display is constructed as follows by the Alert receiver:

° "ACF/FICT": The Product ID Code, X'91', instructs the Alert receiver to display the Software Product Common Name for the first software product identified in the Alert sender's Product Set ID. The example assumes that this Alert was sent by FICT, the fictitious software product ACF/FICT.

° "ABEND CODE": The Data ID, X'0001', indexes this text at the Alert receiver.

° "1023": The Data Encoding, X'00', instructs the Alert receiver to interpret the detailed data as a hexadecimal number. Thus the detailed data X'1023' is displayed as "1023."

A detailed data (X'85' or X'82') subfield can occur in the following Network Alert subvectors: the Detailed Data (X'98') subvector itself, the Supporting Data Correlation (X'48') subvector, and the four "Causes" subvectors (X'94'-X'97'). When this subfield occurs in one of the "Causes" subvectors, the unit of display that is constructed from the data it contains appears within the display created from the subvector itself, at a place indicated by the expression "(detailed data qualifier)." This expression may occur at any place within the text associated with a "Causes" or Recommended Action code point: at the beginning, in the middle, or at the end. Up to three instances of this expression may occur within the text associated with any one code point. The following

examples illustrate the placement of these points:

```
X'F0A3'  FAILURE OCCURRED ON (detailed data qualifier)
X'20A0'  NO RESPONSE FROM THE X.21 NETWORK --
                 (detailed data qualifier) EXPIRED
X'12C0'  RETRY AFTER (detailed data qualifier) (detailed data qualifier)
X'32D1'  LOCAL DCE COMMUNICATIONS INTERFACE (detailed data qualifier)
                 (detailed data qualifier) (detailed data qualifier)
```

See "Associating Code Points with X'82', X'83', X'84', and X'85' Subfields" in topic 2.10.1.19.8 for a discussion of how a X'82' or X'85' subfield is associated with the correct causes or Recommended Action code point in a X'94'&ndashX'97' subvector.

When a X'82' or X'85' subfield occurs in the Detailed Data subvector, the unit of display is either incorporated into a qualified message display or presented independently. See "The Qualified Message Data (X'01') Subfield" in topic 2.10.1.19.9 for a discussion of the architecture for qualified messages.

The following example illustrates the use of the Detailed Data Extended subfield within a "Causes" subvector. Figure 10-12 shows a User Causes (X'94') subvector that might be sent to report an unplugged cable. (In the example the machine type of the Alert sender is 9988.) Figure 10-13 shows a possible unit of display resulting from this subvector.

```
Byte:  0     1     2     3     4_5     6     7     8     9     10-11   12    13    14     15    16_17
       _____ _____ _____ _____ _______ _____ _____ _____ _____ _______ _____ _____ _____ _____ _______
      |X'12'|X'94'|X'04'|X'01'|X'34A2'|X'08'|X'85'|X'21'|X'00'|X'0060'|X'11'|X'F3'|X'04' |X'81'|X'1500'|
      |_____|_____|_____|_____|_______|_____|_____|_____|_____|_______|_____|_____|_____|_____|_______|
                  |____X'01' SF_____ÿ|_____________X'85' SF_____________ÿ|_____X'81' SF_____ÿ|
      |_____________________________________X'94' SV_____________________________________________ÿ|
```

```
Interpretation:

Byte  0      (X'12'):  Length of X'94' SV
Byte  1      (X'94'):  SV key
Byte  2      (X'04'):  Length of X'01' SF
Byte  3      (X'01'):  SF key
Byte  4_5  (X'34A2'):  User cause code point X'34A2':
                            CABLE UNPLUGGED:  (detailed data qualifier)
Byte  6      (X'08'):  Length of X'85' SF
Byte  7      (X'85'):  SF key
Byte  8      (X'21'):  PSID index:  Machine type from HW X'11', Alert sender PSID, SV #1
Byte  9      (X'00'):  Reserved
Byte 10-11 (X'0060'):  Code Point:  PORT NUMBER
Byte 12      (X'11'):  Code Point:  Character Set 00640-00500 plus '$', '#', and '@'.
Byte 13      (X'F3'):  EBCDIC '3'
Byte 14      (X'04'):  Length of X'81' SF
Byte 15      (X'81'):  SF key
Byte 16_17 (X'1500'):  Recommended action code point X'1500':
                              CORRECT INSTALLATION PROBLEM
```

Figure 10-12. Example Encoding of the User Causes (X'94') Subvector

```
 _____________________________________________________________________________
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
|                                  .                                          |
|                                  .                                          |
|                                  .                                          |
|                                                                             |
|      USER CAUSED È CABLE UNPLUGGED: 9988 PORT NUMBER 3                      |
|          ACTIONS È CORRECT INSTALLATION PROBLEM                             |
|                                  .                                          |
|                                  .                                          |
|                                  .                                          |
|                                                                             |
|                                                                             |
```

```
|                                                                      |
|                                                                      |
|                                                                      |
|                                                                      |
|                                                                      |
|                                                                      |
|                                                                      |
|______________________________________________________________________|
```

Figure 10-13. Possible Unit of Display for the Specified X'94' Subvector

The individual elements of the display in Figure 10-13 were derived as follows:

- "USER CAUSED-": From the presence of the X'94' subvector
- "CABLE UNPLUGGED:": From user cause code point X'34A2'
- "9988": Hardware machine type of the Alert sender; retrieved from the first Product Set ID (X'10') subvector, because of product ID code point (X'21')
- "PORT NUMBER": From data ID code point X'0060'
- "3": EBCDIC data, that flowed in the X'85' subfield
- "ACTIONS-": From the presence of the X'81' subfield
- "CORRECT INSTALLATION PROBLEM": From recommended action code point X'1500'.

The formatting of the unit of display shown in Figure 10-13 is just for the purposes of illustration. Architecturally, the only requirement is that all of the information presented there be displayed as a unit by an Alert receiver.

2.10.1.19.6 Product Set ID Indexing

Product identification data flowing in Alerts is displayed in three ways by an Alert receiver:

- Separately, as a part of a Product Set ID display for either the Alert sender or the indicated resource.
- In conjunction with strings of detailed data.
- In conjunction with the text indexed by certain "Causes" or Recommended Action code points.

For the second and third of these uses, the Alert sender must be able to specify a particular Product ID (X'11') subvector, and particular data within this X'11' subvector, to be displayed in conjunction with a given text string. This is accomplished by an encoding structure that appears in the following places: in the Product Set ID Index (X'83') subfield, in the Product ID Code field of the X'82' and X'85' subfields, and in the Product ID Code field of the Qualified Message Data subfield (if EP_ALERT optional subset 5 (Qualified Message Data) is implemented). This structure is shown in Table 10-4.

Table 10-4. One-byte Structure for Indexing a Product Identifier (X'11') Subvector

Bits	Meaning
Bits 0-3	Product ID subvector code: X'0': no product identification data is to be displayed X'2': hardware X'11', machine type (hardware product common name if present) X'3': hardware X'11', serial number or repair ID number X'4': hardware X'11', machine type (hardware product common name if present) plus serial number or repair ID number X'5': hardware X'11', machine type (hardware product common name if present) plus model number

	X'6': hardware X'11', machine type (hardware product common name if present) plus model number plus serial number or repair ID number
	X'9': software X'11', software product common name
Bit 4	Product Set ID indicator:
	0: Alert sender Product Set ID
	1: Indicated resource Product Set ID
Bits 5-7	Count: A three-digit binary number indicating which Product ID subvector is being indexed

The first field here contains a single hexadecimal digit that functions as a code point. It specifies the *type* of X'11' subvector, i.e., hardware or software, to be indexed, and also the data within the subvector to be indexed. The second field is a single bit, indicating whether the X'11' subvector to be indexed is in the Alert sender (first) or the indicated resource (second, or only) Product Set ID. The remaining three bits comprise a binary count, indicating which X'11' subvector, of the type indicated in the first field and in the PSID indicated in the second field, is to be used. Only X'11' subvectors of the type specified in the first field are counted. Figure 10-14 gives examples showing how this counting is done.

```
 ____________________________________________________________________________///
|  Alert major vector
|                  ____________________________________________________________
|                 |  Alert sender PSID                                         |
|                 |                 _________   _________   _________   _________|
| LL  X'0000' .... |  L  X'10' |X'11'(HW)|  |X'11'(HW)|  |X'11'(SW)|  |X'11'(SW)|
|                 |           |   (a)   |  |   (b)   |  |   (c)   |  |   (d)   |
|                 |           |_________|  |_________|  |_________|  |_________|
|                 |____________________________________________________________|
|____________________________________________________________________________///

     ///_____________________________________________
                                                    |
                                                    |
        ____________________________________
       |  Indicated resource PSID           |        |
       |                 _________  _________ |      |
       |  L  X'10' |X'11'(HW)|  |X'11'(SW)| | ..... |
       |           |   (e)   |  |   (f)   | |       |
       |           |_________|  |_________| |       |
       |____________________________________|        |
     ///_____________________________________________|
```

Product ID Subvector Code	PSID	Count	X'11' SV Indexed
X'2', X'3', or X'4' (HW)	B'0'	B'001'	(a)
X'2', X'3', or X'4' (HW)	B'0'	B'010'	(b)
X'9' (SW)	B'0'	B'001'	(c)
X'9' (SW)	B'0'	B'010'	(d)
X'2', X'3', or X'4' (HW)	B'1'	B'001'	(e)
X'9' (SW)	B'1'	B'001'	(f)

Figure 10-14. Examples Illustrating How X'11' Subvectors are Indexed

If the product identification text is to be displayed in conjunction with a "Causes" or Recommended Action code point, it is introduced into the stored text indexed by that code point at a location indicated in the architecture by the expression "(sf83 product text)." The same third digit mechanism that indicates how many Detailed Data (X'82') or Detailed Data Extended (X'85') subfields are associated with a code point also indicates whether a X'83' subfield is associated with it; see "Associating Code Points with X'82', X'83', X'84', and X'85' Subfields" in topic 2.10.1.19.8 for details.

2.10.1.19.7 Resource List Indexing

Resource list indexing is similar in concept and design to the Product Set ID Indexing described in "Product Set ID Indexing" in topic 2.10.1.19.6. The Resource List Index (X'84') subfield identifies specific resource identification data to be displayed in conjunction with a "Causes" or Recommended Action code point. Each X'84' subfield contains four pieces

of information:

- The key of the subvector being indexed into.

- The key of the subfield, within this subvector, being indexed into.

- The count of the entry, within this subfield, being pointed to.

- A code point indicating which data for this entry is to be displayed.

This scheme is based on the encoding of the Hierarchy/Resource List (X'05') subvector: multiple subfields, each of which contains multiple resource entries. It is not, however, permanently tied to the X'05' subvector, since the subvector key is explicitly specified in the X'84' subfield.

When an Alert receiver counts resource entries within a specified subfield, it does not count every entry. Rather, it counts only those entries that have their *Resource List Indexing flag* set. This flag is necessary because an Alert sender building a X'84' subfield may not know what its Hierarchy/Resource List (HRL) subvector will look like by the time it reaches the Alert receiver that processes the X'84' subfield. Various types of Alert pass-through and Alert forwarding, both inside and outside the architecture, involve extending the hierarchy in the HRL subvector, by adding entries at the front of one of its subfields. Thus what leaves the Alert sender as the second entry in a subfield of the HRL may be the fourth entry by the time it reaches the Alert receiver.

To ensure that the Alert sender and Alert receiver agree when they count entries for resource list indexing, the sender explicitly sets a flag for each entry to be considered in the counting. Entries added when a hierarchy is extended never have this flag set, so they are never included in the counting by the Alert receiver. Thus what the sender identifies as the second entry in a subfield (i.e., the second entry in the subfield with its Resource List Indexing flag set) will still be the second entry when the Alert reaches the Alert receiver.

2.10.1.19.8 Associating Code Points with X'82', X'83', X'84', and X'85' Subfields

This section describes the mechanism by which the subfields in the X'94'-X'97' subvectors are associated with the appropriate Causes and Recommended Action code points. Figure 10-15 shows a high-level view of one of the Causes (X'94'-X'96') subvectors. The subvector in this figure contains the following:

- A Causes (X'01') subfield containing three cause code points

- Detailed Data Extended (X'85') subfields (a)-(d), providing substitution text for the text strings indexed by the three cause code points

- A Recommended Actions (X'81') subfield containing four Recommended Action code points

- Detailed Data Extended (X'85') subfields (e) and (f), providing substitution text for the text strings indexed by the four Recommended Action code points

The Cause Undetermined (X'97') subvector differs from the X'94'-X'96' subvectors in that it carries no X'01' subfield. It does, however, carry the X'81' subfield and any X'85' (or X'82') subfields associated with it. Thus there is still a task of associating X'85' (or X'82') subfields with code points in the X'97' subvector, but only for one set of code points rather than for two.

Since it provides additional function, the Detailed Data Extended (X'85') subfield should be used instead of the Detailed Data (X'82') subfield when possible. The X'85' and X'82' subfields are mutually exclusive in a subvector.

```
  "Causes" (X'94'-X'96') Subvector:
 ____________________________________________________________________________//
| X'9x' SV
|  ___________ _________________  __________  __________  __________  __________
|  | X'01' SF | CP_1 CP_2 CP_3 | | X'85' SF | | X'85' SF | | X'85' SF | | X'85' SF |
|  |__________|________________| |__________| |__________| |__________| |__________|
|                                    (a)         (b)         (c)         (d)
|____________________________________________________________________________//

      //_____________________________________________________________
```

```
                                                                  |
          ___________ ______________________ ___________ ___________ |
         | X'81' SF | CP_1 CP_2 CP_3 CP_4 | | X'85' SF | | X'85' SF | |
         |__________|____________________| |__________| |__________| |
                                              (e)         (f)     |
     //_____________________________________________________________|
```

Figure 10-15. Structure of the Subfields within a 'Causes' (X'94'-X'96') Subvector. Note that the X'85' subfields could either be inserted as shown, or all grouped together after the X'81' subfield.

The association of Detailed Data (X'82'), Product Set ID Index (X'83'), Resource List Index (X'84'), and Detailed Data Extended (X'85') subfields with code points is accomplished by means of the code points themselves. The third hexadecimal digit of each code point specifies the number of detailed data subfields associated with that code point, and whether there is a X'83' or X'84' subfield associated with it, according to the following scheme:

X'xx0x'-X'xx9x'	No associated subfields.
X'xxAx'-X'xxBx'	One X'82' or X'85' subfield.
X'xxCx'	Two X'82' or two X'85' subfields.
X'xxDx'	Three X'82' or three X'85' subfields.
X'xxEx'	One X'83' subfield.
X'xxFx'	One X'84' subfield.

No option exists for associating multiple X'83' or X'84' subfields with a code point, or for associating both detailed data and X'83' or X'84' subfields with a code point.

Given this encoding scheme, the Alert receiver can always make the correct association of code points with X'82', X'83', X'84', or X'85' subfields, even if the code points are new replacement code points that it does not recognize. In the subvector shown in Figure 10-15, for example, if CP-1 in the X'01' subfield is X'2162', the Alert receiver knows that none of the X'85' subfields (a) through (d) is associated with it. If CP-2 in the X'01' subfield is X'25D1', the Alert receiver knows that X'85' subfields (a), (b), and (c) are associated with it. Similar processing is done for all of the code points in the X'01' and X'81' subfields, to correlate all of the X'85' subfields with the appropriate code points.

Two features of this scheme require further comment:

° If the Alert receiver receives a replacement code point that it does not have in its table, then it does not have the replacement text that this code point indexes. In this case, the architecture specifies that it display the text for the associated default code point. If, however, the unrecognized replacement code point had one or more X'85' or X'82' subfields associated with it, the text resulting from these subfields will not fit naturally into the text indexed by the default code point. (Since default code points are always of the form X'xx00', and so always have a 0 as their third digit, their text is always designed to have no qualifier text inserted into it.) In this case the Alert receiver should display the default text and the qualifier text it builds from the associated X'85' or X'82' subfields. It may format these text elements in any way that it chooses, so as to communicate to the operator that the display contains both default text and qualifier text.

° Some causes and Recommended Actions code points have qualifier text associated with them, and some do not. The third digit of each code point indicates the number of X'85' or X'82' subfields associated with a code point. Given that scheme, two methods could be used by Alert senders and receivers in order to accomodate both cases:

- Have multiple entries in Alert receivers' tables that index the same text, but have different numbers of X'85' or X'82' subfields associated with them (for example, have code points X'5501' and X'55A1' indexing the same text, but requiring, respectively, 0 and 1 X'85' or X'82' subfields).

- Have a single entry for each text string, with the code point for it corresponding to the *maximum* number of strings of qualifier text associated with it. When a sender has no qualifier text strings to pass, or less than the full number of these strings, it sends "empty" X'85' or X'82' subfields, i.e., subfields consisting only of a Length byte (= X'02') and a Key byte (= X'85' or X'82').

Since the first of these solutions would result in needless duplication in the Alert receivers' tables, the second has been chosen. There are thus two formats for the X'85' and X'82' subfields. Figure 10-16 shows the two formats for the X'85' subfield. (3)

```
"Empty" X'85' Subfield:

  Length    Key
 _______ _______
| X'02' | X'85' |
|_______|_______|
    .           "
    .           .
    .............

"Full" X'85' Subfield:

  Length    Key    Prod. ID  Reserved   Data ID   Data Enc.         Data
 _______ _______ _________ _________ __________ _________ ___________________
| X'xx' | X'85' |  X'xx'  |  X'00'  |  X'xxxx'  |  X'xx'  |  X'xx'.......X'xx' |
|_______|_______|_________|_________|__________|_________|___________________|
    .                                                                        "
    .                                                                        .
    ..........................................................................
```

Figure 10-16. Structure of 'Empty' and 'Full' X'85' Subfields

---

(3) The two formats for the X'82' subfield are similar, i.e. the key is X'82', there is no Reserved field, and the Data ID is one byte instead of two.

2.10.1.19.9 The Qualified Message Data (X'01') Subfield

The Qualified Message Data (X'01') subfield, within the Detailed Data (X'98') subvector, transports a message code that indexes a message stored at the Alert receiver. National language support is the primary reason for transporting the message code rather than the message itself.

In some cases, it is necessary to transport not only a message code, but also *qualifiers*, substitution text that is inserted into the message text indexed by the code. These qualifiers are themselves transported in Detailed Data Extended (X'85') or Detailed Data (X'82') subfields that are pointed to from the Qualified Message Data subfield.

In many ways, the Qualified Message Data subfield is similar to the detailed data subfields, discussed in detail in "The Detailed Data Extended (X'85') and Detailed Data (X'82') Common Subfields" in topic 2.10.1.19.5. It carries a product ID code, a data ID (indicating the type of the message code), an indication of how the message code is itself encoded, and the message code. There are, however, two important differences:

- Whereas the detailed data in a X'85' or X'82' subfield is itself displayed by the Alert receiver, the message code in the Qualified Message Data subfield *indexes* a stored message that is displayed.

- The Qualified Message Data subfield contains a Qualifier Count, indicating how many succeeding X'85' or X'82' subfields are to be associated with the qualified message.

Figure 10-17 illustrates the second of these points. The number shown in each Qualified Message Data (X'01') subfield is the value in its Qualifier Count field. This number indicates how many X'85' subfields are to be processed in association with the X'01' subfield. In the upper example, this number is 2, indicating that the next two X'85' subfields are to be treated as qualifiers of the message indexed by the X'01' subfield. The remaining two X'85' subfields are then treated as independent units of detailed data. In the lower example, the Qualifier Count field indicates that only one X'85' subfield belongs with the message text. It is the responsibility of the Alert sender to ensure that the proper number of detailed data subfields are associated with each Qualified Message Data subfield it sends.

```
Detailed Data Subvector:
```

```
| X'98' SV                                                                |
|  ___________________ ___________ ___________ ___________ ___________   |
|  | X'01' SF (2) (M1) | | X'85' SF | | X'85' SF | | X'85' SF | | X'85' SF |  |
|  |___________________| |_________| |_________| |_________| |_________|  |
|                          (a)         (b)         (c)         (d)        |
|_________________________________________________________________________|
```

Unit of Display:

```
 _________________________________
| (Message indexed by M1 (a) (b) ) |
| (c)                              |
| (d)                              |
|__________________________________|
```

Detailed Data Subvector:

```
 _________________________________________________________________________
| X'98' SV                                                                |
|  ___________ ___________ ___________________ ___________ ___________   |
|  | X'85' SF | | X'85' SF | | X'01' SF (1) (M2) | | X'85' SF | | X'85' SF |  |
|  |__________| |__________| |___________________| |_________| |_________|  |
|     (e)         (f)                                (g)         (h)       |
|_________________________________________________________________________|
```

Unit of Display:

```
 _________________________________
| (e)                              |
| (f)                              |
| (Message indexed by M2 (g) )     |
| (h)                              |
|__________________________________|
```

Figure 10-17. Relationship Between Qualified Message Data and Detailed Data Subfields

The text indexed by the message codes transported in the Qualified Message Data subfield is not architected in the way that the text indexed by the default/replacement Alert code points is. The architecture provides a mechanism that allows an Alert sender to specify to an Alert receiver which message is to be displayed; the messages themselves, however, are not defined by the architecture.

Support of the Qualified Message Data subfield is optional for an Alert receiver. When a particular Alert receiver does not support this subfield, the additional information contained in the message specified by the Alert sender is not available at that Alert receiver.

2.10.1.19.10 Hierarchy Information in Alerts

Hierarchy information in Alerts is reported in the Hierarchy/Resource List (X'05') MS common subvector. Even though this is a common subvector, appearing in a number of management services major vectors, the majority of its features are used only in Alerts. Thus the discussion of this subvector and the hierarchy information that it conveys appears here, in the Alerts section.

The Hierarchy Name List (X'10') Subfield:

The Hierarchy Name List (X'10') subfield provides logical identification of the resource being reported on by an Alert. It lists a hierarchy of resources, physically connected to each other, terminating in the Alert's indicated resource. For each resource, two pieces of information are provided: a name of up to eight characters, and a code point indicating the resource type.

An Alert receiver typically uses the hierarchy information supplied to it in two ways: it is required to present the complete hierarchy passed to it, but it may also, optionally, identify the indicated resource's name and type individually on a dynamic display of incoming Alerts. If an Alert receiver does elect to display the indicated resource individually, there is an additional piece of data from the X'10' subfield that it uses. For each resource identified in the X'10' subfield, the Alert sender specifies whether to display that resource as the Alert's indicated resource. The *display resource name indicator* is used as follows in constructing the individual display for an indicated resource:

- The resource *type* displayed is always the one associated with the last entry in the X'10' subfield.

- To choose a resource *name*, the Alert receiver starts at the end of the X'10' subfield and scans backwards. It skips over any entries for which the display resource name indicator is set to "do not display," scanning until it finds an entry for which the indicator says "display."

- If the resource name that it displays is not that from the last entry in the X'10' subfield, then the Alert receiver indicates on the screen in some way (e.g., by a special character) that the resource type and resource name it is displaying belong to different resources.

The goal of the display resource name indicator is to insure that the identification of the indicated resource is always meaningful to the operator at the Alert receiver. If the Alert receiver always displayed the name and type from the last entry in the X'10' subfield, this would not be the case.

An Alert from an SNA node reporting on a locally-attached non-SNA printer might, for example, have for the last entry in the X'10' subfield: name="PRINTER1," type=X'13' (printer). The name "PRINTER1" here is a local name, having meaning only at its own node; its role is to distinguish one of two local printers from the other one, PRINTER2. Every node in the network of the same type as the Alert sender might very well have its own PRINTER1 and PRINTER2.

In this situation, the Alert sender would set the display resource name indicator for the printer to "do not display"; the indicator for the sender itself would be set to "display." The operator at the Alert receiver would then see not "name=PRINTER1 type=PRINTER," which could have come from any of several nodes in the network, but instead "name=(SENDER) type=PRINTER *," where the asterisk indicates that (SENDER) and PRINTER refer to different resources. The operator thus knows which node has sent the Alert, and also that this node has experienced a problem with one of its printers. The full display of the hierarchy will indicate that the Alert is reporting on PRINTER1 rather than PRINTER2.

The hierarchy reported in the X'10' subfield always ends with the Alert's indicated resource, also termed the origin of the Alert condition. This is ordinarily the resource that has failed. The nature of the hierarchy preceding the indicated resource varies depending on the nature of the Alert sender:

- For a type 2 or type 4 node, sending the Alert major vector in an NMVT on an SSCP-PU session, the hierarchy reported in the X'05' subvector is limited to non-SNA resources below the sender. In this case the remainder of the hierarchy, including the name and type of the sender itself, is provided by the sender's SSCP. The sender sets the *hierarchy complete indicator* in the X'10' subfield to "not complete" in this case, to inform the Alert receiver that the complete hierarchy for this Alert involves not only the entries in the X'10' subfield, but also those supplied by the SSCP.

- For a type 2.1 node sending the Alert major vector in a CP-MSU on a CP-CP session, the hierarchy reported in the X'05' subvector extends up through the sender's own control point, and is complete. Thus the sender sets the hierarchy complete indicator to "complete."

A type 2.1 node that sends its Alert major vectors in NMVTs on an SSCP-PU session may use either of the methods described here: i.e., it may include in the X'10' subfield either a complete hierarchy, or only the non-SNA resources below itself. It sets the hierarchy complete indicator accordingly, to inform the Alert receiver which it has done.

For Alert receivers that have limited space to provide the Hierarchy Name List (X'10') subfield information in a graphic or tabular manner, use the following rules of preservation:

- Save the first name at the top of the list. This is the resource closest to the Alert receiver.

- Save the last name at the bottom of the list. This is the resource for which the Alert has been sent, typically the failing resource.

- Save the remaining names up from the bottom of the list, in ascending order.

The Associated Resources (X'11') Subfield:

The Associated Resources (X'11') subfield identifies one or more resources associated with an Alert condition, but not included in the hierarchy reported in the X'10' subfield. An example of such a resource is the partner LU for a session-related Alert. For such an Alert, one LU, the one more closely associated with the Alert sender's node, is identified as the last entry in the X'10' subfield, while its partner is identified in the X'11' subfield.

While they are similar in overall structure, there are several important differences between the X'10' and the X'11' subfields:

- While the entries in the X'10' subfield are hierarchically related to each other, those in the X'11' subfield are not. If the X'11' subfield identifies more that one resource, these resources are peers: the order in which they appear in the subfield carries no significance.

- While resource names are limited to eight characters in the X'10' subfield, they may be up to 56 characters in length in the X'11' subfield. They must still, however, be *names* only, not concatenations of fields that convey other information about the resources.

- Since associated resources are not included in the dynamic Alerts display, the display resource name indicator is not defined for the X'11' subfield.

Associated resources are not elements of the hierarchies identified in the X'10' subfield; thus an Alert receiver does not display them as a part of these hierarchies. One way for an Alert receiver to handle display of associated resources is to include with the hierarchy display an indication that there are associated resources present for this Alert, then display the associated resources themselves at some regular location. To help the operator find the associated resources for an Alert, a Recommended Action code point X'2012' is defined; the exact text for this code point is specific to each Alert receiver, but the meaning conveyed is "(Review associated resources)."

The Associated Resources subfield makes it possible for an Alert receiver to correlate multiple Alerts that pertain to the same resource, even if these Alerts come from different senders. For example, it is possible to correlate Alerts from different senders when a specific LU is the *partner* of the LUs hierarchically related to each Alert sender.

Flags for Resource List Indexing:

Each entry in the Hierarchy Name List (X'10') and Associated Resources (X'11') subfields of the Hierarchy/Resource List subvector has associated with it a resource list indexing flag. See "Resource List Indexing" in topic 2.10.1.19.7 for a description of how this flag is used.

The Processing Node List (X'20') Subfield:

The Processing Node List (X'20') subfield is used in the nested focal point environment, to allow each focal point processing an MSU to identify itself. With this information, a higher-level focal point can retrace the path that an MSU has taken to reach it, and thus be in a position to, e.g., direct commands to the focal point that first received a particular Alert.

2.10.1.19.11 Correlation in Alerts

Correlation of Multiple Alerts:

The Alert architecture provides a mechanism for correlating multiple Alerts reporting the same failure. Ordinarily these Alerts are sent by different nodes, and they represent different views of the same failure. If, for example, an SNA link is lost, the Alert sent by the node at one end of the link may well contain information not available to the node at the other end of the link. The Alert receiver, however, needs an indication that the two Alerts are reporting the same failure; otherwise the network operator is presented with two partial views of the problem (with no indication that they are views of the same problem), rather than with one unified view.

The general requirement for correlating multiple Alerts is met through the use of the MSU Correlation (X'47') common subvector. This subvector carries one or more subfields that function as correlation *tokens*. The Alert receiver compares each token in an incoming Alert with those in Alerts that were received earlier; a match indicates that two Alerts are reporting the same problem.

In order for this mechanism to work, Alert senders that may be sending Alerts on the same problem must agree upon a token to include in their Alerts *before* there is any problem. The procedures for reaching agreement on a token vary depending on how the prospective Alert senders are related:

- If they are session partners, the token is the Fully Qualified PCID associated with the session.

- If they are commonly involved with a logical unit of work, then the token consists of the NETID, LU Name, Instance Number, and a Sequence Number. This token uniquely identifies the shared logical unit of work.

- If they are nodes at the two ends of a link, they exchange Link Instance Identifier (X'67') control vectors as part of their XID exchange. The correlation token is then defined to be the 4-byte value sent in the Link Instance Identifier by the control point with the alphabetically earlier name, conjoined with that control point's name. Notice that so long as each control point insures that its own 4-byte values are different, this insures that the tokens for different link instances in a network are always different.

- If shared hardware resources are each reporting on the same condition, then a combination of a resource name, resource number, and a sequence number (incremented each time a condition occurs) is used as a token for correlating the Alerts. Alert senders are responsible for defining the meanings of the contents of each of the fields in this token.

Currently the following correlation tokens are defined for the X'47' subvector: the Link Instance Identifier (X'01') subfield, the Resource Instance Identifier (X'10') subfield, and Logical Unit of Work Correlation Token (X'20') subfield, and the Fully Qualified Session PCID (X'60') subfield. Alert receivers will not be affected, however, if additional tokens are added later. Since the only role of a token is to be compared with other tokens, an Alert receiver does not have to understand the meaning of a token's contents. So long as the receiver treats the entire subfield, including its length and key bytes, as the token on which it does the comparisons, it will be unaffected if additional subfields are added to the architecture.

Correlation of an Alert and Supporting Data:

The Alert architecture provides an Alert sender with a means of correlating an Alert with *supporting data* about the problem, i.e., data about the problem stored locally by the Alert sender. The Supporting Data Correlation (X'48') common subvector transports one or more subfields identifying the supporting data associated with an Alert. If, for example, additional data related to a problem has been stored in a log at the Alert sender, the X'48' subvector can transport Detailed Data Extended (X'85') or Detailed Data (X'82') subfields (4) identifying both the file containing the supporting data and one or more search keys for the relevant records in this file. The network operator is then in a position to correlate the Alert with the Alert sender's log entries for the same problem.

Correlation of Alerts and Resolution Notifications:

If EP_ALERT Optional Subset 13 (Resolution Notification Support) is implemented, the Alert architecture provides an Alert/resolution notification sender with an optional means of correlating an Alert with a resolution notification. If implemented, this correlation is provided via a unique token identifying the problem. This unique token, the incident identifier, is carried in the Incident Identification (X'4A') subvector of both the Alert and Resolution major vectors.

Each Alert sent contains one unique incident identifier corresponding to the reported Alert condition. This same incident identifier is included in the resolution notification that signifies that the Alert condition has been resolved. The network operator is then in a position to determine which problems previously reported by Alerts are now resolved.

It is possible that multiple Alerts could have been sent for seemingly different problems. Yet it may later be discovered that one particular problem caused a group of Alerts to be sent. In this case, an incident identifier for each resolved Alert condition may be included in the X'4A' subvector within the Resolution major vector. If the incident identifiers for all of the resolved conditions do not fit within a single X'4A' subvector, then multiple X'4A' subvectors may be included in the Resolution major vector.

The incident identifier is carried in the Incident Identifier (X'01') Incident Identification Subfield. One of five different encoding schemes may be chosen to generate an incident identifier. Since the only role of an incident identifier is to be compared with other incident identifiers, an Alert/resolution notification receiver does not have to understand the meaning of an incident identifier's contents. So long as the receiver treats the entire subfield, including its length and key bytes, as the token on which it does the comparisons, it will be unaffected if additional encoding schemes are added to the architecture. See *SNA*

Formats for details of the encoding schemes.

---

(4) For more information on these subfields, see "The Detailed Data Extended (X'85') and Detailed Data (X'82') Common Subfields" in topic 2.10.1.19.5.

2.10.1.19.12 Examples of Physical and Logical Identification of the Origin of an Alert Condition

The following examples describe how the origin of an Alert condition (the failing network component) is identified in the Alert. This identification falls into two categories: *physical*, which flows in the Product Set ID (X'10') subvector, and *logical*, which may involve either or both of the SNA Address List (X'04') and Hierarchy/Resource List (X'05') subvectors.

In addition to data identifying the failing component, an Alert must also carry data identifying the PU sending the Alert.

Figure 10-18, Figure 10-19, and Figure 10-20 show the structure of the Product Set ID (PSID) subvectors that identify these resources.

```
 _________________________________ _________________________________
|                                 |                                 |
|_________________________________|_________________________________|
|________X'10' subvector__________ÿ|_______X'10' subvector___________ÿ|
                                            (if required)
```

Figure 10-18. Product Set ID (X'10') Subvectors. One X'10' subvector contains data identifying the reporting PUMS. A second X'10' contains data identifying the problem origin. If the problem origin *is* the reporting PUMS, only the first X'10' is generated.

```
 _______ ____________________________ ___________________________________
|       |                            |  °  °  °                          |
|_______|____________________________|___________________________________|
|X'10'ÿ|_____X'11' subvector_______ÿ|____________additional____________ÿ|
 header                                          X'11' subvectors
  data                                              (as needed)
|_________________________________X'10'_________________________________ÿ|
```

Figure 10-19. Product Identifier (X'11') Subvectors within a X'10' Subvector. A X'11' subvector is generated for each product (hardware or software) that makes up the entire product set to be identified.

```
 _____ ________ _________________ _________________ ______________________
|     |        |                 |                 |  °  °  °             |
|_____|________|_________________|_________________|______________________|
|X'11'| product|_____________________subfields_____________________________ÿ|
 header  classi_                    (as required)
  data   fication
|__________________________X'11' subvector__________________________________ÿ|
```

Figure 10-20. Product Identifier (X'11') Subvector. A X'11' subvector contains data to identify one hardware or software product. The product classification field identifies the product as hardware or software, and IBM or non-IBM. See *Systems Network Architecture Formats*, GA27-3136, for a description of which subfields are required to be present.

The examples in this section show all the subvectors that are required in an NMVT Alert, not just those provided by the EP_ALERT base subset.

The following figures illustrate some possible origins of Alert conditions (denoted X^1, X^2, X^3, X4, X5) in a network. The figures show only the portion of the network affected by the Alert condition and the resulting NMVT Alert flow. Below each configuration is an illustration of the NMVT

Alert that flows on the SSCP-PU session. This NMVT is broken down to show explicitly the subvectors that contain the physical and logical identification of the components being identified.

EXAMPLE 1

Consider the type 2 node in Figure 10-21. If a temporary error is detected in the adapter for the link to the type 4 node (see error "X¹"), the PU (reporting PUMS) will generate an Alert. This Alert does not contain an SNA Address List subvector, since the PU name can be determined from the origin of the RU. The absence of an SNA Address List subvector identifies the PU itself as the origin of the Alert condition.

A single Product Set ID (X'10') subvector will contain data identifying the entity that is both the reporting PUMS and the problem origin.

```
                                                  type 2 node
 ___________          type 4 node                ___________
|           |          _______                  |           |
|           |________|       |_________________|X¹         |
|       CP  |        |_______|                 |  PU/PUMS  |
|___________|                                  |___________|
         NMVT   [Alert (A sv), (PSID sv {PU} )
         & (other sv) ]
          ___________________________________________
```

KEY:

[ ] = contained in NMVT
( ) = subvectors contained in Alert major vector
{ } = component identified by a subvector

X¹: denotes a failure within a device implementing the PU (e.g., temporary error in the adapter for the link to the type 4 node)
NMVT: Network Management Vector Transport
Alert: Alert major vector (X'0000')
A sv: Alert subvectors (X'92'-X'98')
PSID sv: Product Set ID subvector (X'10')
other sv: other required (e.g., Date/Time (X'01') or Relative Time (X'42')) and optional (e.g., Self-Defining Text Message (X'31')) subvectors

Figure 10-21. Illustration of Example 1

EXAMPLE 2

Next consider the display in Figure 10-22. If a failure is detected in the display itself (see error "X²"), the LU LMS provides alert information to PUMS, so that PUMS can send an alert. This Alert will contain the SNA Address List subvector because the SNA component that is the problem origin is the LU.

One Product Set ID subvector identifies the reporting PUMS. A second Product Set ID subvector identifies the problem origin, the LU.

```
                                      type 2 node        LU
 ___________        type 4 node        ___________      Display
|           |        _______          |           |      _______
|           |_______|       |_________|           |____|  X²   |
|       CP  |       |_______|         |  PU/PUMS  |    |_______|
|___________|                         |___________|


             NMVT   [Alert (SNAAL sv {l/a of LU} ),
                    (A sv ), (PSID sv { PU } ),
                    (PSID sv { Display} ), (other sv) ]
          ___________________________________________
```

KEY:

[ ] = contained in NMVT
( ) = subvectors contained in Alert major vector
{ } = component identified by a subvector

X²: denotes a failure within a device implementing the LU or part of the LU function (e.g., hardware card failure in the

```
                display)
NMVT:           Network Management Vector Transport
Alert:          Alert major vector (X'0000')
A sv:           Alert subvectors (X'92'-X'98')
PSID sv:        Product Set ID subvector (X'10')
SNAAL sv:       SNA Address List subvector (X'04')
other sv:       other required (e.g., Date/Time (X'01') or Relative Time (X'42'))
                and optional (e.g., Self-Defining Text Message (X'31'))
                subvectors
l/a:            local address
```

Figure 10-22. Illustration of Example 2

EXAMPLE 3

Consider the printer (printer B) attached to the display in Figure 10-23. If a failure is detected that prohibits the LU from communicating with the printer (see error "X³"), the LU provides alert information so that PUMS can send an Alert. This Alert contains the SNA Address List subvector because the SNA component closest to the failure implements (at least part of) the LU. This Alert also contains the Hierarchy Name List subvector to identify the non-SNA addressable resource that is the origin of the Alert condition (the printer).

The Alert NMVT supports the presence of both the Hierarchy Name List and SNA Address List subvectors. These two subvectors will flow together on the SSCP-PU flow when a non-SNA addressable resource, controlled by a network device that is implementing an LU, fails. The SNA Address List subvector provides part of the SNA hierarchy by identifying the LU while the Hierarchy Name List subvector provides the hierarchy from the LU to the failing non-SNA addressable resource.

A Product Set ID subvector will be generated that will describe the product set that contains the PUMS. The second PSID will identify the problem origin. In this case the machine is printer B.

```
                                                LU
                                 type 2 node   Display
 ___________      type 4 node    ___________  (e.g., PC)
|           |      _______      |           |     _____ X³
|           |_____|       |_____|           |____|     |____
|     CP    |     |_______|     |  PU/PUMS  |    |_____|    |
|___________|                   |___________|                __|__
                                                            |     |
                                                            |_____|
                                                          Printer B

                    NMVT  [Alert (SNAAL sv {l/a of LU} )
                          (A sv), (PSID sv {PU} ),
                          (PSID sv {printer B} ),
                          (HRL sv {Printer  B} )
                          (other sv) ]
          ____________________________________________
```

KEY:

```
[ ] = contained in NMVT
( ) = subvectors contained in Alert major vector
{ } = component identified by a subvector
```

```
X³:             denotes a communication failure with the printer that is
                attached to a device implementing the LU or part of the LU
                function (e.g., power failure)
NMVT:           Network Management Vector Transport
Alert:          Alert major vector (X'0000')
A sv:           Alert subvectors (X'92'-X'98')
PSID sv:        Product Set ID subvector (X'10')
SNAAL sv:       SNA Address List subvector (X'04')
HRL sv:         Hierarchy/Resource List subvector (X'05')
other sv:       other required (e.g., Date/Time (X'01') or Relative Time (X'42'))
                and optional (e.g., Self-Defining Text Message (X'31'))
                subvectors
l/a:            local address
```

Figure 10-23. Illustration of Example 3

EXAMPLE 4

Consider the printer (printer B) attached to the display in Figure 10-24. When a failure is detected by the RAS package within the printer (see error "X4") the LMS for the printer provides alert information so that PUMS can send an Alert. This Alert contains the SNA Address List subvector because the SNA component closest to the failure implements (at

least part of) the LU. This Alert also contains the Hierarchy Name List subvector to identify the non-SNA addressable resource that is the origin of the alertable condition (the printer).

The Alert NMVT supports the presence of both the Hierarchy Name List and SNA Address List subvectors. These two subvectors will flow together, on the SSCP-PU flow, when a non-SNA addressable resource, controlled by a network device that is implementing the LU, fails. The SNA Address List subvector provides the SNA hierarchy by identifying the LU while the Hierarchy Name List subvector provides the hierarchy from the LU to the failing non-SNA addressable resource.

A PSID will be generated that will describe the product set that contains the PUMS. A second PSID will identify the problem origin, the printer attached to the LU. There will be no PSID associated with the LU since both the failing component and the PUMS have been positively identified.

```
                                                LU
                                 type 2 node   Display
  ___________        type 4 node  ___________  (e.g., PC)
 |           |        _______    |           |    ______
 |           |______|       |____|           |____|      |____
 |       CP  |      |_______|    |  PU/PUMS  |    |______|    |
 |___________|                   |___________|                 __|__
                                                              |X4   |
                                                              |_____|
                                                            Printer B


                    NMVT  [Alert (SNAAL sv {l/a of LU} )
                          (A sv), (PSID sv {PU} ),
                          (PSID sv {Printer B} )
                          (HRL sv {Printer  B} )
                          (other sv ) ]
            ____________________________________________
```

KEY:

[ ] = contained in NMVT
( ) = subvectors contained in Alert major vector
{ } = component identified by a subvector

X4: denotes a failure in the printer that is attached to a device implementing the LU or part of the LU function
NMVT: Network Management Vector Transport
Alert: Alert major vector (X'0000')
A sv: Alert subvectors (X'92'-X'98')
PSID sv: Product Set ID subvector (X'10')
SNAAL sv: SNA Address List subvector (X'04')
HRL sv: Hierarchy/Resource List subvector (X'05')
other sv: other required (e.g., Date/Time (X'01') or Relative Time (X'42')) and optional (e.g., Self-Defining Text Message (X'31')) subvectors
l/a: local address

Figure 10-24. Illustration of Example 4

EXAMPLE 5

Consider the diskette device (diskette A) attached to the type 2.0 node in Figure 10-25. If a failure is detected in the diskette drive (see error "X5"), the LMS provides alert information to PUMS so that PUMS can send an Alert. This Alert contains the Hierarchy Name List subvector to identify the non-SNA addressable resource that is the origin of the Alert condition. This will provide the hierarchy from the PU to the failing resource. A maximum of five entries (an entry = a resource name + resource type index) are allowed in a Hierarchy Name List subvector.

A PSID will identify the reporting PUMS. A second PSID will be generated to identify the problem origin, the diskette.

```
                                 type 2 node
  ___________        type 4 node  ___________        _____
 |           |        _______    |           |_______|X5   |
 |           |______|       |____|           |       |_____|
 |       CP  |      |_______|    |  PU/PUMS  |      Diskette A
 |___________|                   |___________|

             NMVT   [Alert (A sv),
                    (HRL sv {Diskette A} ),
                    (PSID sv {PU} ),
                    (PSID sv {Diskette A} ),
                    & (other sv) [
            ____________________________________________
```

```
KEY:


[ ] = contained in NMVT
( ) = subvectors contained in Alert major vector
{ } = component identified by a subvector


X5:            denotes a failure in the diskette or diskette drive
               that is attached to the device implementing the PU/PUMS
NMVT:          Network Management Vector Transport
Alert:         Alert major vector (X'0000')
A sv:          Alert subvectors (X'92'-X'98')
PSID sv:       Product Set ID subvector (X'10')
HRL sv:        Hierarchy/Resource List subvector (X'05')
other sv:      other required (e.g., Date/Time (X'01') or Relative Time (X'42'))
               and optional (e.g., Self-Defining Text Message (X'31'))
               subvectors
```

Figure 10-25. Illustration of Example 5

2.10.2 EP_RTM Function Set

```
°°°
```

- Internal MS protocol boundary A (Send NMVT)

  The EP_RTM function set group requests the SEND_DATA_SSCP_PU function set group to send an NMVT RU on the SSCP-PU session with its controlling CP. The output data consists of the complete NMVT and the address of the control point to which it is to be sent. The details of this protocol boundary are described in "Protocol Boundary A--Send NMVT" in topic 2.9.1.3.

- Internal MS protocol boundary E (Send NMVT Response)

  The EP_RTM function set group requests the PUMS_RECEIVE_NMVT function set group to send an NMVT response RU on an SSCP-PU session with a specified CP. The output data consists of the CP address and sense data. The details of this protocol boundary are described in "Protocol Boundary E--Send NMVT Response" in topic 2.9.1.7.

2.10.2.2 Prerequisite Function Sets

See "Role Requirements for Management Services Components" in topic 2.8.1 for information on the relationships between this function set and other function sets.

2.10.2.3 Overview of Subsets

```
 ___________
|  Local    |
|  Display  |
|           |
|  optional |
|  subset 1 |
|___________|____________________________
|                                        |
|           EP_RTM base subset           |
|                                        |
|________________________________________|
```

Figure 10-27. Base and Optional Subsets of EP_RTM Function Set

---

2.10.2.4 EP_RTM Base Subset

Subtopics:

- 2.10.2.4.1 Functions Provided
- 2.10.2.4.2 Formats Supported
- 2.10.2.4.3 Electives
- 2.10.2.4.4 Implementation Requirements

2.10.2.4.1 Functions Provided

The EP_RTM base subset provides the capability to monitor end-user response times as described by "Response-Time Monitoring (RTM)" in topic 1.4.1.

2.10.2.4.2 Formats Supported

The EP_RTM base subset supports the receiving of an NMVT containing a Request Response-Time Monitor (X'8080') major vector.

The EP_RTM base subset supports the sending of an NMVT containing a Response Time Monitor (X'0080') major vector. The following subvectors are built by EP_RTM:

° A Date/Time (X'01') or Relative Time (X'42') MS common subvector

- An SNA Address List (X'04') MS common subvector
  - When sending unsolicited data and status
  - When positively replying to a request for data and status
- A Data Reset Flag (X'45') MS common subvector
  - When a set of RTM counters has been reset
- A Sense Data (X'7D') MS common subvector
  - When a request could not be successfully completed, and the elective for reporting this condition in a reply has been selected
- An RTM Status Reply (X'91') subvector
  - When sending unsolicited data and status
  - When positively replying to a request for data and status
- An RTM Data (X'93') subvector
  - When sending unsolicited data and status
  - When positively replying to a request for data and status for an LU that has accumulated data

---

2.10.2.4.3 Electives

There are two electives available to this base subset:

1. The reporting of requests that cannot be processed by sending sense data in either a negative response or a reply.
2. Always sending a reply for the last configured LU after a request for RTM data and status for all LUs with accumulated data. This elective allows the PU to send replies for earlier LUs immediately, without having to hold each one until it can be determined whether it is the last reply for the request.

2.10.2.4.4 Implementation Requirements

The requirements for implementing the EP_RTM base subset are described by a model consisting of subsets of PU configuration services, an LU LMS, and PU management services.

A Subset of PU Configuration Services:

- Interacts with PUMS via MS protocol boundary 6 as follows:
  - Provides a list of all LUs configured at the node when requested by PUMS

A Subset of an LU LMS:

- Interacts with PUMS via MS protocol boundary 2 as follows:
  - Sets RTM parameters when requested by PUMS. If the LU LMS was not successful, the reason the request could not be completed is passed to PUMS.
  - Provides RTM status and data when requested by PUMS.

    The LU LMS transfers the requested RTM data and status to PUMS. If the request could not be acted upon, the reason is passed to PUMS.

- If RTM parameters so indicate, provides RTM data and status to PUMS whenever an RTM counter overflows, or whenever a session is deactivated

° Manages the collection of RTM data as follows:

- Activates the RTM function whenever PUMS requests that the RTM parameters be set

  Default parameters are used for any values not specified.

- Deactivates the RTM function as follows:

  -- When the node is IPLed

  -- When an ACTPU (cold) is received

  -- When requested by PUMS

- Resets RTM counters for its specific LU as follows:

  -- When the node is IPLed

  -- When an ACTPU (cold) is received

  -- When an ACTLU is received

  -- Whenever the counters for that LU are sent unsolicited

  -- When RTM is deactivated by a request from PUMS

  -- When a reset is requested by PUMS

  -- When a change to the RTM parameters is requested by PUMS

A Subset of PU Management Services:

° Provides the following processes:

- Receiving RTM requests

  -- Upon request by the RECEIVE_REQUEST_SSCP_PU function set group (via protocol boundary D), does the following:

    ° Parses the requests for validity and requests the RECEIVE_REQUEST_SSCP_PU function set group (via protocol boundary E), to send a positive or negative response.

    ° If the request parsed correctly and it was for a single LU, processes the request by requesting the specified LU to set RTM measurement parameters or gather RTM data.

    ° If the request parsed correctly and it was for all LUs, processes the request by requesting all LUs to set RTM parameters or to gather RTM data.

- Sending RTM data

  Upon request by a subset of an LU LMS, does the following:

  -- Receives RTM data and status, or status only, from the LU

  -- Receives sense data from the LU

  After receiving the data, it builds an NMVT containing one or more of the subvectors described in "Formats Supported," except in the following cases involving replies to a request for RTM data and status from all LUs with accumulated data:

  -- The LU has provided status only, and it is not the last configured LU

-- The LU has provided status only, it is the last configured LU, both data and status have been returned for the current request by at least one earlier LU, and Elective 2 (always reply for the last configured LU) has not been implemented

Note that an LU can provide status without accompanying data only in reply to a request for RTM data and status; when RTM information comes from the LU unsolicited, it always includes both data and status. Note also that a reply is always sent to a request for RTM data and status from a specified LU, even if the LU returns status only, regardless of whether the LU is the PU's last configured LU.

After this the NMVT is passed to the SEND_DATA_SSCP_PU function set group via protocol boundary A.

Receiving RTM Requests:

This process is started by the process described in "Receiving NMVTs" in topic 2.9.6.4.3 (RECEIVE_REQUEST_SSCP_PU function set group), when the latter has identified the major vector key X'8080' (Request Response Time Monitor) in a request NMVT. The first task of this process is to parse the RTM major vector. If there is a syntactic error in the major vector, or in one of the subvectors it contains, or if PUMS in the node fails to support some function presupposed by the request, then this process starts the process described in "Sending NMVT Responses" in topic 2.9.6.4.3 to send the appropriate -RSP. The sense data that can be sent by PUMS at this time are detailed in Table 10-48 in topic 2.10.8.2. If no condition requiring a -RSP is found, then this process starts the process described in "Sending NMVT Responses" (via protocol boundary E) to send a +RSP.

Table 10-5. Subvector Conditions of Presence in the Request RTM (X'8080') Major Vector

Request Type	Subvectors present in the Request RTM (X'8080') major vector
Retrieve data for all LUs with accumulated data	X'92'
Retrieve data for a specified LU	X'92', X'04'
Set parameters for all LUs	X'92', X'94'
Set parameters for a specified LU	X'92', X'94', X'04'

The Receiving RTM Requests process handles each of the four types of requests that it may receive differently. Table 10-5 lists the four request types and the subvectors that are present in each. Summarizing the information in Table 10-5:

° The RTM Request (X'92') subvector is present on all requests.

° The SNA Address List (X'04') subvector is present on requests directed to a single LU, and absent from requests directed to all of a PU's LUs.

° The RTM Control (X'94') subvector is present on requests to set RTM parameters, and absent from requests for RTM data.

After determining the type of request it has received, and verifying that the request is syntactically correct, the Receiving RTM Requests process proceeds as follows:

° *Set RTM parameters for a specified LU:*

Since this request expects no reply, no request/reply correlation is needed. This process sends a request to the specified LU's LMS to set its RTM parameters to the values specified in the request. Note that one of these parameters specifies whether the RTM function itself is to be active for the LU.

Each LU LMS retains a single set of RTM parameter settings to use. It receives a default set at system initialization time, and subsequently overwrites them as it receives parameter-setting requests from the Receiving RTM Requests process. When the PU is IPLed or receives an ACTPU (cold), the default parameters supplied at system generation time are restored for all LUs.

Two points should be noted here:

- Since a request to set RTM parameters expects no reply, all exception conditions must be reported on -RSPs. In particular,

the check to see whether the LU local address specified in the request is known to PU configuration services must be done prior to the decision to send a +RSP or a -RSP.

- PU configuration services has knowledge of, and this process is able to communicate with, all LUs configured at the node, not just those that are currently active. If the LU specified in this request is currently inactive, its LMS will accept and save the RTM parameters passed to it on the request, for use whenever the LU does become active.

After passing the RTM parameter settings to the specified LU, this process terminates.

° *Set RTM parameters for all LUs:*

Since this request expects no reply, no request/reply correlation is needed. This process asks PU configuration services for a list of all the LUs configured at this node. It then sends a request to each LU LMS to set its RTM parameters to the values specified in the request. An LMS for an LU that is currently inactive saves the values passed to it, for use when the LU does become active.

After passing the RTM parameter settings to all the LUs, this process terminates.

° *Retrieve RTM data and status for a specified LU:*

This process first builds an entry for the request in its correlation control block. Figure 10-28 shows the structure of this control block entry. The following values are supplied by this process.

- The control point address that was passed to this process from the RECEIVE_REQUEST_SSCP_PU function set group.

  This address is received via MS protocol boundary D item 2.

- The value that came in the PRID field (bits 4-15 of bytes 5-6) of the request NMVT

- A list containing, as its only entry, the LU local address that came in the SNA Address List (X'04') subvector in the request

- A null entry for the NMVT buffer

After creating the control block entry, this process passes to the LMS of the specified LU the request for RTM data. This process then terminates.

° *Retrieve RTM data and status for all LUs that have accumulated data:*

This process first builds an entry for the request in its correlation control block. Figure 10-28 shows the structure of this control block entry. The following values are supplied by this process.

- The control point address that was passed to this process from the RECEIVE_REQUEST_SSCP_PU function set group.

  This address is received via MS protocol boundary D item 2.

- The value that came in the PRID field (bits 4-15 of bytes 5-6) of the request NMVT

- A list containing the local addresses of all the LUs configured at the node; this process obtained the list by asking PU configuration services for it

- A null entry for the NMVT buffer

After creating the control block entry, this process passes to each LU LMS in the node a request for RTM data. This process then terminates.

Table 10-48 in topic 2.10.8.2 lists the sense data that this process can pass to the RECEIVE_REQUEST_SSCP_PU function set group via internal MS protocol boundary E item 2. The address of the CP that sent the request is passed in item 1. The value X'0000 0000' is used to indicate a +RSP. The remaining values, representing the various errors that this process may detect in an RTM request, are to be sent in a -RSP. Sense data X'0870 nn00' is sent for an invalid value in an unformatted subvector. For example, if the RTM measurement definition X'04'is not supported, sense

data X'0870 9408' is used (byte 8 of the X'94' subvector is in error).

The NMVT Buffer:

```
 ___________ ________ _________________ _____________
|  CP ADDR  |  PRID  | LU ADDRESS LIST | NMVT BUFFER |
|___________|________|_________________|_____________|
```

Figure 10-28. EP_RTM's Request/Reply Correlation Control Block

The discussion of the Sending RTM Data process must recognize that there are differences in the process depending upon whether EP_RTM has implemented Elective 2. Recall that Elective 2 allows an implementation of EP_RTM to avoid the overhead associated with lookahead by always sending a reply for its last configured LU in reply to a request for data and status for all LUs with accumulated data, even if that LU has returned only status information. An instance of EP_RTM not choosing to implement Elective 2 does not send a reply in this case, so long as at least one of the preceding LUs has returned both data and status. In summary, Elective 2 eliminates the overhead associated with lookahead, but at the cost of an NMVT flow that is not really necessary.

In order to perform the necessary lookahead, an instance of EP_RTM not opting to implement Elective 2 must make use of an additional column in its correlation control block: the *NMVT buffer*. This buffer holds one NMVT that was built earlier by the Sending RTM Data process. Its purpose is to give the Sending RTM Data process a chance to change the setting of the sequence field in the NMVT header of the buffered NMVT if it turns out that no further replies are forthcoming for the request. A simple example illustrates how this buffer works; see Table 10-6. Note that in this example the first two columns of the correlation control block entry are not shown, since they do not vary.

Table 10-6. Example Illustrating Use of EP_RTM's NMVT Buffer. The example begins with a request for RTM data and status from all LUs with accumulated data, directed to a node at which the four LUs A, B, C, and D are defined.

Sent to CPMS	**LU Addresses**	**Contents of the NMVT Buffer**
° *Receiving NMVTs and Receiving RTM Requests processes receive and process the request* ° *Correlation control block entry for the request is created* ° *Requests for RTM data and status are issued to LUs A, B, C, and D*		
(nothing)	A, B, C, D	(empty)
° *LU A returns data and status* ° *Sending RTM Data process builds NMVT for A* ° *Sending RTM Data process places the NMVT for A in the NMVT buffer*		
(nothing)	B, C, D	NMVT for A (seq. = first)
° *LU B returns data and status* ° *Sending RTM Data process builds NMVT for B* ° *Sending RTM Data process removes the NMVT for A from the NMVT buffer and passes it to the Sending NMVTs process; since another NMVT (for LU B) will also be sent in reply to the request, the sequence setting "first" (rather than "only") is correct.* ° *Sending NMVTs process sends the NMVT for A to CPMS* ° *Sending RTM Data process places the NMVT for B in the RTM buffer*		
NMVT for A (seq. = first)	C, D	NMVT for B (seq. = middle)
° *LU C returns status only* ° *Sending RTM Data process removes C from the correlation control block entry*		
(nothing)	D	NMVT for B (seq. = middle)
° *LU D returns status only* ° *Sending RTM Data process removes D from the correlation control block entry* ° *Sending RTM Data process removes the NMVT for B from the RTM buffer, adjusts its sequence field (from "middle" to "last"), and passes it to the Sending NMVTs process* ° *Sending NMVTs process sends the NMVT for B to CPMS*		

NMVT for B (seq. = "last")	(empty)	(empty)

Sending RTM Data:

This process is started as follows:

- By an LU LMS for the following conditions:

  - To report RTM data and status, or RTM status only if the LU has no accumulated data

  - To report that a request could not be completed

    This will occur only when a request could not be completed and the elective for reporting this condition on a reply has been selected.

The function of this process is to construct an NMVT containing the subvectors described in "Formats Supported"; Table 10-7 shows which subvectors are constructed in which cases.

After building the NMVT, this process starts the process described in "Sending NMVTs" in topic 2.9.5.4.3 (SEND_DATA_SSCP_PU function set group) via protocol boundary A to send the NMVT to CPMS.

Table 10-7. Subvector Conditions of Presence in the RTM (X'0080') Major Vector

Reply Type	**Subvectors present in the RTM (X'0080') major vector**
Positive reply to a request for RTM data and status from an LU with accumulated data	X'91', X'93', X'04'
Positive reply to a request for RTM data and status from an LU with no accumulated data	X'91', X'04'
Negative reply to a request for RTM data and status, if the elective for reporting LU-detected exception conditions in replies has been selected	X'7D', X'04'

Recall the earlier comment in "The NMVT Buffer," that the details of how RTM data is sent to CPMS vary depending on whether Elective 2 has been implemented. Consequently, there are two descriptions of how this data is sent:

Sending Replies If Elective 2 is Implemented:

Except in a single case described below, this process builds an NMVT when it receives RTM data, RTM status, or an SNA sense data from an LU LMS.

- If this process was passed both RTM data and status by the LU LMS when it was started, it builds both the RTM Status Reply (X'91') and RTM Data (X'93') subvectors.

- If this process was passed an SNA sense data by the LU LMS, it builds a Sense Data (X'7D') subvector.

- If this process was passed RTM status without any accompanying data, it must examine its correlation control block to determine what action to take:

  - If the control block entry for this PRID contains other local addresses besides that of the LU that invoked this process, then it does not build an NMVT, and does not start the process described in "Sending NMVTs" in topic 2.9.5.4.3 to send a reply. Its only action is to delete the invoking LU's local address from the control block entry.

    This state of the correlation control block entry occurs only for a request for RTM data and status from all LUs with accumulated data.

  - If the local address of the LU that invoked this process is the only one in the control block entry for this PRID, then this process builds a RTM Status Reply (X'91') subvector.

This state of the correlation control block entry covers three cases:

-- The request was for RTM data and status for a single specified LU.

-- The request was for RTM data and status from all LUs with accumulated data, this status is coming from the last configured LU, and none of the preceding LUs has returned any data.

-- The request was for RTM data and status from all LUs with accumulated data, this status is coming from the last configured LU, and at least one preceding LU has returned both status and data.

° If it has built either the RTM Status Reply (X'91') or the RTM Data (X'93') subvector, this process also builds the SNA Address List (X'04') subvector; see "Building the SNA Address List (X'04') Subvector" in topic 2.10.7.2 for details.

° If it is building an NMVT, then depending on the node's capabilities, this process builds either a Date/Time (X'01') or Relative Time (X'42') subvector; see "Building the Date/Time (X'01') and Relative Time (X'42') Subvectors" in topic 2.10.7.1 for details.

° If it is building an NMVT, then if the data from the LU LMS indicates that RTM counters have been reset, this process builds a Data Reset Flag (X'45') subvector.

° After building all the appropriate subvectors, this process places them in an RTM (X'0080') major vector within an NMVT. See "Building a Management Services Major Vector" in topic 2.10.7.4 and "Building an NMVT" in topic 2.10.7.5 for details. The only constraint on the placement of the subvectors within the major vector is that if it is present, the SNA Address List subvector must appear first.

° After completing an NMVT, this process starts the process described in "Sending NMVTs" in topic 2.9.5.4.3 to send the NMVT to CPMS."Protocol Boundary A--Send NMVT" in topic 2.9.1.3 shows the items passed when starting that process. This process places the following values in these items:

Item 1: A pointer to the NMVT that it has built

Item 2: The address of the control point that sent the request

° After this process has started the "Sending NMVTs" process, it terminates.

Sending Replies If Elective 2 is Not Implemented:

Except in a single case, slightly different from that described earlier, this process builds an NMVT when it receives RTM data, RTM status, or an SNA sense data from an LU LMS.

° If this process was passed both RTM data and status by the LU LMS when it was started, it builds both the RTM Status Reply (X'91') and RTM Data (X'93') subvectors.

° If this process was passed an SNA sense data by the LU LMS, it builds a Sense Data (X'7D') subvector.

° If this process was passed RTM status without any accompanying data, it must examine the correlation control block to determine what action to take. Note that in this case the correlation control block has the fourth column, for the NMVT buffer.

- If the control block entry for this PRID contains other local addresses besides that of the LU that invoked this process, then it does not build an NMVT. Its only action is to delete the invoking LU's local address from the control block entry.

- If the local address of the LU that invoked this process is the only one in the control block entry for this PRID, then this process examines the NMVT buffer:

-- If the buffer is empty, then this process builds an RTM Status (X'91') subvector reporting the status of the LU that invoked it.

-- If the buffer is full, then this process does not build an NMVT for the LU that invoked it. Instead, it updates the sequence field of the NMVT in the buffer (from "first" to "only" or from "middle" to "last"). It then passes the NMVT to the process described in "Sending NMVTs" in topic 2.9.5.4.3 via Protocol Boundary A.

° If it has built either the RTM Status Reply (X'91') or the RTM Data (X'93') subvector, this process also builds the SNA Address List (X'04') subvector; see "Building the SNA Address List (X'04') Subvector" in topic 2.10.7.2 for details.

° If it is building an NMVT, then depending on the node's capabilities, this process builds either a Date/Time (X'01') or Relative Time (X'42') subvector; see "Building the Date/Time (X'01') and Relative Time (X'42') Subvectors" in topic 2.10.7.1 for details.

° If it is building an NMVT, then if the data from the LU LMS indicates that RTM counters have been reset, this process builds a Data Reset Flag (X'45') subvector.

° After building all the appropriate subvectors, this process places them in an RTM (X'0080') major vector within an NMVT. See "Building a Management Services Major Vector" in topic 2.10.7.4 and "Building an NMVT" in topic 2.10.7.5 for details. The only constraint on the placement of the subvectors within the major vector is that if it is present, the SNA Address List subvector must appear first.

° Sending a reply. When Elective 2 is implemented, the decision when to send a reply is straightforward: any reply that is constructed is sent as soon as it is created. The decision is more complicated if EP_RTM does not implement Elective 2:

- Ordinarily, when a reply NMVT is created, it is placed in the NMVT buffer; if the buffer already contains an NMVT, that NMVT is passed to the Sending NMVTs process. When an NMVT is placed in the buffer, its sequence field is set to either "first" or "middle," depending upon whether a reply has already been sent for its request. When an NMVT is removed from the buffer and passed to the Sending NMVTs process because another NMVT is taking its place in the buffer, its sequence field is not changed.

- If the NMVT buffer is empty, and the Sending RTM Data process creates an NMVT for the last remaining LU in the correlation control block entry, then the NMVT is passed directly to the Sending NMVTs process. This NMVT's sequence field is set to "only."

Regardless of whether it is an NMVT it has just built, or one that has been in the NMVT buffer, the Sending RTM Data process starts the Sending NMVTs process in the same way. "Protocol Boundary A--Send NMVT" in topic 2.9.1.3 shows the items passed when starting that process. The Sending RTM Data process places the following values in these items:

Item 1: A pointer to the NMVT that it has built

Item 2: The address of the control point that sent the request

After the Sending RTM Data process has started the Sending NMVTs process, it completes any of the processing described above that it has not finished: e.g., placing an additional NMVT in the NMVT buffer. Then it terminates.

Sending Unsolicited RTM Data:

Unsolicited RTM data is handled the same way by the Sending RTM Data process, regardless of whether it has implemented Elective 2. The only data that this process receives unsolicited from an LU LMS is RTM data and status for the LU. When it receives this data, the process builds the RTM Status Reply (X'91'), the RTM Data (X'93'), the SNA Address List (X'04'), the Data Reset Flag (X'45'), and either the Date/Time (X'01') or the Relative Time (X'42') subvectors; it places these subvectors in an RTM (X'0080') major vector within an NMVT. It then starts the Sending NMVTs process, and passes it the NMVT. "Protocol Boundary A--Send NMVT" in topic 2.9.1.3 shows the items passed when the Sending NMVTs process is started. The Sending RTM Data process places the following values in these items:

Item 1: A pointer to the NMVT that it has built

Item 2: 'NONE' (This data is unsolicited.)

After the Sending RTM Data process has started the Sending NMVTs process, it terminates.

2.10.2.5 EP_RTM Optional Subset 1 (Local Display)

Subtopics:

- 2.10.2.5.1 Functions Provided
- 2.10.2.5.2 Formats Supported
- 2.10.2.5.3 Implementation Requirements

2.10.2.5.1 Functions Provided

EP_RTM optional subset 1 (Local Display) provides the capability to display RTM data at the node implementing this function set group, and to accept commands from the host to enable or disable the local display. The mechanism by which the data is displayed is implementation defined.

2.10.2.5.2 Formats Supported

This optional subset supports the processing of bit 7 (local display of RTM data) in the RTM status and control change mask bytes and RTM status and control indicators bytes of the RTM Control (X'94') subvector. It also supports the setting of bit 7 (local display of RTM data) in the RTM status byte of the RTM Status Reply (X'91') subvector. The setting of this bit is dependant on the status (enabled/disabled) of the local display function.

2.10.2.5.3 Implementation Requirements

The requirements for implementing the EP_RTM optional subset 1 (Local Display) are described by a model consisting of subsets of LU LMS and PU management services.

A Subset of LU LMS:

- Provides a facility for locally displaying RTM data
- Enables and disables the local display of RTM data when requested by PUMS

A Subset of PU Management Services:

- A subset of the "Receiving RTM Requests" process
  - Processes requests containing bit 7 (local display of RTM data) in the RTM status and control change mask bytes and RTM status and control indicator bytes of the RTM Control (X'94') subvector and transfers these requests to an LU LMS
- A subset of the "Sending RTM Data" process
  - Processes LMS data providing the status of local RTM display and transfers data into bit 7 (local display of RTM data) in the RTM status byte of the RTM Status Reply (X'91') subvector

---

2.10.3 EP_QPI Function Set

```
°°°
```

```
°                |                   |         |  EP_QPI function set group  °
°                |                   |         |                             °
°°°°°°°°°°°°°°°°°|°°°°°°°°°°°°°°°°°°°|°°°°°°°°°|°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°
               (A)|                 (D)|      (E)|
                  |                   |         |
                                      |
      .......................    ...........................
      .                     .    .                         .
      .  SEND_DATA_SSCP_PU  .    . RECEIVE_REQUEST_SSCP_PU .
      .  function set group .    .  function set group     .
      .......................    ...........................
```

Figure 10-29. EP_QPI Function Set Group

The EP_QPI function set provides the ability to physically identify a node and optionally, its dependent resources upon request.

Refer to Figure 10-29 throughout the discussion of the EP_QPI function set.

Subtopics:

- 2.10.3.1 Protocol Boundaries with Components Outside EP_QPI
- 2.10.3.2 Prerequisite Function Sets
- 2.10.3.3 Overview of Subsets
- 2.10.3.4 EP_QPI Base Subset

2.10.3.1 Protocol Boundaries with Components Outside EP_QPI

° Input:

- Internal MS protocol boundary D (NMVT Received)

  The EP_QPI function set group receives requests from the RECEIVE_REQUEST_SSCP_PU function set group to process incoming NMVTs. It is passed the NMVT and the name of the CP from which it was received. The details of this protocol boundary are described in "Protocol Boundary D--NMVT Received" in topic 2.9.1.6.

° Output:

- Internal MS protocol boundary A (Send NMVT)

  The EP_QPI function set group requests the SEND_DATA_SSCP_PU function set group to send an NMVT RU on the SSCP-PU session with its controlling CP. The output data consists of the complete NMVT and the address of the control point to which it is to be sent. The details of this protocol boundary are described in "Protocol Boundary A--Send NMVT" in topic 2.9.1.3.

- Internal MS protocol boundary E (Send NMVT Response)

  The EP_QPI function set group requests the RECEIVE_REQUEST_SSCP_PU function set group to send an NMVT response RU on an SSCP-PU session with a specified CP. The output data consists of the CP address and sense data. The details of this protocol boundary are described in "Protocol Boundary E--Send NMVT Response" in topic 2.9.1.7.

2.10.3.2 Prerequisite Function Sets

See "Role Requirements for Management Services Components" in topic 2.8.1 for information on the relationships between this function set and other function sets.

2.10.3.3 Overview of Subsets

```
 ___________________________________________
|                                           |
|     no optional subsets available         |
|         for EP_QPI function set           |
|                                           |
|___________________________________________|
|
```

```
|          EP_QPI base subset              |
|                                          |
|__________________________________________|
```

Figure 10-30. Base and Optional Subsets of EP_QPI Function Set

2.10.3.4 EP_QPI Base Subset

Subtopics:

- 2.10.3.4.1 Functions Provided
- 2.10.3.4.2 Formats Supported
- 2.10.3.4.3 Implementation Requirements

2.10.3.4.1 Functions Provided

The EP_QPI base subset provides the capability to physically identify the SNA node and port-attached devices upon request. For additional details refer to "Query Product ID (QPI)" in topic 1.5.1.

2.10.3.4.2 Formats Supported

The EP_QPI base subset supports the receiving of an NMVT containing a Request Product Set ID (X'8090') major vector.

The EP_QPI base subset supports the sending of NMVTs containing a Reply Product Set ID (X'0090') major vector. The following subvectors are built by EP_QPI:

- A Date/Time (X'01') or Relative Time (X'42') MS common subvector. For details, see "Building the Date/Time (X'01') and Relative Time (X'42') Subvectors" in topic 2.10.7.1.
- An SNA Address List (X'04') MS common subvector. For details, see "Building the SNA Address List (X'04') Subvector" in topic 2.10.7.2.
- A Product Set ID (X'10') MS common subvector. For details, see "Building the Product Set ID (X'10') Subvector" in topic 2.10.7.3.
- A Product Identifier (X'11') MS common subvector. For details, see "Building the Product Set ID (X'10') Subvector" in topic 2.10.7.3.
- An Attached Device Configuration Description (X'82') MS subvector
- A Product Set Attributes (X'84') MS subvector
- An Additional Product Set Attributes (X'86') MS subvector

2.10.3.4.3 Implementation Requirements

The requirements for implementing the EP_QPI base subset are described by a model consisting of subsets of management services and the physical resources manager LMS.

A Subset of physical resources manager LMS: This subset interacts with MS via protocol boundary 1 by providing identification of product sets. The physical resources manager LMS provides the identification of the product set making up the SNA node. If requested by EP_QPI, the physical resources manager LMS also provides identification of product sets in port-attached devices.

For the SNA node and (if requested) each port-attached devices, the physical resources manager LMS replies with a product set identification message. The message contains:

- Sequencing information:

- The only reply for its request

- The first reply for its request

- The last reply for its request

- A middle reply for its request

Note: This sequencing information requires the physical resources manager LMS to "look-ahead" to know if there will be at least one more reply for the product set identification request.

° LU address

° Product information so EP_QPI can build the Product Set ID (X'10') MS common subvector (which includes the Product Identifier (X'11') MS common subvector) and Attached Device Configuration Description (X'82') MS subvector. For details of the X'10' subvector, see "Building the Product Set ID (X'10') Subvector" in topic 2.10.7.3.

A Subset of Management Services: This subset provides the following processes:

° Receiving QPI requests

Upon request by the RECEIVE_REQUEST_SSCP_PU function set group (via protocol boundary D), does the following:

- Parses the request for validity and requests the RECEIVE_REQUEST_SSCP_PU function set group (via protocol boundary E) to send a positive or negative response.

- If the request parsed correctly, processes the request by requesting the physical resources manager LMS (via protocol boundary 1) to provide product set identification. The request from MS to the physical resources manager LMS specifies one of the following:

  -- Only product set identification of the SNA node is requested (i.e., the Request Product Set ID (X'8090') major vector contained a X'81' subvector)

  -- Product set identification of the SNA node and identification of the product set in each port-attached device is requested (i.e., the Request Product Set ID (X'8090') major vector contained a X'83' subvector)

° Sending QPI data

Upon request by a subset of the physical resources manager LMS, does the following:

- Receives (from the physical resources manager LMS via MS application program protocol boundary 1) one data message for the product set comprising the PU and (if requested) one data message for each port-attached product set. The physical resources manager LMS also passes to MS the correlator that it received with the request.

- Uses the correlator to retrieve the PRID from the request.

- Using a data message received from the physical resources manager LMS and the PRID it has retrieved, builds a complete NMVT.

- Passes the NMVT to the SEND_DATA_SSCP_PU function set group via protocol boundary A.

Receiving QPI requests:

This process is started by the process described in "Receiving NMVTs" in topic 2.9.6.4.3 (RECEIVE_REQUEST_SSCP_PU function set group), when the latter has identified the major vector key X'8090' (Request Product Set ID) in a request NMVT. The first task of this process is to parse the product identification major vector. If there is a syntactic error in the major vector, or in one of the subvectors it contains, or if PUMS in the node fails to support some function presupposed by the request, then this process starts the process described in "Sending NMVT Responses" in

topic 2.9.6.4.3 to send the appropriate -RSP. The sense data that can be sent by MS are detailed in "Parsing of NMVTs" in topic 2.10.7.7 and Table 10-49 in topic 2.10.8.3. If no condition requiring a -RSP is found, then this process starts the process described in "Sending NMVT Responses" (via protocol boundary E) to send a +RSP,

This process accepts either the X'81' or the X'83' subvector in the Request Product Set ID (X'8090') major vector.

This process also creates a correlator that it will pass to the physical resources manager LMS with the request for product set identification. EP_QPI retains an association between this correlator and the control point name and PRID from the request. The purpose of this private correlator, used only on protocol boundary 1 is to shield the physical resources manager LMS from differences in, and changes to, the manner in which management services data is transported between nodes. Since knowledge of request/reply correlation is used by transport components as well as by node components to communicate, correlators, such as the PRID, that actually flow between nodes tend to be transport-specific. By mapping the PRID to its own private correlator, EP_QPI insures that the physical resources manager LMS will not be affected by changes to the underlying management services transport.

This process now passes to the physical resources manager LMS (via MS protocol boundary 1) the private correlator and a request for product set identification data. This process then terminates.

"Parsing of NMVTs" in topic 2.10.7.7 and Table 10-49 in topic 2.10.8.3 lists the sense data that this process can send to the RECEIVE_REQUEST_SSCP_PU function set group via internal MS protocol boundary E item 2. (See "Protocol Boundary E--Send NMVT Response" in topic 2.9.1.7.). The address of the CP that sent the request is passed in item 1. The value X'0000 0000' is used to indicate a +RSP. The remaining values, representing the various syntactic errors that this process may detect in a query product identification request, are to be sent in a -RSP.

Sending QPI data:

This process is started by the physical resources manager LMS when it has product set identification data to send. The physical resources manager LMS passes to EP_QPI (via protocol boundary 1) one product set identification message and the private correlator that it received with the request.

This process uses the private correlator passed to it with the data messages to retrieve the PRID and control point name from the original request.

EP_QPI builds subvectors from the product set identification data received from the physical resources manager LMS. See "Formats Supported" in topic 2.10.3.4.2 for the subvectors that must be built.

Using the PRID that it has just retrieved, this process now constructs a complete NMVT enveloping the Reply Product Set ID (X'0090') major vector. The sequence field in byte 7 of NMVT RU must be set to one of the following values based on information received from the physical resources manager LMS with the product set identication:

- B'00', if this is the only Reply Product Set ID (X'0090') major vector for this PRID
- B'01', if this is the last Reply Product Set ID (X'0090') major vector for this PRID
- B'10', if this is the first Reply Product Set ID (X'0090') major vector for this PRID
- B'11', if this is the middle Reply Product Set ID (X'0090') major vector for this PRID

See "Building a Management Services Major Vector" in topic 2.10.7.4 and "Building an NMVT" in topic 2.10.7.5 for details.

This process starts the process described in "Sending NMVTs" in topic 2.9.5.4.3 (SEND_DATA_SSCP_PU function set group) via protocol boundary A, to send the NMVT. "Protocol Boundary A--Send NMVT" in topic 2.9.1.3 shows the items passed when starting that process. This process places the following values in these items:

Item 1: The NMVT it has constructed

Item 2: The address of the control point that sent the request

After this process has started the "Sending NMVTs" process, it terminates.

2.10.4 EP_CHANGE_MGMT Function Set

```
°°°
```

FILE_SERVICES_SUPPORT function set group to send MS data on a SNA/DS conversation. The details of this protocol boundary are described in "Protocol Boundary F--Send MS Bulk Data" in topic 2.9.1.8.

- MS protocol boundary CC (Reporting Cancelation to Operations Management)

  If optional subset 7 (Cancel Command Support) is implemented, the EP_CHANGE_MGMT function set group sends a Cancelation CP-MSU to the EP_OPERATIONS_MGMT function set group for every request that was canceled by an operations management operator. The details of this protocol boundary are described in "Protocol Boundary CC--Reporting Cancelation to Operations Management" in topic 2.9.1.19.

2.10.4.2 Prerequisite Function Sets

See "Role Requirements for Management Services Components" in topic 2.8.1 for information on the relationships between this function set and other function sets.

2.10.4.3 Overview of Subsets

Production Only Activate Support (optional subset 1)	Execution Window Timing Support (optional subset 2)	Activate Report Support (optional subset 3)	Alter Active Install Support (optional subset 4)	Object Disposition Install Support (optional subset 5)	Initiate Command Support (optional subset 6)	Cancel Command Support (optional subset 7)	Activate Last Support (optional subset 8)
EP_CHANGE_MGMT base subset							

Figure 10-32. Base and Optional Subsets of EP_CHANGE_MGMT Function Set

---

2.10.4.4 EP_CHANGE_MGMT Base Subset

Subtopics:

- 2.10.4.4.1 Functions Provided
- 2.10.4.4.2 Formats Supported
- 2.10.4.4.3 Request Activation CP-MSU
- 2.10.4.4.4 Activation Acceptance CP-MSU
- 2.10.4.4.5 Reporting Activation CP-MSU
- 2.10.4.4.6 Implementation Requirements

2.10.4.4.1 Functions Provided

The EP_CHANGE_MGMT base subset provides the capability to respond to change control and activation requests from a change management focal point or local operator interface.

2.10.4.4.2 Formats Supported

The EP_CHANGE_MGMT base subset supports the receiving of a CP-MSU containing either a Request Change Control (X'8050') or a Request Activation (X'8066') major vector.

The base entry point honors only the TRIAL_AND_PRODUCTION request, as indicated in the Change Management Activation Use (X'20') subfield of the Activate (X'81') subvector of the Request Activation (X'8066') major vector, from a focal point (but either type of request from a local

operator).

The following subvectors are included by this function set group in a CP-MSU containing a Change Control (X'0050') major vector as the result of processing a change control request, and passed to the FILE_SERVICES_SUPPORT function set group:

- A Date/Time (X'01') subvector
  - To time-stamp the reply CP-MSU
- A Reporting Installation (X'82') subvector
  - When reporting a successful or unsuccessful attempt to install a change
- A Reporting Removal (X'84') subvector
  - When reporting a successful or unsuccessful attempt to remove a change
- A Reporting Acceptance (X'86') subvector
  - When reporting a successful or unsuccessful attempt to accept a change
- One Reported Change Name (X'88') subvector for each change file referred to in the request major vector and for each of its corequisites
  - To identify a change file referenced by the report
- One Detailed Data (X'98') subvector for each product-unique error
  - To report detailed product-specific data relevant to the change report
- A Reporting Secondary Installation (X'8A') subvector and one or more Secondary Change Name (X'8C') subvectors
  - To report changes not referred to in the request, nor being corequisites of a change referred to in the request, that were installed as part of the operation being reported
- A Reporting Back-Level Status (X'8E') subvector and one or more Back-Level Change Name (X'90') subvectors
  - To report changes not referred to in the request, nor being corequisites of a change referred to in the request, that were put into back-level state as part of the operation being reported
- A Reporting Deletion (X'92') subvector and one or more Deleted Change Name (X'94') subvectors
  - To report changes not referred to in the request nor being a corequisite of a change referred to in the request, that were deleted as part of the operation being reported

In addition, an SNA/FS Action Summary (X'1548') GDS variable is included whenever a change control request required SNA/FS action.

2.10.4.4.3 Request Activation CP-MSU

The structure of the Request Activation CP-MSU is shown in Table 10-8.

Table 10-8. The Request Activation CP-MSU

Structure Name	ID/T	Occurrences Within Parent
CP-MSU	1212	__
Request Activation	8066	1
Execution Window Timing	0B	0-1

Execute Not Before Time	10	1
Activate Command	81	1
Force Indication	10	1
Change Management Activation Use	20	0-1

Table 10-8 shows the formats that change management supports for the Request Activation CP-MSU. A brief description of the required and optional parameters used by change management is provided in this section. For a more detailed description of the formats used by the change management architecture refer to *SNA Formats*.

Required Parameters

° Force Indication (X'10') Activate Subfield

- Force(No): Reject command if target not quiesced (default)

- Force(Yes): Execute command

Optional Parameters

° Change Management Activation Use (X'20') Activate Subfield

° Execution Window Timing (X'0B') MS Common Subvector: Supported by optional subset 2.

2.10.4.4.4 Activation Acceptance CP-MSU

The structure of the Activation Acceptance CP-MSU is shown in Table 10-9.

Table 10-9. The Activation Acceptance CP-MSU

Structure Name	ID/T	Occurrences Within Parent
CP-MSU	1212	__
Activation Acceptance	0066	1
Date/Time	01	1
Activation Acceptance	82	1
Attempt Status	10	1
Additional Reports	14	0-1
Detailed Data	98	°0
SNA Condition Report	1532	0-1

Table 10-9 shows the formats that change management supports for the Activation Acceptance CP-MSU. A brief description of the required and optional parameters used by change management is provided in this section. For a more detailed description of the formats used by the change management architecture refer to *SNA Formats*.

Required Parameters

° Date/Time (X'01') MS Common Subvector

° Attempt Status (X'10') Activation Acceptance Subfield

Optional Parameters

° Detailed Data (X'98') Subvector

- Additional Reports (X'14') Activation Acceptance Subfield:
  Supported by optional subset 3.

- SNA Condition Report (X'1532') GDS Variable

2.10.4.4.5 Reporting Activation CP-MSU

The structure of the Reporting Activation CP-MSU is shown in Table 10-10.

Table 10-10. The Reporting Activation CP-MSU

Structure Name	ID/T	Occurrences Within Parent
CP-MSU	1212	__
Reporting Activation	0067	1
Date/Time	01	1
Reporting Activation	82	1
Activation Status	10	1
Change Management Activation Use Status	20	0-1
Detailed Data	98	°0
SNA Condition Report	1532	0-1

Table 10-10 shows the formats that change management supports for the Reporting Activation CP-MSU, only if optional subset 3 is supported. A brief description of the required and optional parameters used by change management is provided in this section. For a more detailed description of the formats used by the change management architecture refer to *SNA Formats*.

Required Parameters

- Date/Time (X'01') MS Common Subvector

- Activation Status (X'10') Reporting Activation Subfield

Optional Parameters

- Detailed Data (X'98') Subvector

- Change Management Activation Use Status (X'20') Reporting Activation Subfield

- SNA Condition Report (X'1532') GDS Variable

2.10.4.4.6 Implementation Requirements

The requirements for implementing the EP_CHANGE_MGMT base subset are described by a model consisting of a subset of the program supervisor and management services.

A Subset of the Program Supervisor:

- Interacts with MS via protocol boundary 8 as follows:

  - Processes requests from MS to control changes

    -- Installs, removes, and accepts change files on request from MS and reports the result to MS

  - Processes requests from MS to reactivate the entry point.

A Subset of Management Services:

° Provides the following processes:

- Receiving change management requests

  -- Upon request by the FILE_SERVICES_SUPPORT function set group (via protocol boundary G), does the following:

    ° Parses the requests for validity and does the following if an error is detected:

      - Builds a negative reply and passes it to FILE_SERVICES_SUPPORT for forwarding to the requesting LU

      - Builds and sends an Alert for exception conditions that require it

    ° If the request parsed correctly, passes the request to the program supervisor

- Sending change management data

  -- Builds the CP-MSUs described in "Formats Supported" in this section and passes them along with other parameters to the FILE_SERVICES_SUPPORT function set group via protocol boundary F

Receiving Change Management Requests:

This process is started by the process described in "Receiving Requests and Data" in topic 2.9.4.4.3 (FILE_SERVICES_SUPPORT function set), when the latter has identified the major vector key X'8050' (Request Change Control) or X'8066' (Request Activation) in a request CP-MSU.

The input to this process (the CP-MSU contents) was determined at the focal point, at the time the verb was issued, or at the entry point, by way of the local operator interface, as shown in Table 10-11.

Table 10-11. Choosing Major Vector/Subvector and Server Parameters from the Verb

Verb issued by network operator	Major Vector/ Subvector	FS server instructions (E=encoder D=decoder S=source T=target)
Send-and-Install (Destruction Allowed)	X'8050' / X'81'	S:&rel. (FETCH, ABEND, ONLY_ IF_ EXCEPTIONS) T: (CREATE/LOAD_ OR_ REPLACE, ABEND, DETAILED, EXECUTING)
Send-and-Install (Destruction No)	X'8050' / X'81'	S: (FETCH, ABEND, ONLY_ IF_ EXCEPTIONS) T: (CREATE/LOAD, BACKOUT, DETAILED, EXECUTING)
Install	X'8050' / X'81'	E: (ENCODE_ ONLY, ABEND, ONLY_ IF_ EXCEPTIONS) D: (DECODE_ ONLY, ABEND, ONLY_ IF_ EXCEPTIONS)
Remove	X'8050' / X'83'	E: (ENCODE_ ONLY, ABEND, ONLY_ IF_ EXCEPTIONS) D: (DECODE_ ONLY, ABEND, ONLY_ IF_ EXCEPTIONS)
Accept	X'8050' / X'85'	E: (ENCODE_ ONLY, ABEND, ONLY_ IF_ EXCEPTIONS) D: (DECODE_ ONLY, ABEND, ONLY_ IF_ EXCEPTIONS)
Activate	X'8066' / X'81'	SNA/FS action is not requested and there is no server object. SNA/DS service parms are priority=FAST, protection=NO, capacity°4K, security=NO

Notes:

1. For all change management commands with SNA/FS action requested, TARGET_AGENT_REPORTING_ACTION = DETAILED is implied.

2. For Install, Remove, and Accept, the focal point identifies the file using a Stored_Name.

3. For Send-and-Install, the focal point identifies the file using a To-Be-Stored_Name.

The first task of this process is to determine if an SNA/FS error occurred. If so, then it builds a CP-MSU containing an SNA/FS Action Summary (X'1548') GDS variable, and terminates, returning control to the process that started it. The next task of this process is to parse the major vector. If there is a syntactic error in the major vector, or in one of the subvectors it contains, or if MS in the node fails to support some function presupposed by the request, or does not recognize the file name specified in the request, then this process builds and sends the appropriate negative reply in an SNA Condition Report (X'1532') GDS variable. The Format Exception (X'100B xxxx') report codes (as documented in *SNA Formats*) are used. Report codes specific to change control are detailed in Table 10-50 in topic 2.10.8.4.

If the change file named in the server object (the object of the command) is also named in the corequisite list, or if multiple instances of the same change file name occur in the list, this is ignored and processing proceeds.

A finite-state machine is provided to describe how this process checks for state errors. Refer to Figure 10-33. It may be invoked by both a product's SNA/FS application program intervention exit to prevent unwanted storage operations, and by the the program supervisor in handling a change control request. For example, if a SEND_AND_INSTALL uses partial matching, the intervention exit will use the FSM to determine whether to replace a to-be-deleted change, then the program supervisor will use it later to determine the state transition for the change to be installed.

STATE TRANSITION TABLE

State Names:	Sent	Installed on Trial	Installed in Production Removably	Installed in Production Non_ Removably	Back_Level	Reset (not present)
State numbers:	1	2	3	4	5	6
Inputs:						
Create/Load	ÿ (089E0001)	ÿ (089E0001)	ÿ (089E0001)	ÿ (089E0001)	ÿ (089E0001)	1
Replace	6	ÿ (08380001)	ÿ (08380001)	- (b)	ÿ (08380011)	/
Delete	6	ÿ (08380001)	ÿ (08380001)	6 (a)	ÿ (08380011)	ÿ (089A0001)
Fetch	_	_	_	_	_	ÿ (089A0001)
Install on Trial	2	ÿ (08380003)	ÿ (08380004)	ÿ (08380005)	ÿ (08380002)	ÿ (089A0001)
Install in Production Removably	3	3	ÿ (08380004)	ÿ (08380005)	ÿ (08380002)	ÿ (089A0001)
Install in Production Non_ Removably	4	4	ÿ (08380004)	ÿ (08380005)	ÿ (08380002)	ÿ (089A0001)
Remove	ÿ (08380007)	6	6	ÿ (08380006)	ÿ (08380002)	ÿ (089A0001)
Accept	ÿ (08380007)	ÿ (0838000E)	4	ÿ (08380006)	ÿ (08380002)	ÿ (089A0001)

"/" = indicates a should_not_occur condition
"ÿ" = indicates exception condition (followed by exception code)
"n" = indicates state transition (output codes may be listed; see table below)

OUTPUT CODES

Output Code	Description
a	If not critical (else X'0838000F')
b	If agent object does not contain an "Install in production nonremovably" command: X'0838000F' if change is critical, X'08380013' if not If change is required to maintain removability: X'08380011'

"ÿ" = exception condition

"_" = no state transition

Figure 10-33. EP_CHANGE_MGMT Processing State Transitions and Output Codes

Report codes specific to activation are detailed in Table 10-51 in topic 2.10.8.4.

If Activate specifies use of changes installed both on-trial and in-production, then the program supervisor uses on-trial versions of components instead of in-production versions, if they exist. Therefore, it searches for a component (e.g. a microcode module) in the following order: trial version; if not available, production version; if not available, unaltered version.

If the type of activation is orderly and sessions are active, the program supervisor rejects the request. If the type is forced, activation proceeds regardless.

If no condition requiring a negative reply is found, then this process starts the program supervisor. The action taken by the program supervisor depends on the request made. The actions are illustrated in Figure 10-34, Figure 10-35, and Figure 10-36.

Pre_Test	IF a pre_test is required (or desired, and the program supervisor is able to perform it), THEN it is attempted. IF not successful, THEN go to "Reply."
Timing	IF execution window timing is supported and requested, THEN go to "Reply" after saving the request status and setting a timer for later installation at the appropriate date/time.
Installation	The change is installed (along with corequisite changes, if any). IF successful, THEN go to "Post Test."
Automatic Removal	IF automatic removal is required (or desired, and the program supervisor is able to perform it), THEN it is attempted. Go to "Reply."
Post_Test	IF a post_test is required (or desired, and the program supervisor is able to perform it), THEN it is attempted. IF not successful, THEN go to "Automatic Removal."
Automatic Acceptance	IF an automatic acceptance is prohibited, THEN go to "Alter Active Components." IF an automatic acceptance delay is specified, THEN go to "Reply" after saving the request status and setting a timer for later acceptance in the requested number of days. Acceptance is attempted.
Alter Active Components	IF alteration of active components is prohibited, THEN go to "Reply." Alteration of active components is optionally performed.
Reply	Start the process to build reply major vector.

Figure 10-34. Entry Point Processing for Install

```
Timing          IF execution window timing is supported and requested,
                THEN go to "Reply" after saving the request status and setting
                  a timer for later removal at the appropriate date/time.

Removal         The program supervisor attempts to remove the change.

                IF not successful,
                THEN go to "Reply."

Post_Test       IF a post_test is required (or desired, and the program supervisor
                  is able to perform it),
                THEN it is attempted.

                   IF not successful
                   THEN go to "Reply."

                The removed change file is deleted by the program supervisor.

Alter           IF alteration of active components is prohibited,
Active          THEN go to "Reply."
Components
                Alteration of active components is optionally performed.

Reply           Start the process to build reply major vector.
```

Figure 10-35. Entry Point Processing for Remove

```
Timing          IF execution window timing is supported and requested,
                THEN go to "Reply" after saving the request status and setting
                  a timer for later acceptance at the appropriate date/time.

Acceptance      The program supervisor attempts to accept the change.

                The accepted change file is deleted by the program supervisor.

Reply           Start the process to build reply major vector.
```

Figure 10-36. Entry Point Processing for Accept

After this process has started the program supervisor, it terminates, returning control to the process that started it.

Sending Change Management Data:

This process is started by the program supervisor to transfer change management data to MS. When this process is started it is passed this data. This process first constructs a Reporting Installation (X'82'), Reporting Removal (X'84'), Reporting Acceptance (X'86'), or Reporting Activation (X'82') subvector. It then builds other subvectors, described as follows.

A Reported Change Name (X'88') subvector is included for each change data object referenced by the request and for each of its corequisites (note that corequisites may be defined on an install request or by the change object itself).

A Change Object Class (X'96') subvector is included if the global name referenced in any Reported Change Name (X'88') subvector has a value other than MCODE or MCUST as its first token.

One or more Detailed Data (X'98') subvectors are optionally included. Within each such subvector, the only subfield handled by the base focal point is the Detailed Data (X'82') subfield. The other subfield,

Qualified Message Data (X'01'), is not supported. Within each X'82' subfield, change management does not support product set ID indexing (byte 2 = X'00'), only supports the Data ID of STATUS CODE (byte 3 = X'13'), and a Data Encoding of character set 00640-00500 plus (byte 4 = X'11').

A Reporting Secondary Installation (X'8A') subvector, and one or more Secondary Change Name (X'8C') subvectors, is included if a change not referred to in the request (nor one of its corequisites) was installed as part of the operation being reported.

A Reporting Back-Level (X'8E') subvector, and one or more Back-Level Change Name (X'90') subvectors, is included if a change not referred to in the request (nor one of its corequisites) was put into back-level state as part of the operation being reported.

A Reporting Deletion (X'92') subvector, and one or more Deleted Change Name (X'94') subvectors, is included if a change not referred to in the request (nor one of its corequisites) was deleted as part of the operation being reported. Such deletions are change management, not SNA/FS, deletions. SNA/FS deletions are reported in the server object using DELETED_TOKEN_STRING.

The following table shows when secondary installation, back-leveling. and deletions are reported.

Table 10-12. Reporting Subvectors Used Based on State of Primary Change Files

Resultant state of primary change files	Subvectors included in the report
Installed on trial	(X'82', X'88's)
Installed in production removably	(X'82', X'88's), (X'8E', X'90's)
Installed in production non-removably	(X'82', X'88's), (X'92', X'94's)
Removed	(X'84', X'88's), (X'8A', X'8C's) if transiting from installed in production removably, not installed on trial
Accepted	(X'86', X'88's), (X'92', X'94's)
Deleted (explicitly)	SNA/FS deleted token string is used for reporting

Notes:

1. X'92' and X'94' subvectors are used to report change management actions. If an SNA/FS action deleted a change file (for example, in partial matching where the old version is deleted), then the subvectors are not used since such deletion is reported in the server object.

This process then constructs the common subvectors, builds a complete CP-MSU, logs it for later reference (e.g. by a local operator during problem diagnosis), and then starts the process described in "Sending Requests and Data" in topic 2.9.4.4.3 (FILE_SERVICES_SUPPORT function set group via protocol boundary F) to send the report to the focal point. SNA/FS parameters are also returned, if they were provided on the request, to indicate that a file was transferred in the same MU as the CP-MSU. This may occur only if an Install (X'81') subvector was in a Request Change Control (X'8050') major vector originating at the focal point. These parameters are shown in Table 10-13.

Table 10-13. Choosing SNA/FS Server Parameters in the Report Direction

Condition	Major Vector/ Subvector	SNA/FS server instructions (E=encoder D=decoder S=source T=target)
File transferred with request	X'8050' / X'81'	E: (ENCODE_ ONLY, ABEND, ONLY_ IF_ EXCEPTIONS) D: (DECODE_ ONLY, ABEND, ONLY_ IF_ EXCEPTIONS)
File NOT transferred with request	X'8050' / X'81'	none specified (no server object in report MU)
File is NEVER transferred with request	X'8050' / X'83'	none specified (no server object in report MU)
File is NEVER	X'8050' / X'85'	none specified (no server

```
| transferred with     |                  | object in report MU)          |
| request              |                  |                               |
|______________________|__________________|_______________________________|
| File is NEVER        | X'8066' / X'81'  | none specified (no server     |
| transferred with     |                  | object in report MU). SNA/DS  |
| request              |                  | service parms are             |
|                      |                  | priority=FAST, protection=NO, |
|                      |                  | capacity°4K, security=NO.     |
|______________________|__________________|_______________________________|
```

This process then terminates.

2.10.4.5 EP_CHANGE_MGMT Optional Subset 1 (Production-Only Activation Support)

Subtopics:

- 2.10.4.5.1 Functions Provided
- 2.10.4.5.2 Formats Supported
- 2.10.4.5.3 Implementation Requirements

2.10.4.5.1 Functions Provided

EP_CHANGE_MGMT Optional Subset 1 responds to requests from the focal point for activation of only those versions of components marked in-production.

2.10.4.5.2 Formats Supported

This optional subset supports:

° Receiving the Production Only (X'20') value of the Change Management Activation Use (X'20') subfield of the Activate (X'81') subvector of the Request Activation (X'8066') major vector

---

2.10.4.5.3 Implementation Requirements

The details for implementing major vectors containing the subfields used by this subset are described in the base subset.

2.10.4.6 EP_CHANGE_MGMT Optional Subset 2 (Execution Window Timing Support)

Subtopics:

- 2.10.4.6.1 Functions Provided
- 2.10.4.6.2 Formats Supported
- 2.10.4.6.3 Implementation Requirements

2.10.4.6.1 Functions Provided

This optional subset includes support for execution window timing (including automatic acceptance delay and activate force or rejection delay). It requires a time-of-day clock in the entry point accessible by MS.

2.10.4.6.2 Formats Supported

This optional subset supports:

° Receiving the Execute Not Before Time (X'10') subfield of the Execution Window Timing (X'0B') subvector in a major vector

- Receiving the delay setting in the Automatic Acceptance (X'70') subfield in the Install (X'81') subvector in the Request Change Control (X'8050') major vector

- Receiving the delay setting in the Force Indication (X'10') subfield in the Activate (X'81') subvector in the Request Activation (X'8066') major vector.

2.10.4.6.3 Implementation Requirements

The details for implementing major vectors containing the subfields used by this subset are described in the base subset.

2.10.4.7 EP_CHANGE_MGMT Optional Subset 3 (Activation Reporting Support)

Subtopics:

- 2.10.4.7.1 Functions Provided
- 2.10.4.7.2 Formats Supported
- 2.10.4.7.3 Implementation Requirements

2.10.4.7.1 Functions Provided

This optional subset includes support for unsolicited reporting of activation results.

2.10.4.7.2 Formats Supported

This optional subset supports:

- Sending the Additional Reports (X'14') subfield in the Activation Acceptance (X'82') subvector in the Activation Acceptance (X'0066') major vector

- Sending the Activation (X'0067') major vector as supported in the sending of the Activation Acceptance (X'0066') major vector by the base subset to the change management focal point whenever reactivation of the entry point occurs. However, instead of including the Activation Acceptance (X'82') subvector, a Reporting Activation (X'82') subvector is included to report a successful or unsuccessful activation attempt.

2.10.4.7.3 Implementation Requirements

The details for implementing major vectors containing the subfields used by this subset are described in the base subset.

2.10.4.8 EP_CHANGE_MGMT Optional Subset 4 (Alter Active Support)

Subtopics:

- 2.10.4.8.1 Functions Provided
- 2.10.4.8.2 Formats Supported
- 2.10.4.8.3 Implementation Requirements

2.10.4.8.1 Functions Provided

EP_CHANGE_MGMT Optional Subset 4 responds to requests for installation that specify whether alteration of activate components is allowed.

2.10.4.8.2 Formats Supported

```
This optional subset supports:

°   Receiving the Alter Active Components (X'80') subfield of the Install
    (X'81') and Remove (X'83') subvectors of the Request Change Control
    (X'8050') major vector.
```

2.10.4.8.3 Implementation Requirements

```
The details for implementing major vectors containing the subfields used
by this subset are described in the base subset.
```

2.10.4.9 EP_CHANGE_MGMT Optional Subset 5 (Object Disposition Support)

Subtopics:

- 2.10.4.9.1 Functions Provided
- 2.10.4.9.2 Formats Supported
- 2.10.4.9.3 Implementation Requirements

2.10.4.9.1 Functions Provided

```
EP_CHANGE_MGMT Optional Subset 5 responds to requests for installation
that specify a change object disposition.
```

2.10.4.9.2 Formats Supported

```
This optional subset supports:

°   Receiving the Change Object Disposition (X'90') subfield of the
    Install (X'81') subvector of the Request Change Control (X'8050')
    major vector.
```

2.10.4.9.3 Implementation Requirements

```
The details for implementing major vectors containing the subfields used
by this subset are described in the base subset.
```

2.10.4.10 EP_CHANGE_MGMT Optional Subset 6 (Initiate Command Support)

Subtopics:

- 2.10.4.10.1 Functions Provided
- 2.10.4.10.2 Formats Supported
- 2.10.4.10.3 Implementation Requirements

2.10.4.10.1 Functions Provided

```
EP_CHANGE_MGMT Optional Subset 6 provides the capability to respond to
initiation requests from a change management focal point or local operator
interface.
```

2.10.4.10.2 Formats Supported

This optional subset supports the receiving of a CP-MSU containing a Request Initiation (X'8068') major vector and an Initiate Agent Request (X'1730') parameter major vector.

An Initiate Agent Report (X'1731') parameter major vector, reporting the processing status of the initiation request, is included in the CP-MSU by this function set group, as well as the following subvectors in an Initiation (X'0068') major vector and passed to the FILE_SERVICES_SUPPORT function set group:

° A Date/Time (X'01') subvector

- To time-stamp the reply CP-MSU

° One Detailed Data (X'98') subvector for each product-unique error

- To report detailed product-specific data relevant to the initiation report

In addition, for a CP-MSU containing an Initiation (X'0068') major vector, an SNA/FS Action Summary (X'1548') GDS variable is included whenever an initiation request required SNA/FS action, and an SNA Condition Report (X'1532') GDS variable is included within the Initiate Agent Report (X'1731') to report exception conditions if they exist.

2.10.4.10.3 Implementation Requirements

The requirements for implementing the EP_CHANGE_MGMT Initiate Command Support optional subset are described by a model consisting of a subset of the program supervisor and management services.

A Subset of the Program Supervisor:

° Interacts with MS via protocol boundary 8 as follows:

- Processes requests from MS to initiate command procedures

  -- Initiates command procedures on request from MS and reports the result to MS

A Subset of Management Services:

° Provides the following processes:

- Receiving change management requests

  -- Upon request by the FILE_SERVICES_SUPPORT function set group (via protocol boundary G), does the following:

    ° Parses the requests for validity and does the following if an error is detected:

      - Builds a negative reply and passes it to FILE_SERVICES_SUPPORT for forwarding to the requesting LU

      - Builds and sends an Alert for exception conditions that require it

    ° If the request parsed correctly, passes the request to the program supervisor

- Sending change management data

  -- Builds the CP-MSUs described in "Formats Supported" in this section and passes them along with other parameters to the FILE_SERVICES_SUPPORT function set group via protocol boundary F

Receiving Change Management Requests:

This process is started by the process described in "Receiving Requests and Data" in topic 2.9.4.4.3 (FILE_SERVICES_SUPPORT function set), when the latter has identified the major vector key X'8068' (Request Initiation) in a request CP-MSU.

The input to this process (the CP-MSU contents) was determined at the focal point, at the time the verb was issued, or at the entry point, by way of a local operator interface, as shown in Table 10-14.

Table 10-14. Choosing Major Vector/GDS variable and Server Parameters from the Verb

Verb issued by network operator	Major Vector/ GDS variable	FS server instructions (E=encoder D=decoder S=source T=target)
Send-and-Initiate (Destruction Allowed)	X'8068' / X'1730'	S: (FETCH, ABEND, ONLY_ IF_ EXCEPTIONS) T: (CREATE/LOAD_ OR_ REPLACE, ABEND, DETAILED, EXECUTING)
Send-and-Initiate (Destruction No)	X'8068' / X'1730'	S: (FETCH, ABEND, ONLY_ IF_ EXCEPTIONS) T: (CREATE/LOAD, BACKOUT, DETAILED, EXECUTING)
Initiate	X'8068' / X'1730'	E: (ENCODE_ ONLY, ABEND, ONLY_ IF_ EXCEPTIONS) D: (DECODE_ ONLY, ABEND, ONLY_ IF_ EXCEPTIONS)

Notes:

1. For all change management commands with SNA/FS action requested, TARGET_AGENT_REPORTING_ACTION = DETAILED is implied.

2. For Initiate, the focal point identifies the file using a Stored_Name.

3. For Send-and-Initiate, the focal point identifies the file using a To-Be-Stored_Name.

The first task of this process is to determine if an SNA/FS error occurred. If so, then it builds a CP-MSU containing an SNA/FS Action Summary (X'1548') GDS variable, and terminates, returning control to the process that started it. The next task of this process is to parse the major vector and accompanying GDS variable. If there is a syntactic error in the GDS variable or major vector, or in one of the subvectors it contains, or if MS in the node fails to support some function presupposed by the request, or does not recognize the file name specified in the request, then this process builds and sends the appropriate negative reply in an SNA Condition Report (X'1532') GDS variable. The Format Exception (X'100B xxxx') report codes (as documented in *SNA Formats*) are used. No system report codes are specific to command procedure initiation.

If no condition requiring a negative reply is found, then this process starts the program supervisor, which initiates the command procedure. After doing so, the program supervisor either keeps or deletes the command procedure object, based on the Initiate request specification. If an object disposition is not present on the request, the program supervisor determines the final disposition of the initiated object.

After this process has started the program supervisor, it terminates, returning control to the process that started it.

Sending Change Management Data:

This process is started by the program supervisor to transfer change management data to MS. When this process is started it is passed this data. This process first constructs an Initiate Agent Report (X'1731') GDS variable. It then builds subvectors, described as follows.

One or more Detailed Data (X'98') subvectors are optionally included. Within each such subvector, the only subfield handled by the base focal point is the Detailed Data (X'82') subfield. The other subfield, Qualified Message Data (X'01'), is not supported. Within each X'82' subfield, change management does not support product set ID indexing (byte 2 = X'00'), but supports only the Data ID of STATUS CODE (byte 3 = X'13'), and a Data Encoding of character set 00640-00500 plus (byte 4 = X'11').

This process then constructs the common subvectors, builds a complete

CP-MSU, logs it for later reference (e.g., by a local operator during problem diagnosis), and then starts the process described in "Sending Requests and Data" in topic 2.9.4.4.3 (FILE_SERVICES_SUPPORT function set group via protocol boundary F) to send the report to the focal point. This report is not sent if the initiation was requested at the local node. When it is sent, SNA/FS parameters are also returned, if they were provided on the request, to indicate that a file was transferred in the same MU as the CP-MSU. These parameters are shown in Table 10-15.

Table 10-15. Choosing SNA/FS Server Parameters in the Report Direction

Condition	Major Vector/ GDS variable	SNA/FS server instructions (E=encoder D=decoder S=source T=target)
File transferred with request	X'8068' / X'1730'	E: (ENCODE_ ONLY, ABEND, ONLY_ IF_ EXCEPTIONS) D: (DECODE_ ONLY, ABEND, ONLY_ IF_ EXCEPTIONS)
File NOT transferred with request	X'8068' / X'1730'	none specified (no server object in report MU)

This process then terminates.

2.10.4.11 EP_CHANGE_MGMT Optional Subset 7 (Cancel Command Support)

Subtopics:

- 2.10.4.11.1 Functions Provided
- 2.10.4.11.2 Formats Supported
- 2.10.4.11.3 Request Cancelation CP-MSU
- 2.10.4.11.4 Cancelation CP-MSU
- 2.10.4.11.5 Implementation Requirements

2.10.4.11.1 Functions Provided

EP_CHANGE_MGMT Optional Subset 7 provides the capability to respond to cancelation requests from a change management focal point or local operator interface.

EP_OPERATIONS_MGMT may also forward a Request Cancelation CP-MSU (sent by the operations management focal point or a second-level application program it serves) to EP_CHANGE_MGMT for processing. Before EP_OPERATIONS_MGMT forwards such a request to EP_CHANGE_MGMT, the MDS and RTI routing information is saved. After EP_CHANGE_MGMT processes the request, it builds at least one report, and possibly more:

- A Cancelation CP-MSU is always built and sent to EP_OPERATIONS_MGMT over MS protocol boundary CC. EP_OPERATIONS_MGMT then encloses the CP-MSU within an MDS-MU (using the MDS and RTI routing information saved from the Request Cancelation) and sends the MDS-MU to the Request Cancelation originator.

- One or more reports are built and returned to CHANGE_MGMT_NETOP for each request that was affected by the cancel process. Each report carries a sense data value that indicates that the related request was canceled because of operator intervention. If such a cancelation process finds no requests to be cancelled, then EP_CHANGE_MGMT does not send any reports to the change management focal point.

See "Operations Management Function Set" in topic 1.7.3 and "EP_OPERATIONS_MGMT Function Set" in topic 2.10.6 for more information.

2.10.4.11.2 Formats Supported

This optional subset supports the receiving of a CP-MSU containing a Request Cancelation (X'8076') major vector and an Agent Unit of Work (X'1549') GDS variable.

2.10.4.11.3 Request Cancelation CP-MSU

The structure of the Request Cancelation CP-MSU is shown in Table 10-16.

Table 10-16. The Request Cancelation CP-MSU

Structure Name	ID/T	Occurrences Within Parent
CP-MSU	1212	__
Request Cancelation	8076	1
Cancel Command	81	1
Cancel Specific Tasks	10	0-2
Cancel All	20	0-2
Cancel All Deferred	30	0-2
Agent Unit of Work Correlator	1549	°0

Table 10-16 shows the formats that change management supports for the Request Cancelation CP-MSU when optional subset 7 is supported. A brief description of the required and optional parameters used by change management is provided in this section. For a more detailed description of the formats used by the change management architecture refer to *SNA Formats*.

Required Parameters: One of three mutually exclusive subfields:

° Cancel Specific Tasks (X'10')

° Cancel All (X'20'), which cancels all requests (both those that are currently active and those that have been accepted but have not yet executed)

° Cancel All Deferred (X'30'), which cancels all of those requests that are pending execution.

Optional Parameters

° Agent Unit of Work Correlator (X'1549') GDS Variable

2.10.4.11.4 Cancelation CP-MSU

The structure of the Cancelation CP-MSU is shown in Table 10-17

Table 10-17. The Cancelation CP-MSU

Structure Name	ID/T	Occurrences Within Parent
CP-MSU	1212	__
Cancelation	0076	1
Date/Time	01	0-1
Reporting Cancelation	82	1
Cancelation Status	10	1
Detailed Data	98	°0
SNA Condition Report	1532	0-1

Table 10-17 shows the formats that change management supports for the Cancelation CP-MSU when optional subset 7 is supported. A brief description of the required and optional parameters used by change management is provided in this section. For a more detailed description of the formats used by the change management architecture refer to *SNA Formats*.

Required Parameters

° Cancelation Status (X'10') Reporting Cancelation Subfield: Accepted

or not accepted, successful or not successful, intermediate, or ancillary.

Optional Parameters

° Date/Time (X'01') MS Common Subvector

° Detailed Data (X'98') Subvector

° SNA Condition Report (X'1532') GDS Variable

When a local operator cancels a request, a Cancelation (X'0076') major vector is never included in a reply CP-MSU. Instead, one reply for each request that was canceled is sent to the change management focal point. Each reply is correlated with the request (with which it is associated) that was canceled. Each reply includes an SNA Condition Report that indicates "canceled because of local operator intervention."

2.10.4.11.5 Implementation Requirements

The requirements for implementing the EP_CHANGE_MGMT Cancel Command Support optional subset are described by a model consisting of a subset of the program supervisor and management services.

A Subset of the Program Supervisor:

° Interacts with MS via protocol boundary 8 as follows:

- Processes requests from MS to cancel an outstanding request

  -- Cancels a request that has been accepted for processing by MS at a node and reports the result of the cancelation to MS

A Subset of Management Services:

° Provides the following processes:

- Receiving change management requests

  -- Upon request by the FILE_SERVICES_SUPPORT function set group (via protocol boundary G), does the following:

    ° Parses the requests for validity and does the following if an error is detected:

      - Builds a negative reply and passes it to FILE_SERVICES_SUPPORT for forwarding to the requesting LU

      - Builds and sends an Alert for exception conditions that require it

    ° If the request parsed correctly, passes the request to the program supervisor

- Sending change management data

  -- Builds the CP-MSUs described in "Formats Supported" in this section and passes them along with other parameters to the FILE_SERVICES_SUPPORT function set group via protocol boundary F

Receiving Change Management Requests:

This process is started by the process described in "Receiving Requests and Data" in topic 2.9.4.4.3 (FILE_SERVICES_SUPPORT function set), when the latter has identified the major vector key X'8076' (Request Cancelation) in a request CP-MSU.

The input to this process (the CP-MSU contents) was determined at the focal point, at the time the Cancel verb was issued, or at the entry point, by way of a local operator interface. The X'8076' major vector includes a Cancel (X'81') subvector and is accompanied by an Agent Unit of Work Correlator (X'1549') GDS variable that identifies the request to be canceled. Since an SNA/FS action is never requested with a Cancel verb, no server object is present in the SNA/DS MU.

The first task of this process is to parse the major vector and accompanying GDS variable. If a syntactic error exists in the GDS variable or major vector, or in one of the subvectors it contains, or if MS in the node fails to support some function presupposed by the request, then this process builds and sends the appropriate negative reply in an SNA Condition Report (X'1532') GDS variable. The Format Exception (X'100B xxxx') report codes (as documented in *SNA Formats*) are used. Report codes specific to command cancelation are detailed in Table 10-52 in topic 2.10.8.4.

If no condition requiring a negative reply is found, then this process starts the program supervisor, which cancels the appropriate unit of work.

After this process has started the program supervisor, it terminates, returning control to the process that started it.

Sending Change Management Data:

This process is started by the program supervisor to transfer change management data to MS. When this process is started it is passed this data. This process first constructs a Reporting Cancelation (X'82') subvector. It then builds subvectors, described as follows.

One or more Detailed Data (X'98') subvectors are optionally included. Within each such subvector, the only subfield handled by the base focal point is the Detailed Data (X'82') subfield. The other subfield, Qualified Message Data (X'01'), is not supported. Within each X'82' subfield, change management does not support product set ID indexing (byte 2 = X'00'), only supports the Data ID of STATUS CODE (byte 3 = X'13'), and a Data Encoding of character set 00640-00500 plus (byte 4 = X'11').

This process then constructs the common subvectors, builds a complete CP-MSU, logs it for later reference (e.g., by a local operator during problem diagnosis), and then starts the process described in "Sending Requests and Data" in topic 2.9.4.4.3 (FILE_SERVICES_SUPPORT function set group via protocol boundary F) to send the report to the focal point. No SNA/FS parameters are returned when the report is sent, since they are never provided on the request.

This process then terminates.

2.10.4.12 EP_CHANGE_MGMT Optional Subset 8 (Last-Used Activation Support)

Subtopics:

- 2.10.4.12.1 Functions Provided
- 2.10.4.12.2 Formats Supported
- 2.10.4.12.3 Implementation Requirements

2.10.4.12.1 Functions Provided

EP_CHANGE_MGMT Optional Subset 8 responds to requests from the focal point for activation to either "trial" or "production" based on the setting used in the previous activation.

2.10.4.12.2 Formats Supported

This optional subset supports:

° Receiving the Last Used (X'30') value of the Change Management Activation Use (X'20') subfield (CMAU) of the Activate (X'81') subvector of the Request Activation (X'8066') major vector

---

2.10.4.12.3 Implementation Requirements

If the last activation was accomplished using only production components, then CMAU(LAST) is translated by the entry point (when it has kept a history of past CMAU usage) to CMAU(PROD).

If the last activation was accomplished using any trial components, then CMAU(LAST) is translated by the entry point (when it has kept a history of past CMAU usage) to CMAU(TRIAL).

If the entry point does not support change management, or has not kept a history of past CMAU usage, then a CMAU(LAST) is effectively treated as a CMAU(PROD). The command report includes a sense data value to inform the focal point of the specific condition.

The details for implementing major vectors containing the subfields used by this subset are described in the base subset.

2.10.4.13 Example Flows

Subtopics:

- 2.10.4.13.1 Non-Destructive SEND_AND_INSTALL
- 2.10.4.13.2 Failing Non-Destructive SEND_AND_INSTALL
- 2.10.4.13.3 Destructive SEND_AND_INSTALL
- 2.10.4.13.4 Failing INSTALL

2.10.4.13.1 Non-Destructive SEND_AND_INSTALL

Consider the three nodes in a SNA/DS network depicted below. Assume that all three nodes happen to be in the same network (NET1).

```
   Network Operator

         |
         |
    _____|__________                                 _______________
   |     |          |                               |               |
   |              |                               |               |  Remote Production Node
   |             __|_______________________________|_ÿ   Entry     |
   |               |                               |     Point C   |
   |               |                               |_______________|
   |               |
   |               |
   |               |
   |               |
   |               |  Local Distribution Node
   |        Focal  |
   |     "  Point A |
   |     |         |
   |_____|_________|
         |
         |
      ___|________
     |   |        |
     |           |  Local Development Node
     |    Entry  |
     |    Point B |
     |____________|
```

Figure 10-37. Sample configuration

Given the configuration in Figure 10-37, in the example that follows, a microcode customizing data change file with SNA/FS global name MCUST.9135.NA.NET1.C.SYSTEM.V2 has been prepared at a development node (entry point B) and sent, in an unsolicited manner, to a distribution node (focal point A).

A network operator at A wishes to send and install the change file at a remote production node (entry point C). He chooses the following installation options:

- Trial installation
- Removability required
- Pre-test required
- Automatic Removal if test failure or installation failure
- Post-test not to be performed
- Automatic Acceptance prohibited
- Alteration of Active Components to be allowed

A discussion of these Install options can be found in "Change Control" in topic 1.6.2.

Network operator at A issues request:

In the architected model, the network operator then assigns a sequence number and issues the Send_and_Install Change Management request protocol boundary verb which is defined in Appendix B, "Management Services Protocol Boundary Verbs" in topic B.0 (see Table 10-18). However, a typical implementation will assign the sequence number automatically before issuing the verb.

```
                 Network Operator
                        |
                 _______|_______
                |       |       |
                |CHANGE_        |        FOCAL POINT A
                |MGMT_          |
                |NETOP          |
                |               |
                |_______________|
                |               |
                |FILE_          |
                |SERVICES_      |
                |SUPPORT        |
                |               |
                |_______________|_________________________ _____________
                |                                        |             |
                |                                        |             |     /----------\
                |                                        |             |     |          |
                |                                        |   SNA/FS    |     | Storage  |
                |                 SNA/DS                 |   Server    |=====|          |
                |                                        |             |     |          |
                |                                        |             |     \__________/
                |                                        |             |
                |                                        |             |
                |________________________________________|_____________|
```

Figure 10-38. Non-Destructive Send_and_Install (Part 1 of 6). Network operator at A issues Send_and_Install verb to CHANGE_MGMT_NETOP.

The parameters supplied on the Send_and_Install verb by the network operator are:

Table 10-18. Send_and_Install Supplied Parameters

Parameter Name	Parameter Value	Parameter Meaning
CORRELATOR	X'203'	Correlation value for this unit of work
TO_BE_FETCHED_NAME	MCUST, 9135, NA, NET1, C, SYSTEM, V2	SNA/FS Global Name of file to be fetched
TARGET_LIST TARGET_LOCATION	NET1.C	Location(s) to send change file
DESTRUCTION	NO	Prohibit SNA/FS from overwriting any files
REMOVABILITY	YES	Install file so it can be processed by subsequent Remove protocol boundary verb

AUTOMATIC_REMOVAL	YES	Remove Change File if installation or test fails
PRE_TEST	YES	Perform pre-test
POST_TEST	NO	Do not perform post-test
AUTOMATIC_ACCEPTANCE	NO	Do not automatically accept change file if installation and test succeed
ACTIVATION_USE	TRIAL	Components altered by installation will be accessed only by trial activation
ALTER_ACTIVE_COMPONENT	ALLOWED	Components altered by installation may be applied to running code

Request message unit is built at A and sent to C:

CHANGE_MGMT_NETOP at A builds the request CP-MSU, identifies the target destination as NET1.C, chooses server parameters and assigns an agent unit-of-work correlator. It then passes this data to FILE_SERVICES_SUPPORT which issues a Send_Distribution protocol boundary verb for SNA/DS (see Table 10-19).

As a result, the SNA/FS server fetches the change file and encodes the server object. SNA/DS then builds and sends a message unit (MU) to remote node C.

```
 _______________
|               |
|CHANGE_        |            FOCAL POINT A
|MGMT_          |
|NETOP_   |     |
|         |     |
|_________|_____|
|         |     |
|FILE_    |     |
|SERVICES_|     |
|SUPPORT  |     |
|         |     |
|_________|_____|_________________________ _____________
|         |                               |             |
|                                         |             |     /----------\
|             ................ÿ.............|.............|..ÿ..|.....     |
|                                         |    SNA/FS   |     | Storage  |
|                 SNA/DS                  |    Server   |=====|          |
|                                 ..........|.............|....|.....     |
|                                 .         |             |     \__________/
|                                 .         |             |
|                                 .         |             |
|_________________________________________|_____________|
                                  .
                                  .........ÿ........  (MU)
```

Figure 10-39. Non-Destructive Send_and_Install (Part 2 of 6). CHANGE_MGMT_NETOP at A invokes FILE_SERVICES_SUPPORT which issues Send_Distribution.

The parameters supplied on the SNA/DS Send_Distribution verb are:

Table 10-19. Send_Distribution Supplied Parameters

Parameter Name	Parameter Value	Parameter Meaning
DISTRIBUTION_ID		
ORIGIN_DSU	NET1.A	DSU name of dist origin
ORIGIN_AGENT	X'23F0F0F0'	Indicates SNA/MS
ORIGIN_SEQNO	1234	Sequence number
AGENT_CORREL	Agent correlation value	See *SNA Formats*

DESTINATION	NET1.C	DSU name of recipient
DEST_AGENT	X'23F0F0F0'	Indicates SNA/MS
AGENT_OBJECT	CP-MSU consisting of: Request Change Control Major Vector consisting of: Install subvector	See *SNA Formats*
SERVICE_PARMS		
PRIORITY	DATA8	Use at least this priority
PROTECTION	LEVEL2	Safe Store must be used
CAPACITY	16MEG	Handle at least 16M object
SECURITY	LEVEL2	Security required
REPORTING_REQUESTED		
EXCEPTION	YES	Request Exception Report
INTEGRITY	HIGH	Use confirmation protocols
REPORT_TO_DSU	NET1.A	DSU to send dist reports to
SERVER	X'24F0F0F0'	Indicates SNA/File Services
SPECIFIC_SERVER_INFO		As defined by SNA/FS
SOURCE_INSTRUCTION	FETCH	
	ABEND	
	ONLY_IF_EXCEPTIONS	
TARGET_INSTRUCTION	CREATE/LOAD	
	EXECUTING	
	BACKOUT	
	DETAILED	
D_O_CLASS	X'10200000'	
D_O_GLOBAL_NAME	TO_BE_FETCHED_NAME (MCUST, 9135, NA, NET1, C, SYSTEM, V2)	

Request message unit is received at C:

SNA/DS at C receives the message unit and the SNA/FS server decodes the server object and stores the change file. Then SNA/DS invokes FILE_SERVICES_SUPPORT, which issues the Receive_Distribution SNA/DS protocol boundary verb and is returned parameters (see Table 10-20).

FILE_SERVICES_SUPPORT then invokes EP_CHANGE_MGMT, passing it the CP-MSU, source destination (NET1.A), server parameters and agent unit-of-work correlator.

```
|         |                              |             |     /----------\
|              ........................|.............|....|....      |
|                                      |    SNA/FS   |     |  Storage |
|                SNA/DS                |    Server   |=====|          |
|                            .......ÿ......|.............|..ÿ..|....      |
|                            .           |             |     \__________/
|                            .           |             |
|                            .           |             |
|_____________________________________|_____________|
                             .
          (MU)  ........ÿ........
```

Figure 10-40. Non-Destructive Send_and_Install (Part 3 of 6). FILE_SERVICES_SUPPORT at C issues Receive_Distribution and is returned parameters. It then invokes EP_CHANGE_MGMT.

The parameters returned on the SNA/DS Receive_Distribution verb are:

Table 10-20. Receive_Distribution Returned Parameters

Parameter Name	Parameter Value	Parameter Meaning
DISTRIBUTION_ID		
ORIGIN_DSU	NET1.A	DSU name of dist origin
ORIGIN_AGENT	X'23F0F0F0'	Indicates SNA/MS
ORIGIN_SEQNO	1234	Sequence number
AGENT_CORREL	Agent correlation value	See *SNA Formats*
AGENT_OBJECT	CP-MSU consisting of: Request Change Control Major Vector consisting of: Install subvector	See *SNA Formats*
REPORTING_REQUESTED		
EXCEPTION	YES	Request Exception Report
REPORT_TO_DSU	NET1.A	DSU to send dist reports to
DISTRIBUTION_TIME	hh.mm.ss.hh.GMT	GMT time at which distribution originated
SPECIFIC_SERVER_REPORT		As defined by SNA/FS
SERVER_SUMMARY_REPRT	ALL_SUCCESSFUL	
TARGET_INSTRUCTION	CREATE/LOAD	
	EXECUTING	
	BACKOUT	
	DETAILED	
D_O_CLASS	X'10200000'	
D_O_LOCAL_NAME	LOCAL.NAME	
D_O_GLOBAL_NAME	STORED_NAME (MCUST, 9135, NA, NET1, C, SYSTEM, V2)	

Response message unit is built at C and sent to A:

EP_CHANGE_MGMT at C parses the CP-MSU, extracting the Request Change Control major vector. Based on the Install subvector encodings, a pre-test is performed successfully on the change file and it is installed on trial removably. To report installation, it builds a reply CP-MSU, identifies the target destination as NET1.A, chooses server parameters and specifies the agent unit-of-work correlator. Finally, it invokes

FILE_SERVICES_SUPPORT which issues a Send_Distribution protocol boundary verb for SNA/DS (see Table 10-21).

As a result, the SNA/FS server encodes a new server object and SNA/DS builds and sends a message unit back to node A.

```
                 _________________
                |                 |
                |EP_              |            ENTRY POINT C
                |CHANGE_   |      |
                |MGMT      |      |
                |          |      |
                |__________|______|
                |          |      |
                |FILE_     |      |
                |SERVICES_|       |
                |SUPPORT   |      |
                |          |      |
                |__________|______|_________________________ _______________
                |          |                                |              |
                |                                          |              |     /----------\
                |              .............ÿ...............|..............|..ÿ..|....      |
                |                                          |    SNA/FS    |     | Storage  |
                |                 SNA/DS                   |    Server    |=====|          |
                |                              ...........|..............|....|....      |
                |                              .           |              |     \__________/
                |                              .           |              |
                |                              .           |              |
                |______________________________________________|______________|
                                               .
                                               ........ÿ........  (MU)
```

Figure 10-41. Non-Destructive Send_and_Install (Part 4 of 6). EP_CHANGE_MGMT at C invokes FILE_SERVICES_SUPPORT which issues Send_Distribution.

The parameters supplied on the SNA/DS Send_Distribution verb are:

Table 10-21. Send_Distribution Supplied Parameters

Parameter Name	Parameter Value	Parameter Meaning
DISTRIBUTION_ID		
ORIGIN_DSU	NET1.C	DSU name of dist origin
ORIGIN_AGENT	X'23F0F0F0'	Indicates SNA/MS
ORIGIN_SEQNO	5678	Sequence number
AGENT_CORREL	Agent correlation value	See *SNA Formats*
DESTINATION	NET1.A	DSU name of recipient
DEST_AGENT	X'23F0F0F0'	Indicates SNA/MS
AGENT_OBJECT	CP-MSU consisting of:	See *SNA Formats*
	Change Control Major Vector consisting of:	
	Date/Time subvector	
	Reporting Installatio subvector	
	Reported Change Name subvector	
	FS Action Summary GDS variable	
SERVICE_PARMS		
PRIORITY	DATA8	Use at least this priority
PROTECTION	LEVEL2	Safe Store must be used
CAPACITY	16MEG	

SECURITY	LEVEL2	Handle at least 16M object Security required
REPORTING_REQUESTED		
EXCEPTION	YES	Request Exception Report
INTEGRITY	HIGH	Use confirmation protocols
REPORT_TO_DSU	NET1.A	DSU to send dist reports to
SERVER	X'24F0F0F0'	Indicates SNA/File Services
SPECIFIC_SERVER_INFO		As defined by SNA/FS
ENCODER_INSTRUCTIO	ENCODE_ONLY ABEND ONLY_IF_EXCEPTIONS	
DECODER_INSTRUCTIO	DECODE_ONLY ABEND ONLY_IF_EXCEPTIONS	
D_O_CLASS	X'10200000'	
D_O_GLOBAL_NAME	STORED_NAME (MCUST, 9135, NA, NET1, C, SYSTEM, V2)	

Response message unit is received at A:

SNA/DS at A receives the message unit and the SNA/FS server decodes the server object. Then SNA/DS invokes FILE_SERVICES_SUPPORT, which issues the Receive_Distribution SNA/DS protocol boundary verb and is returned parameters (see Table 10-22).

FILE_SERVICES_SUPPORT then invokes CHANGE_MGMT_NETOP, passing it the CP-MSU, source destination (NET1.C), server parameters and agent unit-of-work correlator.

```
                 _________________
                |                 |
                |CHANGE_          |            FOCAL POINT A
                |MGMT_            |
                |NETOP    "       |
                |         |       |
                |_________|_______|
                |         |       |
                |FILE_    |       |
                |SERVICES_|       |
                |SUPPORT  |       |
                |         |       |
                |_________|_______|_________________________ _____________
                |         |                               |             |
                |         |                               |             |     /----------\
                |            ...............................|.............|....|....      |
                |                                         |    SNA/FS   |     | Storage  |
                |                 SNA/DS                  |    Server   |=====|          |
                |                                 .....ÿ.....|.............|..ÿ..|....      |
                |                                 .         |             |     \__________/
                |                                 .         |             |
                |                                 .         |             |
                |_________________________________________|_____________|
                                                  .
                          (MU)   .........ÿ.........
```

Figure 10-42. Non-Destructive Send_and_Install (Part 5 of 6). FILE_SERVICES_SUPPORT at A issues Receive_Distribution and is returned parameters. It then invokes CHANGE_MGMT_NETOP.

The parameters returned on the SNA/DS Receive_Distribution verb are:

Table 10-22. Receive_Distribution Returned Parameters

Parameter Name	Parameter Value	Parameter Meaning
DISTRIBUTION_ID		
ORIGIN_DSU	NET1.C	DSU name of dist origin
ORIGIN_AGENT	X'23F0F0F0'	Indicates SNA/MS
ORIGIN_SEQNO	5678	Sequence number
AGENT_CORREL	Agent correlation value	See *SNA Formats*
AGENT_OBJECT	CP-MSU consisting of: Change Control Major Vector consisting of: Date/Time subvector Reporting Installatio subvector Reported Change Name subvector FS Action Summary GDS variable	See *SNA Formats*
REPORTING_REQUESTED		
EXCEPTION	YES	Request Exception Report
REPORT_TO_DSU	NET1.A	DSU to send dist reports to
DISTRIBUTION_TIME	hh.mm.ss.hh.GMT	GMT time at which distribution originated
SPECIFIC_SERVER_REPORT		As defined by SNA/FS
DECODER_INSTRUCTIO	DECODE_ONLY ABEND ONLY_IF_EXCEPTIONS	
D_O_CLASS	X'10200000'	
D_O_GLOBAL_NAME	STORED_NAME (MCUST, 9135, NA, NET1, C, SYSTEM, V2)	

Network operator at A is issued response:

After parsing the reply CP-MSU and extracting data, CHANGE_MGMT_NETOP at A builds and issues, to the network operator, the Reporting_Installation protocol boundary verb for Change Management defined in Appendix B, "Management Services Protocol Boundary Verbs" in topic B.0 (see Table 10-23). Note that the Correlator value is the same as that originally provided by the network operator on the Send_and_Install verb.

```
                Network Operator

                        "
                 ________|________
                |        |        |
                |CHANGE_ |        |            FOCAL POINT A
                |MGMT_           |
                |NETOP           |
                |                |
                |________________|
                |                |
                |FILE_           |
                |SERVICES_       |
                |SUPPORT         |
                |                |
                |________________|___________________________ _____________
                |                                           |             |
```

```
                |                                       |              |     /----------\
                |                                       |              |     |          |
                |                                       |    SNA/FS    |     | Storage  |
                |                 SNA/DS                |    Server    |=====|          |
                |                                       |              |     |          |
                |                                       |              |     \__________/
                |                                       |              |
                |                                       |              |
                |_______________________________________|______________|
```

Figure 10-43. Non-Destructive Send_and_Install (Part 6 of 6). CHANGE_MGMT_NETOP at A issues Reporting_Installation verb to network operator.

The parameters returned on the Reporting_Installation verb to the network operator are:

Table 10-23. Reporting_Installation Returned Parameters

Parameter Name	Parameter Value	Parameter Meaning
CORRELATOR	X'203'	Correlates report to initial request
TIME_STAMP	Date/Time of Installation	From Date/Time subvector in returned CP-MSU
TARGET_LOCATION	NET1.C	Location where file stored
STORED_NAME	MCUST, 9135, NA, NET1, C, SYSTEM, V2	SNA/FS Global Name of Stored File
FS_ACTION_SUMMARY		As defined by SNA/FS
	ALL_SUCCESSFUL	
	NO_BACKOUT_ATTEMPTE	
	ABEND_NOT_APPLICABL	
	CONTINUE_NOT_ATTEMP	ED
INSTALLATION_RESULTS		
INSTALLATION_STATU	SUCCESSFUL	Installation successful
WHEN_EFFECTIVE	ACTIVATION_REQUIRED	Upon activation
REPORTED_CHANGE_NA	E_LIST	Change file(s) installed
CHANGE_FILE_NA	E MCUST, 9135, NA, NET1, C, SYSTEM, V2	
PRE_TEST_STATUS	SUCCESSFUL	Test was successful
POST_TEST_STATUS	NOT_ATTEMPTED	Test not attempted
AUTOMATIC_REMOVAL_RESU	TS	
AUTOMATIC_REMOVAL_	TATUSNOT_ATTEMPTED	Removal not attempted
WHEN_EFFECTIVE	NOT_APPLICABLE	Effective time not applicable
AUTOMATIC_ACCEPTANCE_S	ATUS NOT_ATTEMPTED	Acceptance not attempted
REMOVABILITY_STATUS	INSTALLED_REMOVABLY	Change file can be removed
ACTIVATION_USE_STATUS	TRIAL	Change file installed on trial

2.10.4.13.2 Failing Non-Destructive SEND_AND_INSTALL

Consider the three nodes in a SNA/DS network depicted below. Assume that all three nodes happen to be in the same network (NET1).

```
Network Operator

        |
        |
 ____|_________                                 _____________
|    |         |                               |             |
|              |                               |             |  Remote Production Node
|            __|_______________________________|_ÿ   Entry   |
|              |                               |     Point C |
|              |                               |_____________|
|              |
|              |
|              |
|              |
|              |
|              |  Local Distribution Node
|       Focal  |
|     "  Point A |
|    |         |
|____|_________|
       |
       |
    ___|_______
   |   |       |
   |           |  Local Development Node
   |    Entry  |
   |    Point B |
   |___________|
```

Figure 10-44. Sample Configuration

Given the configuration in Figure 10-44, in the example that follows, a microcode customizing data change file with SNA/FS global name MCUST.9135.NA.NET1.C.SYSTEM.V3 has been prepared at a development node (entry point B) and sent, in an unsolicited manner, to a distribution node (focal point A). This change file is a new version (V3) of the customizing data change file (V2) successfully installed in the previous example.

A network operator at A wishes to send and install the change file at a remote production node (entry point C). He chooses the following installation options:

- Trial installation
- Removability required
- Pre-test required
- Automatic Removal if test failure or installation failure
- Post-test not to be performed
- Automatic Acceptance prohibited
- Alteration of Active Components to be allowed

A discussion of these Install options can be found in "Change Control" in topic 1.6.2.

In this example, however, the remote production node (entry point C) does not have enough storage space to accommodate the new change file.

Network operator at A issues request:

In the architected model, the network operator then assigns a sequence number and issues the Send_and_Install Change Management request protocol boundary verb which is defined in Appendix B, "Management Services Protocol Boundary Verbs" in topic B.0 (see Table 10-24). However, a typical implementation will assign the sequence number automatically before issuing the verb.

```
                Network Operator
                        |
                 _______|_______
                |       |       |
                |CHANGE_        |        FOCAL POINT A
                |MGMT_          |
                |NETOP          |
                |               |
```

```
                |_______________|
                |               |
                |FILE_          |
                |SERVICES_      |
                |SUPPORT        |
                |               |
                |_______________|_________________________ _____________
                |                                        |             |
                |                                        |             |     /----------\
                |                                        |             |     |          |
                |                                        |    SNA/FS   |     | Storage  |
                |                 SNA/DS                 |    Server   |=====|          |
                |                                        |             |     |          |
                |                                        |             |     \__________/
                |                                        |             |
                |                                        |             |
                |________________________________________|_____________|
```

Figure 10-45. Failing Non-Destructive Send_and_Install (Part 1 of 6). Network operator at A issues Send_and_Install verb to CHANGE_MGMT_NETOP.

The parameters supplied on the Send_and_Install verb by the network operator are:

Table 10-24. Send_and_Install Supplied Parameters

Parameter Name	Parameter Value	Parameter Meaning
CORRELATOR	X'497'	Correlation value for this unit of work
TO_BE_FETCHED_NAME	MCUST, 9135, NA, NET1, C, SYSTEM, V3	SNA/FS Global Name of file to be fetched
TARGET_LIST TARGET_LOCATION	NET1.C	Location(s) to send change file
DESTRUCTION	NO	Prohibit SNA/FS from overwriting any files
REMOVABILITY	YES	Install file so it can be processed by subsequent Remove protocol boundary verb
AUTOMATIC_REMOVAL	YES	Remove Change File if installation or test fails
PRE_TEST	YES	Perform pre-test
POST_TEST	NO	Do not perform post-test
AUTOMATIC_ACCEPTANCE	NO	Do not automatically accept change file if installation and test succeed
ACTIVATION_USE	TRIAL	Components altered by installation will be accessed only by trial activation
ALTER_ACTIVE_COMPONENT	ALLOWED	Components altered by installation may be applied to running code

Request message unit is built at A and sent to C:

CHANGE_MGMT_NETOP at A builds the request CP-MSU, identifies the target destination as NET1.C, chooses server parameters and assigns an agent unit-of-work correlator. It then passes this data to FILE_SERVICES_SUPPORT which issues a Send_Distribution protocol boundary verb for SNA/DS (see Table 10-25).

As a result, the SNA/FS server fetches the change file and encodes the server object. SNA/DS then builds and sends a message unit (MU) to remote node C.

```
                 _______________
                |               |
                |CHANGE_        |            FOCAL POINT A
                |MGMT_          |
                |NETOP_   |     |
                |         |     |
                |_________|_____|
                |         |     |
                |FILE_    |     |
                |SERVICES_|     |
                |SUPPORT  |     |
                |         |     |
                |_________|_____|_________________________ _____________
                |                                        |             |
                |                                        |             |    /----------\
                |             ................ÿ.............|.............|..ÿ..|.....     |
                |                                        |    SNA/FS   |     | Storage  |
                |                 SNA/DS                 |    Server   |=====|          |
                |                                .........|.............|....|.....     |
                |                                .        |             |     \__________/
                |                                .        |             |
                |                                .        |             |
                |________________________________________|_____________|
                                                 .
                                                 .........ÿ........  (MU)
```

Figure 10-46. Failing Non-Destructive Send_and_Install (Part 2 of 6). CHANGE_MGMT_NETOP at A invokes FILE_SERVICES_SUPPORT which issues Send_Distribution.

The parameters supplied on the SNA/DS Send_Distribution verb are:

Table 10-25. Send_Distribution Supplied Parameters

Parameter Name	Parameter Value	Parameter Meaning
DISTRIBUTION_ID		
ORIGIN_DSU	NET1.A	DSU name of dist origin
ORIGIN_AGENT	X'23F0F0F0'	Indicates SNA/MS
ORIGIN_SEQNO	9012	Sequence number
AGENT_CORREL	Agent correlation value	See *SNA Formats*
DESTINATION	NET1.C	DSU name of recipient
DEST_AGENT	X'23F0F0F0'	Indicates SNA/MS
AGENT_OBJECT	CP-MSU consisting of: Request Change Control Major Vector consisting of: Install subvector	See *SNA Formats*
SERVICE_PARMS		
PRIORITY	DATA8	Use at least this priority
PROTECTION	LEVEL2	Safe Store must be used
CAPACITY	16MEG	
SECURITY	LEVEL2	Handle at least 16M object Security required
REPORTING_REQUESTED		
EXCEPTION	YES	Request Exception Report
INTEGRITY	HIGH	Use confirmation protocols
REPORT_TO_DSU	NET1.A	DSU to send dist reports to
SERVER	X'24F0F0F0'	Indicates SNA/File Services
SPECIFIC_SERVER_INFO		As defined by

```
|                       |                        |     SNA/FS            |
|     SOURCE_INSTRUCTION|     FETCH              |                       |
|                       |                        |                       |
|                       |     ABEND              |                       |
|                       |                        |                       |
|                       |     ONLY_IF_EXCEPTIONS |                       |
|                       |                        |                       |
|     TARGET_INSTRUCTION|     CREATE/LOAD        |                       |
|                       |                        |                       |
|                       |     EXECUTING          |                       |
|                       |                        |                       |
|                       |     BACKOUT            |                       |
|                       |                        |                       |
|                       |     DETAILED           |                       |
|                       |                        |                       |
|     D_O_CLASS         |     X'10200000'        |                       |
|                       |                        |                       |
|     D_O_GLOBAL_NAME   |     TO_BE_FETCHED_NAME |                       |
|                       |     (MCUST, 9135, NA,  |                       |
|                       |     NET1, C, SYSTEM,   |                       |
|                       |     V3)                |                       |
|_______________________|________________________|_______________________|
```

Request message unit is received at C:

SNA/DS at C receives the message unit and the SNA/FS server decodes the server object and in attempting to store the change file discovers there is not enough storage space to accommodate the new version. An SNA/FS data object exception report with a CREATING_ALLOCATION_EXCEPTION (X'084C0002') is returned to SNA/DS. SNA/DS then invokes FILE_SERVICES_SUPPORT, which issues the Receive_Distribution SNA/DS protocol boundary verb and is returned parameters (see Table 10-26).

FILE_SERVICES_SUPPORT then invokes EP_CHANGE_MGMT, passing it the CP-MSU, source destination (NET1.A), server parameters and agent unit of work correlator.

```
                 _______________
                |               |
                |EP_            |            ENTRY POINT C
                |CHANGE_        |
                |MGMT     "     |
                |         |     |
                |_________|_____|
                |         |     |
                |FILE_    |     |
                |SERVICES_|     |
                |SUPPORT  |     |
                |         |     |
                |_________|_____|_________________________ _____________
                |         |                               |             |
                |         |                               |             |     /----------\
                |               ............................|.............|....|....      |
                |                                         |    SNA/FS   |    |  Storage |
                |                 SNA/DS                  |    Server   |=====|          |
                |                             .......ÿ......|.............|..ÿ..|....      |
                |                             .           |             |     \__________/
                |                             .           |             |
                |                             .           |             |
                |_____________________________________________________|_____________|
                                              .
                          (MU)  ........ÿ........
```

Figure 10-47. Failing Non-Destructive Send_and_Install (Part 3 of 6). FILE_SERVICES_SUPPORT at C issues Receive_Distribution and is returned parameters. It then invokes EP_CHANGE_MGMT.

The parameters returned on the SNA/DS Receive_Distribution verb are:

Table 10-26. Receive_Distribution Returned Parameters

Parameter Name	Parameter Value	Parameter Meaning
DISTRIBUTION_ID		
ORIGIN_DSU	NET1.A	DSU name of dist origin
ORIGIN_AGENT	X'23F0F0F0'	Indicates SNA/MS
ORIGIN_SEQNO	9012	Sequence number
AGENT_CORREL	Agent correlation value	See *SNA Formats*

AGENT_OBJECT	CP-MSU consisting of: Request Change Control Major Vector consisting of: Install subvector	See *SNA Formats*
REPORTING_REQUESTED		
EXCEPTION	YES	Request Exception Report
REPORT_TO_DSU	NET1.A	DSU to send dist reports to
DISTRIBUTION_TIME	hh.mm.ss.hh.GMT	GMT time at which distribution originated
SPECIFIC_SERVER_REPORT		As defined by SNA/FS
SERVER_SUMMARY_REPRT	NONE_SUCCESSFUL	
	ALL_BACKED_OUT	
	ABEND_NOT_APPLICABLED	
	CONTINUE_NOT_ATTEMPED	
TARGET_INSTRUCTION	CREATE/LOAD	
	EXECUTING	
	BACKOUT	
	DETAILED	
D_O_CLASS	X'10200000'	
D_O_GLOBAL_NAME	TO_BE_STORED_NAME (MCUST, 9135, NA, NET1, C, SYSTEM, V3)	
SNA_CONDITION_REPORT		
SNA_REPORT_COD	X'084C0002'	

Response message unit is built at C and sent to A:

EP_CHANGE_MGMT discovers that an SNA/FS exception code is present in the server parameters. As a result, the CP-MSU is not parsed and the Install command in the Request Change Control major vector is not processed. Instead EP_CHANGE_MGMT builds a reply CP-MSU containing the FS Action Summary, identifies the target destination as NET1.A, chooses server parameters and specifies the agent unit-of-work correlator. Finally, it invokes FILE_SERVICES_SUPPORT which issues a Send_Distribution protocol boundary verb for SNA/DS (see Table 10-27).

As a result, the SNA/FS server encodes a new server object and SNA/DS builds and sends a message unit back to node A.

```
 _________________
|                 |
|EP_              |            ENTRY POINT C
|CHANGE_   |      |
|MGMT      |      |
|          |      |
|__________|______|
|          |      |
|FILE_     |      |
|SERVICES_|       |
|SUPPORT   |      |
|          |      |
|__________|______|_________________________ _______________
|          |                                |               |
|                                         |               |     /----------\
|              .............ÿ...............|...............|..ÿ..|....      |
|                                         |    SNA/FS     |     | Storage  |
|                 SNA/DS                  |    Server     |=====|          |
|                               ...........|...............|....|....      |
|                               .         |               |     \__________/
|                               .         |               |
|                               .         |               |
|_______________________________________|_______________|
                                .
                                ........ÿ........  (MU)
```

Figure 10-48. Failing Non-Destructive Send_and_Install (Part 4 of 6). EP_CHANGE_MGMT at C invokes FILE_SERVICES_SUPPORT which issues Send_Distribution.

The parameters supplied on the SNA/DS Send_Distribution verb are:

Table 10-27. Send_Distribution Supplied Parameters

Parameter Name	Parameter Value	Parameter Meaning
DISTRIBUTION_ID		
ORIGIN_DSU	NET1.C	DSU name of dist origin
ORIGIN_AGENT	X'23F0F0F0'	Indicates SNA/MS
ORIGIN_SEQNO	3456	Sequence number
AGENT_CORREL	Agent correlation value	See *SNA Formats*
DESTINATION	NET1.A	DSU name of recipient
DEST_AGENT	X'23F0F0F0'	Indicates SNA/MS
AGENT_OBJECT	CP-MSU consisting of: FS Action Summary GDS variable	See *SNA Formats*
SERVICE_PARMS		
PRIORITY	DATA8	Use at least this priority
PROTECTION	LEVEL2	Safe Store must be used
CAPACITY	16MEG	Handle at least 16M object
SECURITY	LEVEL2	Security required
REPORTING_REQUESTED		
EXCEPTION	YES	Request Exception Report
INTEGRITY	HIGH	Use confirmation protocols
REPORT_TO_DSU	NET1.A	DSU to send dist reports to
SERVER	X'24F0F0F0'	Indicates SNA/File Services
SPECIFIC_SERVER_INFO		As defined by SNA/FS
ENCODER_INSTRUCTIO	ENCODE_ONLY	
	ABEND	
	ONLY_IF_EXCEPTIONS	
DECODER_INSTRUCTIO	DECODE_ONLY	
	ABEND	
	ONLY_IF_EXCEPTIONS	
D_O_CLASS	X'10200000'	
D_O_GLOBAL_NAME	TO_BE_STORED_NAME (MCUST, 9135, NA, NET1, C, SYSTEM, V3)	
SNA_CONDITION_REPO	T	
SNA_REPORT_COD	X'084C0002'	
SUPPLEMENTAL_R	PORT	
	TARGET_INSTRUCT	ON
	(CREATE/LOAD, EXECUTING, BACKOUT, DETAILED)	

Response message unit is received at A:

SNA/DS at A receives the message unit and the SNA/FS server decodes the server object. Then SNA/DS invokes FILE_SERVICES_SUPPORT, which issues the Receive_Distribution SNA/DS protocol boundary verb and is returned parameters (see Table 10-28).

FILE_SERVICES_SUPPORT then invokes CHANGE_MGMT_NETOP, passing it the CP-MSU, source destination (NET1.C), distribution time, server parameters and agent unit-of-work correlator.

```
                 ________________
                |                |
                |CHANGE_         |            FOCAL POINT A
                |MGMT_           |
                |NETOP    "      |
                |        |       |
                |________|_______|
                |        |       |
                |FILE_   |       |
                |SERVICES_|      |
                |SUPPORT |       |
                |        |       |
                |________|_______|_________________________ _____________
                |        |                                |             |
                |        |                                |             |     /----------\
                |           ...............................|.............|....|....      |
                |                                         |    SNA/FS   |     | Storage  |
                |                SNA/DS                   |    Server   |=====|          |
                |                                 .....ÿ.....|.............|..ÿ..|....      |
                |                                 .       |             |     \__________/
                |                                 .       |             |
                |                                 .       |             |
                |_________________________________________|_____________|
                                                  .
                         (MU)   .........ÿ.........
```

Figure 10-49. Failing Non-Destructive Send_and_Install (Part 5 of 6). FILE_SERVICES_SUPPORT at A issues Receive_Distribution and is returned parameters. It then invokes CHANGE_MGMT_NETOP.

The parameters returned on the SNA/DS Receive_Distribution verb are:

Table 10-28. Receive_Distribution Returned Parameters

Parameter Name	Parameter Value	Parameter Meaning
DISTRIBUTION_ID		
ORIGIN_DSU	NET1.C	DSU name of dist origin
ORIGIN_AGENT	X'23F0F0F0'	Indicates SNA/MS
ORIGIN_SEQNO	3456	Sequence number
AGENT_CORREL	Agent correlation value	See *SNA Formats*
AGENT_OBJECT	CP-MSU consisting of: FS Action Summary GDS variable	See *SNA Formats*
REPORTING_REQUESTED		
EXCEPTION	YES	Request Exception Report
REPORT_TO_DSU	NET1.A	DSU to send dist reports to
DISTRIBUTION_TIME	hh.mm.ss.hh.GMT	GMT time at which distribution originated
SPECIFIC_SERVER_REPORT		As defined by SNA/FS
D_O_CLASS	X'10200000'	
D_O_GLOBAL_NAME	TO_BE_STORED_NAME (MCUST, 9135, NA, NET1, C, SYSTEM,	

```
|                       |     V3)                |                       |
|     SNA_CONDITION_REPO|T                       |                       |
|                       |                        |                       |
|         SNA_REPORT_COD|                        |                       |
|                       |         X'084C0002'    |                       |
|         SUPPLEMENTAL_R|PORT                    |                       |
|                       |         TARGET_INSTRUCT|ON                     |
|                       |         (CREATE/LOAD,  |                       |
|                       |         EXECUTING,     |                       |
|                       |         BACKOUT,       |                       |
|                       |         DETAILED)      |                       |
|_______________________|________________________|_______________________|
```

Network operator at A is issued response:

Based on the error identified in the FS_Action_Summary of the reply CP-MSU, CHANGE_MGMT_NETOP at A builds and issues, to the network operator, a Reply_To_Send protocol boundary verb for File Services defined in Appendix B, "Management Services Protocol Boundary Verbs" in topic B.0 (see Table 10-29). Note that the Correlator value is the same as that originally provided by the network operator on the Send_and_Install verb.

```
                Network Operator

                       "
                 _______|_______
                |       |       |
                |CHANGE_ |       |            FOCAL POINT A
                |MGMT_           |
                |NETOP           |
                |                |
                |________________|
                |                |
                |FILE_           |
                |SERVICES_       |
                |SUPPORT         |
                |                |
                |________________|_________________________ _____________
                |                                          |             |
                |                                          |             |     /----------\
                |                                          |             |     |          |
                |                                          |    SNA/FS   |     | Storage  |
                |                 SNA/DS                   |    Server   |=====|          |
                |                                          |             |     |          |
                |                                          |             |     \__________/
                |                                          |             |
                |                                          |             |
                |__________________________________________|_____________|
```

Figure 10-50. Failing Non-Destructive Send_and_Install (Part 6 of 6). CHANGE_MGMT_NETOP at A issues Reply_to_Send verb to network operator.

The parameters returned on the Reply_To_Send verb to the network operator are:

Table 10-29. Reply_to_Send Returned Parameters

Parameter Name	Parameter Value	Parameter Meaning
CORRELATOR	X'497'	Correlates report to initial request
TIME_STAMP	Date/Time of Installation	From the SNA/DS Distribution_Time parameter
TARGET_LOCATION	NET1.C	Target of attempted send
FS_ACTION_SUMMARY	NONE_SUCCESSFUL ALL_BACKED_OUT ABEND_NOT_APPLICABL CONTINUE_NOT_ATTEMPED	As defined by SNA/FS
SNA_CONDITION_REPORT		
SNA_REPORT_CODE	X'084C0002'	Create/Load Allocation Exception code

```
|                       |                       |                       |
|     REPORTED_ON_TOKEN_|TRINGMCUST, 9135, NA,  |     SNA/FS global     |
|                       |     NET1, C, SYSTEM,  |     name of           |
|                       |     V3                |     Create/Load error |
|                       |                       |     file              |
|_______________________|_______________________|_______________________|
```

2.10.4.13.3 Destructive SEND_AND_INSTALL

Consider the three nodes in a SNA/DS network depicted below. Assume that all three nodes happen to be in the same network (NET1).

```
   Network Operator

        |
        |
  _____|_________                                  _____________
 |     |         |                                |             |
 |               |                                |             |  Remote Production Node
 |             __|________________________________|_ÿ   Entry   |
 |               |                                |     Point C |
 |               |                                |_____________|
 |               |
 |               |
 |               |
 |               |
 |               |
 |               |  Local Distribution Node
 |        Focal  |
 |     "  Point A |
 |     |         |
 |_____|_________|
        |
        |
     ___|_______
    |   |       |
    |           |  Local Development Node
    |    Entry  |
    |    Point B |
    |___________|
```

Figure 10-51. Sample Configuration

Given the configuration in Figure 10-51, in the example that follows, a microcode customizing data change file with SNA/FS global name MCUST.9135.NA.NET1.C.SYSTEM.V3 has been prepared at a development node (entry point B) and sent, in an unsolicited manner, to a distribution node (focal point A). This change file is the same version (V3) of the customizing data change file that could not be successfully stored in the previous example.

A network operator at A wishes to send and install the change file at a remote production node (entry point C), but knowing that there is insufficient storage space at the remote node he wishes to direct the SNA/FS server at C in the destruction of an existing version of the microcode customizing data change file. Specifically, he wishes to reclaim storage currently occupied by the oldest version of the same change file, that being the one having the lowest character value in token 7 (the version number) of its SNA/FS global identifier.

The network operator chooses the following installation options:

- Production installation
- Removability prohibited
- Pre-test required
- Post-test required
- Alteration of Active Components to be allowed

A discussion of these Install options can be found in "Change Control" in topic 1.6.2.

Network operator at A issues request:

In the architected model, the network operator then assigns a sequence number and issues the Send_and_Install Change Management request protocol boundary verb which is defined in Appendix B, "Management Services Protocol Boundary Verbs" in topic B.0 (see Table 10-30). However, a typical implementation will assign the sequence number automatically before issuing the verb.

```
                Network Operator
                        |
                 _______|_______
                |       |       |
                |CHANGE_        |        FOCAL POINT A
                |MGMT_          |
                |NETOP          |
                |               |
                |               |
                |               |
                |               |
                |FILE_          |
                |SERVICES_      |
                |SUPPORT        |
                |               |
                |               |
                |               |
                |               
```

CHANGE_MGMT_NETOP at A builds the request CP-MSU, identifies the target destination as NET1.C, chooses server parameters and assigns an agent unit-of-work correlator. It then passes this data to FILE_SERVICES_SUPPORT which issues a Send_Distribution protocol boundary verb for SNA/DS (see Table 10-31).

As a result, the SNA/FS server fetches the change file and encodes the server object. SNA/DS then builds and sends a message unit (MU) to remote node C.

```
 _______________
|               |
|CHANGE_        |            FOCAL POINT A
|MGMT_          |
|NETOP_   |     |
|         |     |
|_________|_____|
|         |     |
|FILE_    |     |
|SERVICES_|     |
|SUPPORT  |     |
|         |     |
|_________|_____|_______________________ _____________
|         |                              |             |
|                                      |             |     /----------\
|             ................ÿ.............|.............|..ÿ..|.....     |
|                                      |    SNA/FS    |     | Storage  |
|                 SNA/DS               |    Server    |=====|          |
|                                 ..........|.............|....|.....     |
|                                 .        |             |     \_________/
|                                 .        |             |
|                                 .        |             |
|_________________________________________|_____________|
                                  .
                                  .........ÿ........  (MU)
```

Figure 10-53. Destructive Send_and_Install (Part 2 of 6). CHANGE_MGMT_NETOP at A invokes FILE_SERVICES_SUPPORT which issues Send_Distribution.

The parameters supplied on the SNA/DS Send_Distribution verb are (Notice that the CREATE/LOAD_OR_REPLACE instruction is to be used by the SNA/FS target server, thus allowing a previous version of the change file to be overwritten):

Table 10-31. Send_Distribution Supplied Parameters

Parameter Name	Parameter Value	Parameter Meaning
DISTRIBUTION_ID		
ORIGIN_DSU	NET1.A	DSU name of dist origin
ORIGIN_AGENT	X'23F0F0F0'	Indicates SNA/MS
ORIGIN_SEQNO	7890	Sequence number
AGENT_CORREL	Agent correlation value	See *SNA Formats*
DESTINATION	NET1.C	DSU name of recipient
DEST_AGENT	X'23F0F0F0'	Indicates SNA/MS
AGENT_OBJECT	CP-MSU consisting of: Request Change Control Major Vector consisting of: Install subvector	See *SNA Formats*
SERVICE_PARMS		
PRIORITY	DATA8	Use at least this priority
PROTECTION	LEVEL2	Safe Store must be used
CAPACITY	16MEG	
SECURITY	LEVEL2	Handle at least 16M object
		Security required

REPORTING_REQUESTED		
EXCEPTION	YES	Request Exception Report
INTEGRITY	HIGH	Use confirmation protocols
REPORT_TO_DSU	NET1.A	DSU to send dist reports to
SERVER	X'24F0F0F0'	Indicates SNA/File Services
SPECIFIC_SERVER_INFO		As defined by SNA/FS
SOURCE_INSTRUCTION	FETCH	
	ABEND	
	ONLY_IF_EXCEPTIONS	
TARGET_INSTRUCTION	CREATE/LOAD_OR_REPLCE	
	EXECUTING	
	ABEND	
	DETAILED	
D_O_CLASS	X'10200000'	
D_O_GLOBAL_NAME	TO_BE_FETCHED_NAME (MCUST, 9135, NA, NET1, C, SYSTEM, V3)	
	DELETING_MATCH_FLAG (7, SELECT_LOWEST)	

Request message unit is received at C:

SNA/DS at C receives the message unit and the SNA/FS server decodes the server object. Since the server locates two previous versions (MCUST.9135.NA.NET1.C.SYSTEM.V1 and MCUST.9135.NA.NET1.C.SYSTEM.V2) of the change file identified in the object identifier, it replaces MCUST.9135.NA.NET1.C.SYSTEM.V1 based on the criteria specified in the server parameters. Then SNA/DS invokes FILE_SERVICES_SUPPORT, which issues the Receive_Distribution SNA/DS protocol boundary verb and is returned parameters (see Table 10-32).

FILE_SERVICES_SUPPORT then invokes EP_CHANGE_MGMT, passing it the CP-MSU, source destination (NET1.A), server parameters and agent unit-of-work correlator.

Figure 10-54. Destructive Send_and_Install (Part 3 of 6).
FILE_SERVICES_SUPPORT at C issues Receive_Distribution and is returned parameters. It then invokes EP_CHANGE_MGMT.

The parameters returned on the SNA/DS Receive_Distribution verb are:

Table 10-32. Receive_Distribution Returned Parameters

Parameter Name	Parameter Value	Parameter Meaning
DISTRIBUTION_ID		
ORIGIN_DSU	NET1.A	DSU name of dist origin
ORIGIN_AGENT	X'23F0F0F0'	Indicates SNA/MS
ORIGIN_SEQNO	7890	Sequence number
AGENT_CORREL	Agent correlation value	See *SNA Formats*
AGENT_OBJECT	CP-MSU consisting of: Request Change Control Major Vector consisting of: Install subvector	See *SNA Formats*
REPORTING_REQUESTED		
EXCEPTION	YES	Request Exception Report
REPORT_TO_DSU	NET1.A	DSU to send dist reports to
DISTRIBUTION_TIME	hh.mm.ss.hh.GMT	GMT time at which distribution originated
SPECIFIC_SERVER_REPORT		As defined by SNA/FS
SERVER_SUMMARY_REPRT	ALL_SUCCESSFUL	
TARGET_INSTRUCTIONCE	CREATE/LOAD_OR_REPL	
	EXECUTING	
	ABEND	
	DETAILED	
D_O_CLASS	X'10200000'	
D_O_LOCAL_NAME	LOCAL.NAME	
D_O_GLOBAL_NAME	STORED_NAME (MCUST, 9135, NA, NET1, C, SYSTEM, V3)	
	DELETED_NAME (MCUST, 9135, NA, NET1, C, SYSTEM, V1)	

Response message unit is built at C and sent to A:

EP_CHANGE_MGMT at C parses the CP-MSU, extracting the Request Change Control major vector. Based on the Install subvector encodings, a pre-test is performed successfully on the change file and it is installed in production non-removably. After installation, a post-test is successfully performed and the component altered by the change file is applied to the running code. To report installation, EP_CHANGE_MGMT builds a reply CP-MSU, identifies the target destination as NET1.A, chooses server parameters and specifies the agent unit-of-work correlator. Finally, it invokes FILE_SERVICES_SUPPORT which issues a Send_Distribution protocol boundary verb for SNA/DS (see Table 10-33).

As a result, the SNA/FS server encodes a new server object and SNA/DS builds and sends a message unit back to node A.

```
                 ________________
                |                |
                |EP_             |            ENTRY POINT C
                |CHANGE_   |     |
                |MGMT      |     |
                |          |     |
```

```
                |________|______|
                |        |      |
                |FILE_   |      |
                |SERVICES_|     |
                |SUPPORT |      |
                |        |      |
                |________|______|_______________________ _____________
                |        |                              |             |
                |                                  |             |    /----------\
                |             .............ÿ..............|.............|..ÿ..|....      |
                |                                  |    SNA/FS    |     | Storage  |
                |                SNA/DS            |    Server    |=====|          |
                |                            ...........|.............|....|....      |
                |                            .          |             |    \__________/
                |                            .          |             |
                |                            .          |             |
                |____________________________________|_____________|
                                             .
                                             ........ÿ........  (MU)
```

Figure 10-55. Destructive Send_and_Install (Part 4 of 6). EP_CHANGE_MGMT at C invokes FILE_SERVICES_SUPPORT which issues Send_Distribution.

The parameters supplied on the SNA/DS Send_Distribution verb are:

Table 10-33. Send_Distribution Supplied Parameters

Parameter Name	Parameter Value	Parameter Meaning
DISTRIBUTION_ID		
ORIGIN_DSU	NET1.C	DSU name of dist origin
ORIGIN_AGENT	X'23F0F0F0'	Indicates SNA/MS
ORIGIN_SEQNO	4321	Sequence number
AGENT_CORREL	Agent correlation value	See *SNA Formats*
DESTINATION	NET1.A	DSU name of recipient
DEST_AGENT	X'23F0F0F0'	Indicates SNA/MS
AGENT_OBJECT	CP-MSU consisting of: Change Control Major Vector consisting of: Date/Time subvector Reporting Installatio subvector Reported Change Name subvector FS Action Summary GDS variable	See *SNA Formats*
SERVICE_PARMS		
PRIORITY	DATA8	Use at least this priority
PROTECTION	LEVEL2	Safe Store must be used
CAPACITY	16MEG	
SECURITY	LEVEL2	Handle at least 16M object Security required
REPORTING_REQUESTED		
EXCEPTION	YES	Request Exception Report
INTEGRITY	HIGH	Use confirmation protocols
REPORT_TO_DSU	NET1.A	DSU to send dist

		reports to
SERVER	X'24F0F0F0'	Indicates SNA/File Services
SPECIFIC_SERVER_INFO		As defined by SNA/FS
ENCODER_INSTRUCTIO	ENCODE_ONLY	
	ABEND	
	ONLY_IF_EXCEPTIONS	
DECODER_INSTRUCTIO	DECODE_ONLY	
	ABEND	
	ONLY_IF_EXCEPTIONS	
D_O_CLASS	X'10200000'	
D_O_GLOBAL_NAME	STORED_NAME (MCUST, 9135, NA, NET1, C, SYSTEM, V3)	
	DELETED_NAME (MCUST, 9135, NA, NET1, C, SYSTEM, V1)	

Response message unit is received at A:

SNA/DS at A receives the message unit and the SNA/FS server decodes the server object. Then SNA/DS invokes FILE_SERVICES_SUPPORT, which issues the Receive_Distribution SNA/DS protocol boundary verb and is returned parameters (see Table 10-34).

FILE_SERVICES_SUPPORT then invokes CHANGE_MGMT_NETOP, passing it the CP-MSU, source destination (NET1.C), server parameters and agent unit-of-work correlator.

```
                 ________________
                |                |
                |CHANGE_         |            FOCAL POINT A
                |MGMT_           |
                |NETOP    "      |
                |        |       |
                |________|_______|
                |        |       |
                |FILE_   |       |
                |SERVICES_|      |
                |SUPPORT |       |
                |        |       |
                |________|_______|_________________________ _____________
                |        |                                |             |
                |        |                                |             |     /----------\
                |           ...............................|.............|....|....      |
                |                                         |    SNA/FS   |     | Storage  |
                |                SNA/DS                   |    Server   |=====|          |
                |                                .....ÿ.....|.............|..ÿ..|....      |
                |                                .          |             |     \_________/
                |                                .          |             |
                |                                .          |             |
                |________________________________________|_____________|
                                                 .
                          (MU)   .........ÿ.........
```

Figure 10-56. Destructive Send_and_Install (Part 5 of 6).
FILE_SERVICES_SUPPORT at A issues Receive_Distribution and is returned parameters. It then invokes CHANGE_MGMT_NETOP.

The parameters returned on the SNA/DS Receive_Distribution verb are:

Table 10-34. Receive_Distribution Returned Parameters

Parameter Name	Parameter Value	Parameter Meaning
DISTRIBUTION_ID		
ORIGIN_DSU	NET1.C	DSU name of dist origin
ORIGIN_AGENT	X'23F0F0F0'	Indicates SNA/MS

ORIGIN_SEQNO	4321	Sequence number
AGENT_CORREL	Agent correlation value	See *SNA Formats*
AGENT_OBJECT	CP-MSU consisting of: Change Control Major Vector consisting of: Date/Time subvector Reporting Installatio subvector Reported Change Name subvector FS Action Summary GDS variable	See *SNA Formats*
REPORTING_REQUESTED		
EXCEPTION	YES	Request Exception Report
REPORT_TO_DSU	NET1.A	DSU to send dist reports to
DISTRIBUTION_TIME	hh.mm.ss.hh.GMT	GMT time at which distribution originated
SPECIFIC_SERVER_REPORT		As defined by SNA/FS
DECODER_INSTRUCTIO	DECODE_ONLY ABEND ONLY_IF_EXCEPTIONS	
D_O_CLASS	X'10200000'	
D_O_GLOBAL_NAME	STORED_NAME (MCUST, 9135, NA, NET1, C, SYSTEM, V3) DELETED_NAME (MCUST, 9135, NA, NET1, C, SYSTEM, V1)	

Network operator at A is issued response:

After parsing the reply CP-MSU and extracting data, CHANGE_MGMT_NETOP at A builds and issues, to the network operator, the Reporting_Installation protocol boundary verb for Change Management defined in Appendix B, "Management Services Protocol Boundary Verbs" in topic B.0 (see Table 10-35). Note that the Correlator value is the same as that originally provided by the network operator on the Send_and_Install verb.

```
                Network Operator

                        "
                 _______|_______
                |       |       |
                |CHANGE_ |       |            FOCAL POINT A
                |MGMT_           |
                |NETOP           |
                |                |
                |________________|
                |                |
                |FILE_           |
                |SERVICES_       |
                |SUPPORT         |
                |                |
                |________________|___________________________ _____________
                |                                           |             |
                |                                           |             |     /----------\
                |                                           |             |     |          |
                |                                           |   SNA/FS    |     | Storage  |
                |                 SNA/DS                    |   Server    |=====|          |
```

```
              |                                      |             |     |         |
              |                                      |             |     \_________/
              |                                      |             |
              |                                      |             |
              |______________________________________|_____________|
```

Figure 10-57. Destructive Send_and_Install (Part 6 of 6).
CHANGE_MGMT_NETOP at A issues Reporting_Installation verb to network operator.

The parameters returned on the Reporting_Installation verb to the network operator are:

Table 10-35. Reporting_Installation Returned Parameters

Parameter Name	Parameter Value	Parameter Meaning
CORRELATOR	X'723'	Correlates report to initial request
TIME_STAMP	Date/Time of Installation	From Date/Time subvector in returned CP-MSU
TARGET_LOCATION	NET1.C	Location where file stored
STORED_NAME	MCUST, 9135, NA, NET1, C, SYSTEM, V3	SNA/FS Global Name of Stored File
DELETED_NAME	MCUST, 9135, NA, NET1, C, SYSTEM, V1	SNA/FS Global Name of Deleted File
FS_ACTION_SUMMARY	ALL_SUCCESSFUL NO_BACKOUT_ATTEMPTE ABEND_NOT_APPLICABL CONTINUE_NOT_ATTEMPED	As defined by SNA/FS
INSTALLATION_RESULTS		
INSTALLATION_STATU	SUCCESSFUL	Installation successful
WHEN_EFFECTIVE	IN_USE	Activation not required
REPORTED_CHANGE_NAE_LIST		
CHANGE_FILE_NAE	MCUST, 9135, NA, NET1, C, SYSTEM, V3	Change file(s) installed
PRE_TEST_STATUS	SUCCESSFUL	Test was successful
POST_TEST_STATUS	SUCCESSFUL	Test was successful
AUTOMATIC_REMOVAL_RESUTS		
AUTOMATIC_REMOVAL_TATUS	NOT_ATTEMPTED	Removal not attempted
WHEN_EFFECTIVE	NOT_APPLICABLE	Effective time not applicable
AUTOMATIC_ACCEPTANCE_STATUS	NOT_ATTEMPTED	Acceptance not attempted
REMOVABILITY_STATUS	INSTALLED_NONREMOVALY	Change file may not be removed
ACTIVATION_USE_STATUS	PRODUCTION	Change file installed in production

---

2.10.4.13.4 Failing INSTALL

Consider the two nodes in a SNA/DS network depicted below. Assume that both nodes happen to be in the same network (NET1).

```
Network Operator

      |
      |
 _____|_________                                 _____________
|     |         |                               |             |
|               |                               |             |  Remote Production Node
|             __|_______________________________|_ÿ   Entry   |
|               |                               |     Point C |
|               |                               |_____________|
|               |
|               |
|               |
|               |
|               |  Local Distribution Node
|        Focal  |
|        Point A |
|               |
|_______________|
```

Figure 10-58. Sample Configuration

Given the configuration in Figure 10-58, in the example that follows, a microcode customizing data change file specifying system parameters with SNA/FS global name MCUST.9135.NA.NET1.C.LIB1.V2 and another microcode customizing data change file specifying printer parameters with SNA/FS global name MCUST.9135.NA.NET1.C.LIB4.V1 have been stored at, but not yet installed on, a remote production node (entry point C).

A network operator at A wishes to install both change files, as corequisites, at C. He chooses the following installation options:

- Trial installation
- Removability required
- Pre-test required
- Automatic Removal if test failure or installation failure
- Post-test not to be performed
- Automatic Acceptance prohibited
- Alteration of Active Components to be allowed

A discussion of these Install options can be found in "Change Control" in topic 1.6.2.

Network operator at A issues request:

In the architected model, the network operator then assigns a sequence number and issues the Install Change Management request protocol boundary verb which is defined in Appendix B, "Management Services Protocol Boundary Verbs" in topic B.0 (see Table 10-36). However, a typical implementation will assign the sequence number automatically before issuing the verb.

```
                Network Operator
                        |
               ________|________
              |        |        |
              |CHANGE_          |        FOCAL POINT A
              |MGMT_            |
              |NETOP            |
              |                 |
              |_________________|
              |                 |
              |FILE_            |
              |SERVICES_        |
              |SUPPORT          |
              |                 |
              |_________________|_________________________ _____________
              |                                          |             |
              |                                          |             |     /----------\
              |                                          |             |     |          |
              |                                          |    SNA/FS   |     | Storage  |
              |                 SNA/DS                   |    Server   |=====|          |
              |                                          |             |     |          |
              |                                          |             |     \__________/
```

```
              |                                        |             |
              |                                        |             |
              |________________________________________|_____________|
```

Figure 10-59. Failing Install (Part 1 of 6). Network operator at A issues Install verb to CHANGE_MGMT_NETOP.

The parameters supplied on the Install verb by the network operator are:

Table 10-36. Install Supplied Parameters

Parameter Name	Parameter Value	Parameter Meaning
CORRELATOR	X'42'	Correlation value for this unit of work
STORED_NAME	MCUST, 9135, NA, NET1, C, LIB1, V2	SNA/FS Global Name of stored change file
COREQUISITE_CHANGE_NAM_LIST CHANGE_FILE_NAME	MCUST, 9135, NA, NET1, C, LIB4, V1	SNA/FS Global Name of corequisite change file(s)
TARGET_LIST TARGET_LOCATION	NET1.C	Location(s) to send change file
REMOVABILITY	YES	Install file so it can be processed by subsequent Remove protocol boundary verb
AUTOMATIC_REMOVAL	YES	Remove Change File if installation or test fails
PRE_TEST	YES	Perform pre-test
POST_TEST	NO	Do not perform post-test
AUTOMATIC_ACCEPTANCE	NO	Do not automatically accept change file if installation and test succeed
ACTIVATION_USE	TRIAL	Components altered by installation will be accessed only by trial activation
ALTER_ACTIVE_COMPONENT	ALLOWED	Components altered by installation may be applied to running code

Request message unit is built at A and sent to C:

CHANGE_MGMT_NETOP at A builds the request CP-MSU, identifies the target destination as NET1.C, chooses server parameters and assigns an agent unit-of-work correlator. It then passes this data to FILE_SERVICES_SUPPORT which issues a Send_Distribution protocol boundary verb for SNA/DS (see Table 10-37).

As a result, the SNA/FS server encodes the server object. SNA/DS then builds and sends a message unit (MU) to remote node C.

```
              _________________
              |                |
              |CHANGE_         |            FOCAL POINT A
              |MGMT_           |
              |NETOP_   |      |
              |         |      |
              |_________|______|
              |         |      |
              |FILE_    |      |
              |SERVICES_|      |
              |SUPPORT  |      |
              |         |      |
              |_________|______|_________________________ _____________
              |         |                                |             |
              |                                          |             |     /----------\
```

```
|            ................ÿ.............|.............|..ÿ..|.....     |
|                                          |    SNA/FS   |     | Storage  |
|                SNA/DS                    |    Server   |=====|          |
|                                 ..........|.............|....|.....     |
|                                 .        |             |     \_________/
|                                 .        |             |
|                                 .        |             |
|_________________________________________|_____________|
                                  .
                                  .........ÿ........  (MU)
```

Figure 10-60. Failing Install (Part 2 of 6). CHANGE_MGMT_NETOP at A invokes FILE_SERVICES_SUPPORT which issues Send_Distribution.

The parameters supplied on the SNA/DS Send_Distribution verb are:

Table 10-37. Send_Distribution Supplied Parameters

Parameter Name	Parameter Value	Parameter Meaning
DISTRIBUTION_ID		
ORIGIN_DSU	NET1.A	DSU name of dist origin
ORIGIN_AGENT	X'23F0F0F0'	Indicates SNA/MS
ORIGIN_SEQNO	8765	Sequence number
AGENT_CORREL	Agent correlation value	See *SNA Formats*
DESTINATION	NET1.C	DSU name of recipient
DEST_AGENT	X'23F0F0F0'	Indicates SNA/MS
AGENT_OBJECT	CP-MSU consisting of: Request Change Control Major Vector consisting of: Install subvector Corequisite Change subvector	See *SNA Formats*
SERVICE_PARMS		
PRIORITY	DATA8	Use at least this priority
PROTECTION	LEVEL2	Safe Store must be used
CAPACITY	16MEG	
SECURITY	LEVEL2	Handle at least 16M object Security required
REPORTING_REQUESTED		
EXCEPTION	YES	Request Exception Report
INTEGRITY	HIGH	Use confirmation protocols
REPORT_TO_DSU	NET1.A	DSU to send dist reports to
SERVER	X'24F0F0F0'	Indicates SNA/File Services
SPECIFIC_SERVER_INFO		As defined by SNA/FS
ENCODER_INSTRUCTIO	ENCODE_ONLY ABEND ONLY_IF_EXCEPTIONS	
DECODER_INSTRUCTIO	DECODE_ONLY ABEND	

```
|                      |     ONLY_IF_EXCEPTIONS |                      |
|                      |                        |                      |
|     D_O_CLASS        |     X'10200000'        |                      |
|                      |                        |                      |
|     D_O_GLOBAL_NAME  |     STORED_NAME        |                      |
|                      |     (MCUST, 9135, NA,  |                      |
|                      |     NET1, C, LIB1, V2) |                      |
|______________________|________________________|______________________|
```

Request message unit is received at C:

SNA/DS at C receives the message unit and the SNA/FS server decodes the server object. Then SNA/DS invokes FILE_SERVICES_SUPPORT, which issues the Receive_Distribution SNA/DS protocol boundary verb and is returned parameters (see Table 10-38).

FILE_SERVICES_SUPPORT then invokes EP_CHANGE_MGMT, passing it the CP-MSU, source destination (NET1.A), server parameters and agent unit-of-work correlator.

```
              _________________
             |                 |
             |EP_              |            ENTRY POINT C
             |CHANGE_          |
             |MGMT     "       |
             |         |       |
             |_________|_______|
             |         |       |
             |FILE_    |       |
             |SERVICES_|       |
             |SUPPORT  |       |
             |         |       |
             |_________|_______|_________________________ _____________
             |         |                                |             |
             |         |                                |             |     /----------\
             |               ..........................|.............|....|....      |
             |                                          |    SNA/FS   |    |  Storage |
             |                 SNA/DS                   |    Server   |=====|          |
             |                             .......ÿ......|.............|..ÿ..|....      |
             |                             .            |             |     \__________/
             |                             .            |             |
             |                             .            |             |
             |__________________________________________|_____________|
                                           .
                      (MU)  ........ÿ........
```

Figure 10-61. Failing Install (Part 3 of 6). FILE_SERVICES_SUPPORT at C issues Receive_Distribution and is returned parameters. It then invokes EP_CHANGE_MGMT.

The parameters returned on the SNA/DS Receive_Distribution verb are:

Table 10-38. Receive_Distribution Returned Parameters

Parameter Name	Parameter Value	Parameter Meaning
DISTRIBUTION_ID		
ORIGIN_DSU	NET1.A	DSU name of dist origin
ORIGIN_AGENT	X'23F0F0F0'	Indicates SNA/MS
ORIGIN_SEQNO	8765	Sequence number
AGENT_CORREL	Agent correlation value	See *SNA Formats*
AGENT_OBJECT	CP-MSU consisting of: Request Change Control Major Vector consisting of: Install subvector Corequisite Change subvector	See *SNA Formats*
REPORTING_REQUESTED		

EXCEPTION	YES	Request Exception Report
REPORT_TO_DSU	NET1.A	DSU to send dist reports to
DISTRIBUTION_TIME	hh.mm.ss.hh.GMT	GMT time at which distribution originated
SPECIFIC_SERVER_REPORT		As defined by SNA/FS
DECODER_INSTRUCTIO	DECODE_ONLY	
	ABEND	
	ONLY_IF_EXCEPTIONS	
D_O_CLASS	X'10200000'	
D_O_GLOBAL_NAME	STORED_NAME (MCUST, 9135, NA, NET1, C, LIB1, V2)	

Response message unit is built at C and sent to A:

EP_CHANGE_MGMT at C parses the CP-MSU, extracting the Request Change Control major vector. A new microcode change file with SNA/FS global name MCODE.9135.NA.FUNCTEC.A38069.CONTROL required by the new customizing data files, is discovered to be missing. This requirement is described in the coverletter for the maintenance but was missed by the network operator.

To report the error, EP_CHANGE_MGMT builds a reply CP-MSU that includes an SNA Condition Report as well as a Change Management Reply major vector containing installation status, file identification and additional error related detailed data ((5201,01), (52aa,02) (52ff,01)). EP_CHANGE_MGMT identifies the report target destination as NET1.A and specifies the agent unit-of-work correlator (note that no SNA/FS server parameters are specified in this case). Finally, it invokes FILE_SERVICES_SUPPORT which issues a Send_Distribution protocol boundary verb for SNA/DS (see Table 10-39).

As a result, SNA/DS builds and sends a message unit back to node A.

Figure 10-62. Failing Install (Part 4 of 6). EP_CHANGE_MGMT at C invokes FILE_SERVICES_SUPPORT which issues Send_Distribution.

The parameters supplied on the SNA/DS Send_Distribution verb are:

Table 10-39. Send_Distribution Supplied Parameters

Parameter Name	Parameter Value	Parameter Meaning
DISTRIBUTION_ID		
ORIGIN_DSU	NET1.C	DSU name of dist origin

ORIGIN_AGENT	X'23F0F0F0'	Indicates SNA/MS
ORIGIN_SEQNO	2109	Sequence number
AGENT_CORREL	Agent correlation value	See *SNA Formats*
DESTINATION	NET1.A	DSU name of recipient
DEST_AGENT	X'23F0F0F0'	Indicates SNA/MS
AGENT_OBJECT	CP-MSU consisting of: Change Control Major Vector consisting of: Date/Time subvector Reporting Installatio subvector Reported Change Name subvector for first change file Reported Change Name subvector for coreq change file Detailed Data subvector SNA Condition Report GDS Variable identifying missing change file	See *SNA Formats*
SERVICE_PARMS		
PRIORITY	DATA8	Use at least this priority
PROTECTION	LEVEL2	Safe Store must be used
CAPACITY	16MEG	Handle at least 16M object
SECURITY	LEVEL2	Security required
REPORTING_REQUESTED		
EXCEPTION	YES	Request Exception Report
INTEGRITY	HIGH	Use confirmation protocols
REPORT_TO_DSU	NET1.A	DSU to send dist reports to

Response message unit is received at A:

SNA/DS at A receives the message unit and invokes FILE_SERVICES_SUPPORT, which issues the Receive_Distribution SNA/DS protocol boundary verb and is returned parameters (see Table 10-40).

FILE_SERVICES_SUPPORT then invokes CHANGE_MGMT_NETOP, passing it the CP-MSU, source destination (NET1.C) and agent unit-of-work correlator.

```
                 ________________
                |                |
                |CHANGE_         |            FOCAL POINT A
                |MGMT_           |
```

```
                |NETOP    "       |
                |        |       |
                |________|_______|
                |        |       |
                |FILE_    |       |
                |SERVICES_|       |
                |SUPPORT  |       |
                |        |       |
                |________|_______|_______________________ _____________
                |         |                              |             |
                |         |                              |             |     /----------\
                |            ....................         |             |     |          |
                |                              .         |    SNA/FS   |     | Storage  |
                |                 SNA/DS       .         |    Server   |=====|          |
                |                              .         |             |     |          |
                |                              .         |             |     \__________/
                |                              .         |             |
                |                              .         |             |
                |______________________________________|_____________|
                                               .
                         (MU)   .........ÿ........
```

Figure 10-63. Failing Install (Part 5 of 6). FILE_SERVICES_SUPPORT at A issues Receive_Distribution and is returned parameters. It then invokes CHANGE_MGMT_NETOP.

The parameters returned on the SNA/DS Receive_Distribution verb are:

Table 10-40. Receive_Distribution Returned Parameters

Parameter Name	Parameter Value	Parameter Meaning
DISTRIBUTION_ID		
ORIGIN_DSU	NET1.C	DSU name of dist origin
ORIGIN_AGENT	X'23F0F0F0'	Indicates SNA/MS
ORIGIN_SEQNO	2109	Sequence number
AGENT_CORREL	Agent correlation value	See *SNA Formats*
AGENT_OBJECT	CP-MSU consisting of: Change Control Major Vector consisting of: Date/Time subvector Reporting Installatio subvector Reported Change Name subvector for first change file Reported Change Name subvector for coreq change file Detailed Data subvector SNA Condition Report GDS Variable identifying missing change file	See *SNA Formats*
REPORTING_REQUESTED		
EXCEPTION	YES	Request Exception Report
REPORT_TO_DSU	NET1.A	DSU to send dist reports to

DISTRIBUTION_TIME	hh.mm.ss.hh.GMT	GMT time at which distribution originated

Network operator at A is issued response:

After parsing the reply CP-MSU and extracting data, CHANGE_MGMT_NETOP at A builds and issues, to the network operator, the Reporting_Installation protocol boundary verb for Change Management defined in Appendix B, "Management Services Protocol Boundary Verbs" in topic B.0 (see Table 10-41). Note that the Correlator value is the same as that originally provided by the network operator on the Install verb.

```
                Network Operator

                        "
                 ________|________
                |        |        |
                |CHANGE_ |        |            FOCAL POINT A
                |MGMT_            |
                |NETOP            |
                |                 |
                |_________________|
                |                 |
                |FILE_            |
                |SERVICES_        |
                |SUPPORT          |
                |                 |
                |_________________|___________________________ _____________
                |                                           |             |
                |                                           |             |     /----------\
                |                                           |             |     |          |
                |                                           |    SNA/FS   |     | Storage  |
                |                 SNA/DS                    |    Server   |=====|          |
                |                                           |             |     |          |
                |                                           |             |     \__________/
                |                                           |             |
                |                                           |             |
                |___________________________________________|_____________|
```

Figure 10-64. Failing Install (Part 6 of 6). CHANGE_MGMT_NETOP at A issues Reporting_Installation verb to network operator.

The parameters returned on the Reporting_Installation verb to the network operator are:

Table 10-41. Reporting_Installation Returned Parameters

Parameter Name	Parameter Value	Parameter Meaning
CORRELATOR	X'42'	Correlates report to initial request
TIME_STAMP	Date/Time of Installation	From Date/Time subvector in returned CP-MSU
TARGET_LOCATION	NET1.C	Location where file stored
INSTALLATION_RESULTS		
INSTALLATION_STATU	NOT_ATTEMPTED	Install not attempted
WHEN_EFFECTIVE	NOT_APPLICABLE	Time not applicable
REPORTED_CHANGE_NA	E_LIST	
CHANGE_FILE_NA	E MCUST, 9135, NA, NET1, C, LIB1, V2	Change file(s) being reported on
CHANGE_FILE_NA	E MCUST, 9135, NA, NET1, C, LIB4, V1	
PRE_TEST_STATUS	NOT_ATTEMPTED	Test not attempted
POST_TEST_STATUS	NOT_ATTEMPTED	Test not attempted
AUTOMATIC_REMOVAL_RESU	TS	
AUTOMATIC_REMOVAL_	TATUSNOT_ATTEMPTED	Removal not

		attempted
WHEN_EFFECTIVE	NOT_APPLICABLE	Effective time not applicable
AUTOMATIC_ACCEPTANCE_STATUS	NOT_ATTEMPTED	Acceptance not attempted
REMOVABILITY_STATUS	NOT_INSTALLED	Installation was not successful
ACTIVATION_USE_STATUS	NOT_INSTALLED	Installation was not successful
DETAILED_DATA	(5201,01), (52aa,02), (52ff,01)	Error codes unique to entry point implementation
SNA_CONDITION_REPORT		
SNA_REPORT_CODE	X'08380012'	Missing corequisite code
REPORTED_ON_TOKEN	STRINGMCODE, 9135, NA, FUNCTEC, A38069, CONTROL	SNA/FS global name of missing corequisite file

2.10.5 EP_COMMON_OPERATIONS_SERVICES Function Set

```
     ........................
     .                      .
     . MULTIPLE_DOMAIN_SUPPORT .
     . function set group      .
     ........................
          "      |      "
        (O)|   (Q)|   (R)|
           |      |      |
°°°°°°°°°°°|°°°°°°|°°°°°°|°°°°°°°°°°°
```

- Input:

  - Internal MS protocol boundary D (NMVT Received)

    The EP_COMMON_OPERATIONS_SERVICES function set group receives requests from the RECEIVE_REQUEST_SSCP_PU function set group to process incoming NMVTs. It is passed the NMVT and the name of the CP from which it was received. The details of this protocol boundary are described in "Protocol Boundary D--NMVT Received" in topic 2.9.1.6.

  - Internal MS protocol boundary Q (MDS-MU Received)

    The EP_COMMON_OPERATIONS_SERVICES function set group receives requests from the MULTIPLE_DOMAIN_SUPPORT function set group to process incoming MDS-MUs. It is passed the complete MDS-MU. The details of this protocol boundary are described in "Protocol Boundary Q--MDS-MU Received" in topic 2.9.1.12.

  - MS application program protocol boundary A-2

    The EP_COMMON_OPERATIONS_SERVICES function set group receives replies to earlier commands, as well as unsolicited messages to operators, from served network management application programs in its node. It is passed the one or more completed MS major vectors, and, in the case of replies, the correlator that it passed to the application program with the request. The details of this protocol boundary are described in Table 10-44 in topic 2.10.5.4.4.

- Output:

  - Internal MS protocol boundary A (Send NMVT)

    The EP_COMMON_OPERATIONS_SERVICES function set group requests the SEND_DATA_SSCP_PU function set group to send an NMVT RU on the SSCP-PU session with its controlling CP. The output data consists of the complete NMVT and the address of the control point to which it is to be sent. The details of this protocol boundary are described in "Protocol Boundary A--Send NMVT" in topic 2.9.1.3.

  - Internal MS protocol boundary E (Send NMVT Response)

    The EP_COMMON_OPERATIONS_SERVICES function set group requests the SEND_DATA_SSCP_PU function set group to send an NMVT response RU on an SSCP-PU session with a specified CP. The output data consists of the CP address and sense data. The details of this protocol boundary are described in "Protocol Boundary E--Send NMVT Response" in topic 2.9.1.7.

  - Internal MS protocol boundary O (Register Application program Name)

    An application program utilizing the MDS transport registers itself with the MULTIPLE_DOMAIN_SUPPORT function set group via this protocol boundary. The details of this protocol boundary are described in "Protocol Boundary O--Register Application Program Name" in topic 2.9.1.11.

    The process by which an application program registers itself with MULTIPLE_DOMAIN_SUPPORT is described in "Registering with MDS" in topic 2.9.2.9.1.

  - Internal MS protocol boundary R (Send MDS-MU)

    The EP_COMMON_OPERATIONS_SERVICES function set group requests the MULTIPLE_DOMAIN_SUPPORT function set group to send an MDS-MU. The output data is the complete MDS-MU. The details of this protocol boundary are described in "Protocol Boundary R--Send MDS-MU" in topic 2.9.1.13.

  - MS application program protocol boundary A-1

    The EP_COMMON_OPERATIONS_SERVICES function set group passes commands to served network management application programs in its node. For each command it passes a complete MS major vector, together with a correlator that the application program will retain to pass back with the reply. In order to shield served application programs from differences in transport, and from the introduction of new transports, this correlator is defined solely by EP_COMMON_OPERATIONS_SERVICES. The details of this protocol boundary are described in Table 10-43 in topic 2.10.5.4.4.

2.10.5.2 Prerequisite Function Sets

See "Role Requirements for Management Services Components" in topic 2.8.1 for information on the relationships between this function set and other function sets.

2.10.5.3 Overview of Subsets

```
 _____________________________________________________
|                                                     |
|          no optional subsets available for          |
|     EP_COMMON_OPERATIONS_SERVICES function set      |
|                                                     |
|_____________________________________________________|
|                                                     |
|      EP_COMMON_OPERATIONS_SERVICES base subset      |
|                                                     |
|_____________________________________________________|
```

Figure 10-66. Base and Optional Subsets of EP_COMMON_OPERATIONS_SERVICES Function Set

---

2.10.5.4 EP_COMMON_OPERATIONS_SERVICES Base Subset

Subtopics:

- 2.10.5.4.1 Functions Provided
- 2.10.5.4.2 Formats Supported
- 2.10.5.4.3 Electives
- 2.10.5.4.4 Implementation Requirements

2.10.5.4.1 Functions Provided

The EP_COMMON_OPERATIONS_SERVICES base subset provides the capability to support communication between network operators and served network management applicatons, as described in "Common Operations Services Function Set" in topic 1.7.2.

2.10.5.4.2 Formats Supported

The EP_COMMON_OPERATIONS_SERVICES base subset supports the receiving of an NMVT or MDS-MU containing one of four management services major vectors:

- Execute Command (X'8061')
- Analyze Status (X'8062')
- Query Resource Data (X'8063')
- Test Resource (X'8064')

The base subset passes each of these major vectors to a served network management application program, which is named in a Name List (X'06') subvector within the major vector.

The EP_COMMON_OPERATIONS_SERVICES base subset also supports receipt, from served network management application programs, of the combinations of management services major vectors shown in Table 10-42.

Table 10-42. Management Services Major Vectors Supported by EP_COMMON_OPERATIONS_SERVICES

Major Vectors	Number
Reply to Execute Command (X'0061')	1
Text Data (X'1300')	0-1
Structured Data (X'1307')	0-1

Transparent Coded Datastream (X'1309')	0-1
Reply to Analyze Status (X'0062')	1
Begin Data Parameters (X'130A')	1
Structured Data (X'1307')	0-n
End Parameter Data (X'130B')	1
Reply to Query Resource Data (X'0063')	1
Begin Data Parameters (X'130A')	1
Structured Data (X'1307')	1-n
End Parameter Data (X'130B')	1
Reply to Test Resource (X'0064')	1
Begin Data Parameters (X'130A')	1
Structured Data (X'1307')	0-n
End Parameter Data (X'130B')	1
Send Message to Operator (X'006F')	1
Text Data (X'1300')	0-1
Structured Data (X'1307')	0-1
Transparent Coded Datastream (X'1309')	0-1

The major vectors having keys beginning with X'13' are MS *parameter* major vectors; see "Parameter Major Vectors" in topic 1.7.2.1.3 for a discussion of MS major vectors of this type.

2.10.5.4.3 Electives

The only elective available to this subset is the reporting of requests that cannot be processed by sending sense data in either a negative response or reply.

2.10.5.4.4 Implementation Requirements

The requirements for implementing the EP_COMMON_OPERATIONS_SERVICES base subset are described by a model consisting of a subset of Management Services.

A Subset of Management Services:

° Receiving common operations services requests

Upon request by the RECEIVE_REQUEST_SSCP_PU function set group (via protocol boundary D), or by the MULTIPLE_DOMAIN_SUPPORT function set group (via protocol boundary Q), does the following:

- Parses the request for validity.

- If an error was detected, either requests the RECEIVE_REQUEST_SSCP_PU function set group (via protocol boundary E), to send a negative response, or builds an MDS-MU containing a negative reply and passes it to the MULTIPLE_DOMAIN_SUPPORT function set group (via protocol boundary R).

- If the request was in an NMVT and parsed correctly, requests the RECEIVE_REQUEST_SSCP_PU function set group (via protocol boundary E), to send a positive response.

- If the request parsed correctly, passes two items to the network management application program identified in the request: the major vector, and a correlator defined by EP_COMMON_OPERATIONS_SERVICES (via MS application program protocol boundary A-1).

° Sending common operations services data

Upon request by a served network management application program, does the following:

- Receives two or more completed management services major vectors from the application program. For a reply, the application program also passes to EP_COMMON_OPERATIONS_SERVICES the correlator that it received with the request. (Via MS application program protocol boundary A-2).

- For a reply, uses the correlator to retrieve the PRID or Agent

Unit-of-Work Correlator (X'1549') GDS variable saved from the request.

- Using the major vectors and the PRID or Agent Unit-of-Work Correlator (X'1549') GDS variable, builds a complete NMVT or MDS-MU.

- Passes the NMVT to the SEND_DATA_SSCP_PU function set group via protocol boundary A, or passes the MDS-MU to the MULTIPLE_DOMAIN_SUPPORT function set group via protocol boundary R.

Receiving Common Operations Services Requests:

° If the request came in an NMVT:

This process is started by the process described in "Receiving NMVTs" in topic 2.9.6.4.3 (RECEIVE_REQUEST_SSCP_PU function set group), when the latter has identified one of four major vector keys in a request NMVT:

- X'8061' (Execute Command)
- X'8062' (Analyze Status)
- X'8063' (Query Resource Data)
- X'8064' (Test Resource)

The first task of this process is to parse the common operations services major vector; in particular, it searches for the Destination Application program Name (X'50') subfield within the Name List (X'06') subvector. If it cannot locate this subfield, or if the network management application program identified in the subfield is not known to it, then this process starts the process described in "Sending NMVT Responses" in topic 2.9.6.4.3 to send the appropriate -RSP. The sense data that can be sent by PUMS at this time are detailed in Table 10-47 in topic 2.10.7.7 and Table 10-53 in topic 2.10.8.4. If no condition requiring a -RSP is found, then this process starts the process described in "Sending NMVT Responses" (via protocol boundary E) to send a +RSP,

Next this process creates a correlator to pass to the served application program when it passes it the major vector from the request. EP_COMMON_OPERATIONS_SERVICES retains an association between this correlator and the PRID from the request. The purpose of this private correlator, used only on the EP_COMMON_OPERATIONS_SERVICES application program protocol boundaries A-1 and A-2, is to shield served application programs from differences in, and changes to, the manner in which management services data is transported between nodes. Since knowledge of request/reply correlation is used by transport components as well as by the application programs using these components to communicate, correlators, such as the PRID, that actually flow between nodes tend to be transport-specific. By mapping the PRID to its own private correlator, EP_COMMON_OPERATIONS_SERVICES insures that the network management application programs it serves will not be affected by changes to the underlying management services transport.

This process now passes the major vector from the request, along with its private correlator, to the application program, via MS application program protocol boundary A-1. See Table 10-43 for details of this protocol boundary.

After passing the major vector and correlator to the designated application program, this process terminates.

° If the request came in an MDS-MU:

This process is started by the process described in "Routing MDS-MU Received to Local MS Application Programs" in topic 2.9.2.4.8 (MULTIPLE_DOMAIN_SUPPORT function set group), when the latter has identified EP_COMMON_OPERATIONS_SERVICES (in its own node) as the destination application program for an MDS-MU. The first task of this process is to parse the MDS-MU, and within it, the common operations services major vector. Within the major vector, it searches for the same element it did when the major vector was enveloped in an NMVT: the Destination Application program Name (X'50') subfield within the Name List (X'06') subvector. If it cannot locate this subfield, or if the network management application program identified in the subfield is not known to it, or if it detects an error in the MDS-MU itself, then this process constructs an MDS-MU to report the error. It then starts the process described in "Processing Send MDS-MU" in topic 2.9.2.4.8 to send the MDS-MU to the COMMON_OPERATIONS_SERVICES_NETOP application program that originated the request. The sense data that can be sent by EP_COMMON_OPERATIONS_SERVICES at this time are detailed in Table 10-53 in topic 2.10.8.4. The general format and construction of an application program-generated MDS-MU for a negative reply are discussed in "Building Negative Replies for MDS-MUs" in

topic 2.9.2.9.4.

As it did in the case of the NMVT, this process creates a private correlator to pass to the application program when it passes it the major vector. In the case of an MDS-MU the correlator is associated with the Agent Unit-of-Work Correlator (X'1549') GDS variable from the request. The routing information from the request MDS-MU is also retained, so that the reply can be sent to the requester.

This process now passes the major vector from the request, along with its private correlator, to the application program, via MS application program protocol boundary A-1. See Table 10-43 for details of this protocol boundary.

After passing the major vector and correlator to the designated application program, this process terminates.

Table 10-43. MS Application Protocol Boundary A-1

Origin	EP_COMMON_OPERATIONS_SERVICES
Destination	The served network management application identified in the common operations services request.
Data Content	1. A pointer to the common operations services major vector received in an NMVT or MDS-MU 2. A private token for request/reply correlation
Initiates	The network management applicatoin identified in the common operations services request.

Sending Common Operations Services Data:

This process is started by a served network management application program when the application program has common operations services data to send. The application program passes to EP_COMMON_OPERATIONS_SERVICES (via protocol boundary A-2), a number of management services major vectors, and, in the case of a reply, the correlator that it received with the request. See Table 10-44 for a description of MS application program protocol boundary A-2.

Table 10-42 in topic 2.10.5.4.2 summarizes the combinations of major vectors that a network management application program can pass to EP_COMMON_OPERATIONS_SERVICES. "Common Operations Services Function Set" in topic 1.7.2 describes more fully which major vectors are to be sent by an application program in reply to each of the common operations services commands, and which may be sent unsolicited. EP_COMMON_OPERATIONS_SERVICES, however, makes no attempt to enforce these rules; it envelopes and sends the major vectors it receives, without examining their contents.

EP_COMMON_OPERATIONS_SERVICES bases its next action on whether it is going to envelope the major vectors it has received in an NMVT, or a CP-MSU within an MDS-MU. For replies, the rule is to send the reply in the same format that the corresponding request was received. Thus if the private EP_COMMON_OPERATIONS_SERVICES correlator corresponds to a PRID, it builds an NMVT. If this correlator corresponds to an Agent Unit-of-Work Correlator (X'1549') GDS variable, it builds a CP-MSU and encloses it within an MDS-MU. Unsolicited common operations services data, i.e., a Send Message to Operator, is sent in an NMVT if a SSCP-PU session exists. Otherwise, the data is enclosed within a CP-MSU and sent in an MDS-MU.

° If the application program is building an NMVT:

For a reply, this process uses the correlator passed to it with the major vectors to retrieve the PRID from the original request.

Using the PRID that it has just retrieved, this process now constructs a complete NMVT enveloping the major vectors. If, however, the enveloped major vectors would result in the construction of an NMVT that exceeds the maximum allowable NMVT size for a solicited reply, multiple NMVTs are constructed. Before doing so, the MS major vector is expanded to include the NMVT Count (X'44') subvector in such a way that the subvector will be contained within the first NMVT. The process now constructs the individual NMVTs with each containing a physical slice of the total reply and each having the appropriate sequencing flags set in its NMVT header.

This process starts the process described in "Sending NMVTs" in topic 2.9.5.4.3 (SEND_DATA_SSCP_PU function set group) via protocol

boundary A, to send the NMVT. "Protocol Boundary A--Send NMVT" in topic 2.9.1.3 shows the items passed when starting that process. This process places the following values in these items:

Item 1: The NMVT it has constructed

Item 2: The address of the node's controlling control point (present only on replies)

This process starts the "Sending NMVTs" process once for each NMVT it has constructed. After doing so, it terminates.

° If the application program is building an MDS-MU:

For a reply, this process uses the correlator passed to it with the major vectors to retrieve the Agent Unit-of-Work Correlator (X'1549') GDS variable from the original request.

Using the Agent Unit-of-Work Correlator (X'1549') GDS variable that it has just retrieved, this process now constructs a complete MDS-MU. The destination NAU and application program for the MDS-MU are those from which the request MDS-MU originated. The major vectors from the application program are enveloped within a CP_MSU, which flows as the application program GDS variable in the MDS-MU.

This process starts the process described in "Processing Send MDS-MU" in topic 2.9.2.4.8 (MULTIPLE_DOMAIN_SUPPORT function set group) via protocol boundary R, to send the MDS-MU to the application program identified in the MDS Routing Information structure within the MDS-MU. As "Protocol Boundary R--Send MDS-MU" in topic 2.9.1.13 indicates, two items are passed when starting this process:

- A pointer to the MDS-MU

- An indication that default routing is to be used

After this process has started the "Processing Send MDS-MU" process, it terminates.

Table 10-44. MS Application Protocol Boundary A-2

Origin	A served network management application
Destination	EP_COMMON_OPERATIONS_SERVICES
Data Content	1. A pointer to a set of common operations services major vectors that the served application has built 2. The private token for request/reply correlation that was passed to the application with the request **Note:** Item 2 is passed only on replies.
Initiates	Process described in "Sending Common Operations Services Data"

2.10.5.5 Common Operations Services Commands, Replies, and Unsolicited Traffic

The architecture for common operations services provides for the transfer of four commands from a network operator to a specified network management application program, and the return of the corresponding replies from the application program to the operator. It also provides for the transfer of a message from an application program to a specified operator. The remainder of this section provides a brief description of how each of these five functions is implemented. Before proceeding, the reader should review "Parameter Major Vectors" in topic 1.7.2.1.3.

Subtopics:

- 2.10.5.5.1 Execute Command
- 2.10.5.5.2 Analyze Status
- 2.10.5.5.3 Query Resource Data
- 2.10.5.5.4 Test Resource
- 2.10.5.5.5 Send Message to Operator

2.10.5.5.1 Execute Command

° *Execute Command:*

This command is entered by a network operator to request that a command be sent; the command itself is not interpreted by the common operations services component in the control point. However, it is placed in either the Self-Defining Text Message (X'31') subvector or one of three parameter major vectors. In this case the Execute Command (X'8061') MS major vector is built, with the following subvectors:

- X'06' The Name List subvector, with a Destination Application program Name (X'50') subfield, is built to transport the operator-specified name of the destination application program to which the command is to be routed.

- X'31' The Self-Defining Text Message subvector is conditionally built as follows, to transport the command:

  -- Coded Character Set ID (X'02') subfield: always set to X'028001F4', i.e., Coded Graphic Character Set 00640-00500

  -- Text Message (X'30') subfield: this subfield contains, as text, the command to be transported.

If the above mentioned Self-Defining Text Message (X'31') subvector is not built, the MS major vector is followed by one of three parameter major vectors:

- Text Data (X'1300')

- Structured Data (X'1307')

- Transparent Coded Datastream (X'1309')

If any of these parameter major vectors is built, the Self-Defining Text Message (X'31') subvector is not built within the Execute Command (X'8061') MS major vector.

° *Reply to Execute Command:*

A positive reply to the Execute Command major vector consists of an empty Reply to Execute Command (X'0061') major vector, unless the reply spans multiple NMVTs in which case it contains the NMVT Count (X'44') subvector which is built as follows:

- Number of Replies (X'01') subfield: always set to X'0001'.

- Number of NMVTs (X'10') subfield: this subfield contains the number of NMVTs spanned by the reply.

- Maximum Size of NMVTs (X'11') subfield: this subfield contains the maximum size of the NMVTs that transport the reply.

In either case, the MS major vector is followed by one of three parameter major vectors:

- Text Data (X'1300')

- Structured Data (X'1307')

- Transparent Coded Datastream (X'1309')

A negative reply to the Execute Command major vector consists of a Reply to Execute Command major vector containing a Sense Data (X'7D') subvector indicating why the command failed. No parameter major vector is present in this case.

2.10.5.5.2 Analyze Status

° *Analyze Status:*

This command is entered by a network operator to request that information about one or more specified resources be gathered and analyzed. The reply to this command reports the joint state of all the specified resources. In this case the Analyze Status (X'8062')

major vector is built. This major vector contains a single subvector, the Name List (X'06') subvector, which is built as follows:

- The Destination Application program Name (X'50') subfield is always built, to carry the operator-specified name of the destination application program to which the command is to be routed.

- The Associated Resource Name List (X'01') subfield is always built.

  Note: This contents of this subfield are not interpreted either by the common operations services component in the control point or by EP_COMMON_OPERATIONS_SERVICES; its entries are specified by the network operator, and interpreted by the served application program to which EP_COMMON_OPERATIONS_SERVICES routes the request. There are three mutually incompatible ways in which the subfield is currently used:

  -- The entries in the list may identify individual discrete resources to which the command carried in the request is to be applied.

  -- The entries in the list may identify entities that delimit a *range* of resources to which the command is to be applied. In the case of a complex link, for example, the subfield might contain only the names of the resources at the two ends of the link, with the understanding that the request applies not only to these two specific resources, but also to all the other resources between them on the complex link.

  -- Some or all of the entries in the list may serve as nicknames that index lists of actual resource names known to the served application program that processes the request.

  Obviously the network operator initiating a common operations services request must understand in which of these three ways the served application program receiving the request will interpret the Associated Resource Name List subfield.

° *Reply to Analyze Status:*

A positive reply to the Analyze Status major vector consists of an empty Reply to Analyze Status (X'0062') major vector, unless the reply spans multiple NMVTs in which case it contains the NMVT Count (X'44') subvector which is built as follows:

- Number of Replies (X'01') subfield: always set to X'0001'.

- Number of NMVTs (X'10') subfield: this subfield contains the number of NMVTs spanned by the reply.

- Maximum Size of NMVTs (X'11') subfield: this subfield contains the maximum size of the NMVTs that transport the reply.

In either case, the MS major vector is followed by a list of parameter major vectors:

- One Begin Data Parameters (X'130A') major vector

- Zero or more Structured Data (X'1307') major vectors

- One End Parameter Data (X'130B') major vector.

Each Structured Data major vector present in the reply contains data for a single resource. The Hierarchy/Resource List (X'05') subvector within the Structured Data major vector contains a Resource Identifier (X'01') subfield that identifies the resource for which analysis has been completed.

A negative reply to the Analyze Status major vector consists of a Reply to Analyze Status major vector containing a Sense Data (X'7D') subvector indicating why the command failed. No parameter major vector is present in this case.

2.10.5.5.3 Query Resource Data

- *Query Resource Data:*

  This command is entered by a network operator to request that information be gathered and returned from one or more specified resources. In this case the Query Resource Data (X'8063') major vector is built, with the following subvector:

  - X'06' The Name List subvector is built in exactly the same way it was for the Analyze Status (X'8062') major vector.

- *Reply to Query Resource Data:* The replies to the Query Resource Data major vector have exactly the same structure as those to the Analyze Status major vector, except that a positive reply always includes at least one Structured Data (X'1307') major vector. Use of the NMVT Count (X'44') subvector for a single reply spanning multiple NMVTs applies here as well.

2.10.5.5.4 Test Resource

- *Test Resource:*

  This command is entered by a network operator to request that one or more specified resources be tested. The reply to this command reports the state of each resource. In this case the Test Resource (X'8064') major vector is built, with the following subvectors:

  - X'06' The Name List subvector is built in exactly the same way it was for the Analyze Status (X'8062') major vector.

  - X'81' The Test Setup Data subvector is built based on information provided by the operator.

- *Reply to Test Resource:*

  A positive reply to the Test Resource major vector consists of an Reply to Test Resource (X'0064') major vector, followed by a list of parameter major vectors:

  - One Begin Data Parameters (X'130A') major vector

  - Zero or more Structured Data (X'1307') major vectors

  - One End Parameter Data (X'130B') major vector.

  The Reply to Test Resource major vector contains a Test Result Data (X'81') subvector reporting the results of the test that was requested. Each Structured Data major vector present in the reply contains data for a single resource.

  If the reply spans multiple NMVTs, it contains the NMVT Count (X'44') subvector which is built as follows:

  - Number of Replies (X'01') subfield: always set to X'0001'.

  - Number of NMVTs (X'10') subfield: this subfield contains the number of NMVTs spanned by the reply.

  - Maximum Size of NMVTs (X'11') subfield: this subfield contains the maximum size of the NMVTs that transport the reply.

  A negative reply to the Test Resource major vector consists of a Reply to Test Resource major vector containing a Sense Data (X'7D') subvector

---

2.10.5.5.5 Send Message to Operator

- *Send Message to Operator:* This function is invoked by a network management application program, to send a message to a network operator. In this case a Send Message to Operator (X'006F') major vector is built; the included Name List (X'06') carries, as text, the identity of the operator to which the message is to be delivered. Following the Send Message to Operator major vector is one of three parameter major vectors:

- Text Data (X'1300')

- Structured Data (X'1307')

- Transparent Coded Datastream (X'1309')

2.10.6 EP_OPERATIONS_MGMT Function Set

```
      ........................          ....................
      .                      .          .                  .
      . MULTIPLE_DOMAIN_SUPPORT .       . MS_CAPS          .
      . function set group      .       . function set group .
      ........................          ....................
            "      |      "                 "          |
         (O)|   (Q)|   (R)|        (T)  ______|    _______| (V)
            |      |      |            |          |
°°°°°°°°°°°|°°°°°°|°°°°°°|°°°°°°°°°°°°|°°°°°°°°°°|°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°
°          |      |      |            |          |                              °   _______
°          |      |      |            |          |                              °  |appl. 2 |  .
°          |            |            |                                       °  _______  | .
°                                                                              °|appl. 1 | |.
°    ___________________________________________|            (A-3)             °|        | |
°    |                                          |___________________________ÿ|        | |
°    |                                          |            (A-4)            °|        | |
°    |                                          |___________________________|        | |
°    |                                  MS      |                             °|        |_|
°    |__________________________________________|                             °|        |
°                    |   "                      EP_OPERATIONS_MGMT             °|_______|
°                    |   |                      function set group             °  second-level
°°°°°°°°°°°°°°°°°°°°°|°°°|°°°°°°°°°°°°°°°°°°°
```

passed across protocol boundary V from the MS_CAPS function set group. The application program must have previously registered itself with MS_CAPS via protocol boundary T. The details of this protocol boundary are described in "Protocol Boundary V--Current Focal Point" in topic 2.9.1.17.

The process by which an application program registers itself with MS_CAPS is described in "Application Program Registration with MS_CAPS" in topic 2.9.3.10.

- Internal MS protocol boundary CC (Reporting Cancelation to Operations Management)

  The EP_OPERATIONS_MGMT function set group receives Cancelation CP-MSUs from the EP_CHANGE_MGMT function set group. The details of this protocol boundary are described in "Protocol Boundary CC--Reporting Cancelation to Operations Management" in topic 2.9.1.19.

° Output:

- Internal MS protocol boundary O (Register Application program Name)

  An application program utilizing the MDS transport registers itself with the MULTIPLE_DOMAIN_SUPPORT function set group via this protocol boundary. The details of this protocol boundary are described in "Protocol Boundary O--Register Application Program Name" in topic 2.9.1.11.

  The process by which an application program registers itself with MULTIPLE_DOMAIN_SUPPORT is described in "Registering with MDS" in topic 2.9.2.9.1.

- Internal MS protocol boundary R (Send MDS-MU)

  The EP_OPERATIONS_MGMT function set group requests the MULTIPLE_DOMAIN_SUPPORT function set group to send an MDS-MU. The output data is the complete MDS-MU. The details of this protocol boundary are described in "Protocol Boundary R--Send MDS-MU" in topic 2.9.1.13.

- MS application program program protocol boundary A-3

  The EP_OPERATIONS_MGMT function set group passes commands to second-level network management application program programs in its node. For each command it passes a complete MDS-MU, together with a correlator that the application program program will retain to pass back with the reply. The details of this protocol boundary are described in Table 10-45 in topic 2.10.6.4.3.

- Internal MS protocol boundary T (Register MS Category)

  In the process of initialization, management services application programs register themselves with the MS_CAPS function set group via protocol boundary T if they want focal point information. The application program passes its name and the category for which it is registering. The details of this protocol boundary are described in "Protocol Boundary T--Register MS Category" in topic 2.9.1.15.

  The process by which an application program registers itself with MS_CAPS is described in "Application Program Registration with MS_CAPS" in topic 2.9.3.10.

- Internal MS protocol boundary BB (Cancelation from Operations Management)

  The EP_OPERATIONS_MGMT function set group passes certain cancel request CP-MSUs to EP_CHANGE_MGMT for processing. Only cancel requests specifying that commands sent to EP_CHANGE_MGMT are to be canceled are affected. This protocol boundary is described in "Protocol Boundary BB--Cancelation From Operations Management" in topic 2.9.1.18.

2.10.6.2 Prerequisite Function Sets

See "Role Requirements for Management Services Components" in topic 2.8.1 for information on the relationships between this function set and other function sets.

2.10.6.3 Overview of Subsets

```
 ___________________________________________________
|                                                   |
|          no optional subsets available for        |
|           EP_OPERATIONS_MGMT function set         |
|                                                   |
|___________________________________________________|
|                                                   |
|           EP_OPERATIONS_MGMT base subset          |
|                                                   |
|___________________________________________________|
```

Figure 10-68. Base and Optional Subsets of EP_OPERATIONS_MGMT Function Set

---

2.10.6.4 EP_OPERATIONS_MGMT Base Subset

Subtopics:

- 2.10.6.4.1 Functions Provided
- 2.10.6.4.2 Formats Supported
- 2.10.6.4.3 Implementation Requirements

2.10.6.4.1 Functions Provided

The EP_OPERATIONS_MGMT base subset provides the capability to receive operations management commands from network operators and replies or reports from second-level application programs. Unsolicited messages may also be received from application programs served by EP_OPERATIONS_MGMT and sent to the operations management focal point.

2.10.6.4.2 Formats Supported

The EP_OPERATIONS_MGMT base subset supports the receiving of an MDS-MU containing one of five management services major vectors:

- Request Activation (X'8066')
- Operate (X'8070')
- Request Deactivation (X'8071')
- Set Clock (X'8075')
- Request Cancelation (X'8076')

In addition to the above major vectors, the following GDS variables and parameter major vectors may be received within the same MDS-MU (see request descriptions for usage):

- Transparent Coded Datastream (X'1309') parameter major vector
- Text Command (X'1320') GDS variable
- Partial Format Processing Method (X'1321') GDS variable
- Display Datastream (X'1322') GDS variable
- Context-Identified Values (X'1323') GDS variable
- Context Identifier Group (X'1324') GDS variable
- Context Identifier (X'1325') GDS variable
- Value Group (X'1326') GDS variable

- Character Value Descriptor (X'1327') GDS variable
- Value Processing Method (X'1328') GDS variable
- Value (X'1329') GDS variable
- Hexadecimal Value Descriptor (X'132A') GDS variable
- Nested Value Processing Method (X'132B') GDS variable
- Value Instance Identifier (X'1331') GDS variable
- Agent Unit of Work (X'1549') GDS variable
- Routing and Targeting Instructions (X'154D') GDS variable

The base subset processes the request itself if it is the target. It is considered to be the target of the request if there is no RTI in the CP-MSU or if the Name List within the RTI lacks either a Cascaded Resource Name List or Destination Application Name subfield.

If an RTI is present, then the EP_OPERATIONS_MGMT passes the request, across protocol boundary A-3, to a the second-level application program named within the RTI. EP_OPERATIONS_MGMT may also forward Request Cancelation CP-MSUs to EP_CHANGE_MGMT See "Request Cancelation CP-MSU" in topic 1.7.3.1.10 for more information.

The EP_OPERATIONS_MGMT base subset also supports receipt, from second-level network management application programs, the following management services major vectors:

- Activation Acceptance (X'0066')
- Activation (X'0067')
- Operate Report (X'0070')
- Deactivation Acceptance (X'0071')
- Deactivation (X'0072')
- Set Clock Report (X'0075')
- Cancelation (X'0076')
- Routing/Parsing Report (X'0077')

Also supported is the receipt of the following GDS variables and parameter major vectors from application programs served by EP_OPERATIONS_MGMT (see each reply and report for usage descriptions):

- Text Data (X'1300') parameter major vector
- Transparent Coded Datastream (X'1309') parameter major vector
- Partial Format Processing Method (X'1321') GDS variable
- Display Datastream (X'1322') GDS variable
- Context-Identified Values (X'1323') GDS variable
- Context Identifier Group (X'1324') GDS variable
- Context Identifier (X'1325') GDS variable
- Value Group (X'1326') GDS variable
- Character Value Descriptor (X'1327') GDS variable
- Value Processing Method (X'1328') GDS variable
- Value (X'1329') GDS variable

- Hexadecimal Value Descriptor (X'132A') GDS variable

- Nested Value Processing Method (X'132B') GDS variable

- Value Instance Identifier (X'1331') GDS variable

- SNA Condition Report (X'1532') GDS variable

- Agent Unit of Work (X'1549') GDS variable

- Routing and Targeting Instructions (X'154D') GDS variable

See "Request Cancelation CP-MSU" in topic 1.7.3.1.10 for more information concerning the processing of Cancelation CP-MSUs sent from EP_CHANGE_MGMT.

2.10.6.4.3 Implementation Requirements

The requirements for implementing the EP_OPERATIONS_MGMT base subset are described by a model consisting of a subset of management services.

A Subset of Management Services

Receiving Operations Management Requests:

This process is started by the process described in "Routing MDS-MU Received to Local MS Application Programs" in topic 2.9.2.4.8 (MULTIPLE_DOMAIN_SUPPORT function set group), when the latter has identified EP_OPERATIONS_MGMT (in its own node) as the destination application program in the MDS header.

Upon request by the MULTIPLE_DOMAIN_SUPPORT function set group (via protocol boundary Q), EP_OPERATIONS_MGMT does the following:

- Parses the request for validity.

- If an error was detected, builds an MDS-MU containing a negative reply and passes it to the MULTIPLE_DOMAIN_SUPPORT function set group (via protocol boundary R). The general format and construction of an application program-generated MDS-MU for a negative reply are discussed in "Building Negative Replies for MDS-MUs" in topic 2.9.2.9.4.

- If the MDS-MU parses correctly, then either of the following occurs:

  - If an RTI GDS variable is present containing at least a Destination Application Name, the entire MDS-MU is passed to the identified second-level network management application program (using application program protocol boundary A-3 described below).

Table 10-45. MS Application Protocol Boundary A-3	
Origin	EP_OPERATIONS_MGMT
Destination	The second-level network management application program defined in the Routing and Targeting Instructions (X'154D') GDS variable of the operations management request.
Data Content	1. An entire MDS-MU containing the operations management request.
Initiates	The network management application identified in the operations management request.

  - If an RTI GDS variable is not present; or, if an RTI is present but lacks a Destination Application Name, then one of the following conditions applies:

    -- If the request does not contain a Request Cancelation (X'8076') major vector that specifies that requests sent to EP_CHANGE_MGMT are to be canceled, then EP_OPERATIONS_MGMT is the intended target of the request.

-- If the request does contain a Request Cancelation (X'8076') major vector that specifies that requests sent to EP_CHANGE_MGMT are to be canceled, EP_OPERATIONS_MGMT (or the destination application program named in the RTI) does the following:

° Saves the MDS and RTI routing information from the request.

° Forwards the CP-MSU to EP_CHANGE_MGMT (in the same node) using protocol boundary BB, described in "Protocol Boundary BB--Cancelation From Operations Management" in topic 2.9.1.18 and "EP_CHANGE_MGMT Optional Subset 7 (Cancel Command Support)" in topic 2.10.4.11.

After processing the MDS-MU (or passing it to either a designated second-level application program or EP_CHANGE_MGMT), this process terminates.

Sending Operations Management Data:

Upon request by a second-level application program (or, as the result of processing a command itself), EP_OPERATIONS_MGMT does the following:

° Builds (or receives from a second-level application program or EP_CHANGE_MGMT) a complete management services major vector, along with any necessary GDS variables and parameter major vectors. The routing and targeting instructions information will have as its origin and destination information an exact reversal of the origin and destination of the original request. In other words, the routing and targeting instructions of the reply and/or report will indicate that it is to be sent back to the sender of the original request. For a reply or report, the Correlator Control subvector correlator information that it received within the Routing and Targeting Instructions GDS variable (using MS application program protocol boundary A-4 described below) is included.

Table 10-46. MS Application Protocol Boundary A-4

Origin	A second-level network management application program
Destination	EP_OPERATIONS_MGMT
Data Content	1. Major vectors together with GDS variables and parameter major vectors that constitute a reply or report that the served application program has built
Initiates	Process described in "Sending Operations Management Data"

° Using the major vectors, parameter major vectors, and GDS variables, builds a complete MDS-MU.

For a reply or report, this process echoes the Agent Unit-of-Work Correlator (X'1549') GDS variable passed to it in the original request. Or it may echo a private correlator (contained in the Correlator Control (X'0D') subvector of the request. The Correlator Control subvector settings, as explained in "MDS Message Types and Application-Level Flows" in topic 1.7.3.3.1, indicate how the correlator is to be interpreted by the second-level application that is the destination of the reply/report, and thus how the reply/report is related to the original request.

This process then constructs a complete MDS-MU. The destination NAU and application program for the MDS-MU are those from which the request MDS-MU originated. The major vectors and GDS variables from the application program are enveloped within a CP_MSU, which flows as the application program GDS variable in the MDS-MU.

This process starts the process described in "Processing Send MDS-MU" in topic 2.9.2.4.8 (MULTIPLE_DOMAIN_SUPPORT function set group) via protocol boundary R, to send the MDS-MU to the application program identified in the MDS Routing Information structure within the MDS-MU. As "Protocol Boundary R--Send MDS-MU" in topic 2.9.1.13 indicates, two items are passed when starting this process:

- A pointer to the MDS-MU

- "DEFAULT" or "DIRECT" (as indicated by MS_CAPS in "Protocol Boundary T--Register MS Category" in topic 2.9.1.15.

After this process has started the "Processing Send MDS-MU" in topic 2.9.2.4.8 process, it terminates.

2.10.6.5 Operations Management Replies, Reports, and Unsolicited Traffic

The architecture for operations management provides for the transfer of operations commands from a network operator to a specified network management application program, and the return of the corresponding replies and reports from the application program to the operator. It also provides for the transfer of unsolicited messages from an application program to a specified operator. The remainder of this section provides a description of these structures.

Subtopics:

- 2.10.6.5.1 Activation Acceptance Reply and Activation Report
- 2.10.6.5.2 Deactivation Acceptance Reply and Deactivation Report
- 2.10.6.5.3 Cancelation
- 2.10.6.5.4 Set Clock Report
- 2.10.6.5.5 Operate Report
- 2.10.6.5.6 Routing/Parsing Report

2.10.6.5.1 Activation Acceptance Reply and Activation Report

These structures are sent in response to a Request Activation command from the operator. The Activation Acceptance reply indicates whether the command has been accepted for execution. The Activation report gives an indication of the success of the execution, as well as the stage of execution, if in fact the Request Activation command was accepted.

Activation Acceptance: The Activation Acceptance (X'0066') major vector is built with the following subvectors:

° The Date/Time (X'01') subvector, which specifies the date and time that the reply was built.

° The Activation Acceptance (X'82') subvector, which includes the following subfield:

- The Attempt Status (X'10') subfield, which specifies if the target of the Request Activation command will or will not attempt the activation.

° The Detailed Data (X'98') subvector, which transports product-specific data concerning activation acceptance. The product-specific information is carried in one of two mutually exclusive subfields:

- The Detailed Data Extended (X'85') subfield, which carries a 4-byte data ID code point identifying the type of detailed data that follows it.

- The Detailed Data (X'82') subfield, which carries a 2-byte data ID code point identifying the type of detailed data that follows it.

In addition to the Activation Acceptance major vector, the Routing and Targeting Instructions (X'154D') may be included within the CP-MSU. This GDS variable may specify the route that the fully intact MDS-MU takes to get to the intended target of the reply, after reaching the operations management focal point itself. The information in the Routing and Targeting Instructions is a reversal of origin and destination information carried in the original request. The origin and destination application names are reversed, as are the origin instance and destination instance identifiers, if present. If more than one application name appears in the cascaded resource name list, then their order is reversed also. The Routing and Targeting Instructions is not required unless second-level application programs are involved as originators or destinations of requests, replies, or reports. See "The Routing and Targeting Instructions GDS Variable" in topic 1.7.3.3.1 for more information.

If the execution will not be attempted, the SNA Condition Report (X'1532') GDS variable (X'1532') is also included within the CP-MSU to indicate the reason.

Activation: The Activation major vector (X'0067') is built with the

following subvectors:

° The Date/Time (X'01') subvector, which specifies the date and time that the report was built.

° The Detailed Data (X'98') subvector, which transports product-specific data concerning the attempted activation. The product-specific information is carried in one of two mutually exclusive subfields:

- The Detailed Data Extended (X'85') subfield, which carries a 4-byte data ID code point identifying the type of detailed data that follows it.

- The Detailed Data (X'82') subfield, which carries a 2-byte data ID code point identifying the type of detailed data that follows it.

° The Reporting Activation (X'82') subvector, which may carry the following information:

- The Activation Status (X'10') subfield, which indicates whether or not the activation was successful.

- The Activation Stage (X'18') subfield, which indicates, at the time the report is built, the target's stage of activation. Multiple reports can be sent for each defined stage of activation.

° The Change Management Activation Use Status (X'20') subfield, which indicates:

- Both trial and production components were searched for in the activation

- Only production components were searched for in the activation

- The components searched were those searched in the preceding activation

- Change management in general, or the Change Management Activation Use subfield of the request is not supported supported by the receiver of the command

- A special condition prevails at the reporting node.

  The special condition indication, if present, indicates that one of two special conditions exists at the reporting node:

  -- When the CMAU (change management activation use) parameter in the request was set to "trial and production" (or set to "last" with the previous activation having specified "trial and production"), only production components were found despite having searched for trial and production components.

  -- When the CMAU parameter in the request was set to "production only," production activation was requested to override a previous failing trial activation.

In addition to the Activation major vector, the Routing and Targeting Instructions (X'154D') may be included within the CP-MSU. This GDS variable may specify the route that the fully intact MDS-MU takes to get to the intended target of the reply, after reaching the operations management focal point itself. The information in the Routing and Targeting Instructions is a reversal of origin and destination information carried in the original request. The origin and destination application names are reversed, as are the origin instance and destination instance identifiers, if present. If more than one application program name appears in the cascaded resource name list, then their order is reversed also. The Routing and Targeting Instructions is not required unless second-level application programs are involved as originators or destinations of requests, replies, or reports. See "The Routing and Targeting Instructions GDS Variable" in topic 1.7.3.3.1 for more information.

If the execution is not successful, the SNA Condition Report (X'1532') GDS variable, containing the applicable sense data in the SNA_REPORT_CODE field, (X'1532') is also included within the CP-MSU to indicate the reason.

2.10.6.5.2 Deactivation Acceptance Reply and Deactivation Report

These structures are sent in response to an Request Deactivation command from the operator. The Deactivation Acceptance reply indicates whether the command has been accepted for execution. The Deactivation report gives an indication of the success of the execution if the Request Deactivation command was accepted.

Deactivation Acceptance: The Deactivation Acceptance (X'0071') major vector is built with the following subvectors:

- The Date/Time (X'01') subvector, which specifies the date and time that the reply was built.

- The Deactivation Acceptance (X'82') subvector, which includes the following subfield:

  - The Attempt Status (X'10') subfield, which specifies if the target of the Request Deactivation command will or will not attempt the deactivation.

- The Detailed Data (X'98') subvector, which transports product-specific data concerning the acceptance/rejection of the deactivation. The product-specific information is carried in one of two mutually exclusive subfields:

  - The Detailed Data Extended (X'85') subfield, which carries a 4-byte data ID code point identifying the type of detailed data that follows it.

  - The Detailed Data (X'82') subfield, which carries a 2-byte data ID code point identifying the type of detailed data that follows it.

In addition to the Deactivation Acceptance major vector, the Routing and Targeting Instructions (X'154D') may be included within the CP-MSU. This GDS variable may specify the route that the fully intact MDS-MU takes to get to the intended target of the reply, after reaching the operations management focal point itself. The information in the Routing and Targeting Instructions is a reversal of origin and destination information carried in the original request. The origin and destination application names are reversed, as are the origin instance and destination instance identifiers, if present. If more than one application name appears in the cascaded resource name list, then their order is reversed also. The Routing and Targeting Instructions is not required unless second-level application programs are involved as originators or destinations of requests, replies, or reports. See "The Routing and Targeting Instructions GDS Variable" in topic 1.7.3.3.1 for more information.

If the execution will not be attempted, the SNA Condition Report (X'1532') GDS variable (X'1532'), containing sense data in the SNA_REPORT_CODE field, is also included within the CP-MSU to indicate the reason.

Deactivation: The Deactivation major vector (X'0072') is built with the following subvectors:

- The Date/Time (X'01') subvector, which specifies the date and time that the report was built.

- The Detailed Data (X'98') subvector, which transports product-specific data concerning the attempted deactivation. The product-specific information is carried in one of two mutually exclusive subfields:

  - The Detailed Data Extended (X'85') subfield, which carries a 4-byte data ID code point identifying the type of detailed data that follows it.

  - The Detailed Data (X'82') subfield, which carries a 2-byte data ID code point identifying the type of detailed data that follows it.

- The Reporting Deactivation (X'82') subvector, which may carry the following information:

  - The Deactivation Status (X'10') subfield, which indicates whether or not the deactivation was successful.

In addition to the Deactivation major vector, the Routing and Targeting Instructions (X'154D') may be included within the CP-MSU. This GDS variable may specify the route that the fully intact MDS-MU takes to get to the intended target of the reply, after reaching the operations management focal point itself. The information in the Routing and Targeting Instructions is a reversal of origin and destination information carried in the original request. The origin and destination application names are reversed, as are the origin instance and destination instance

identifiers, if present. If more than one application name appears in the cascaded resource name list, then their order is reversed also. The Routing and Targeting Instructions is not required unless second-level application programs are involved as originators or destinations of requests, replies, or reports. See "The Routing and Targeting Instructions GDS Variable" in topic 1.7.3.3.1 for more information.

If the execution is not successful, the SNA Condition Report (X'1532') GDS variable, (X'1532') containing sense data in the SNA_REPORT_CODE field, is also included within the CP-MSU to indicate the reason.

2.10.6.5.3 Cancelation

This structure is sent in response to a Request Cancelation command from the operator. The Cancelation indicates whether or not the command has been accepted for execution. If the execution is attempted, additional Cancelations may be sent, at implementation-defined intervals, giving an indication that the execution is in progress but not yet completed. Finally, the report may be used to indicate the success or failure of the execution, once it has normally or abnormally completed.

EP_CHANGE_MGMT may be a target of a Request Cancelation sent by the operations management operator.

Note that before EP_OPERATIONS_MGMT forwards a cancel request to EP_CHANGE_MGMT, the MDS and RTI routing information is saved. After EP_CHANGE_MGMT processes the request, it builds at least one report, and possibly more:

- A Cancelation CP-MSU is always built and sent to EP_OPERATIONS_MGMT over MS protocol boundary CC. EP_OPERATIONS_MGMT then encloses the CP-MSU within an MDS-MU (using the MDS and RTI routing information saved from the Request Cancelation) and sends the MDS-MU to the Request Cancelation originator.

- One or more reports are built and returned to the change management focal point for each request that was affected by the cancel process. Each report carries a sense code that indicates that the related request was canceled becuase of operator intervention. If the cancelation process found no requests to be canceled, then EP_CHANGE_MGMT does not send any reports to the change management focal point.

The Cancelation (X'0076') major vector is built with the following subvectors:

- The Date/Time (X'01') subvector, which specifies the date and time that the report was built.

- The Reporting Cancelation (X'82') subvector, which includes the following subfield:

  - The Cancelation Status (X'10') subfield, which may indicate:

    -- acceptance or rejection of the request

    -- the success (or failure) of the execution, if in fact the request was accepted earlier

    -- an indication that, if the request has been accepted, execution is currently proceeding but has not yet completed.

  - The Detailed Data (X'98') subvector, which transports product-specific data concerning cancelation acceptance/rejection (or success/failure, depending on the purpose of the report) of the cancel execution. The product-specific information is carried in one of two mutually exclusive subfields:

    -- The Detailed Data Extended (X'85') subfield, which carries a four-byte data ID code point identifying the type of detailed data which follows it.

    -- The Detailed Data (X'82') subfield, which carries a two-byte data ID code point identifying the type of detailed data which follows it.

In addition to the Cancelation major vector, the Routing and Targeting Instructions (X'154D') may be included within the CP-MSU. This GDS variable may specify the route that the fully intact MDS-MU takes to get to the intended target of the reply, after reaching the operations management focal point itself. The information in the Routing and

Targeting Instructions is a reversal of origin and destination information carried in the original request. The origin and destination application names are reversed, as are the origin instance and destination instance identifiers, if present. If more than one application name appears in the cascaded resource name list, then their order is reversed also. The Routing and Targeting Instructions is not required unless second-level application programs are involved as originators or destinations of requests, replies, or reports. See "The Routing and Targeting Instructions GDS Variable" in topic 1.7.3.3.1 for more information.

If the execution will not be attempted (or is attempted but is not successful), the SNA Condition Report (X'1532') GDS variable, containing sense data in the SNA_REPORT_CODE field, is also included within the CP-MSU.

2.10.6.5.4 Set Clock Report

This structure is sent in response to a Set Clock command from the operator. The Set Clock Report indicates whether or not the command has been accepted for execution. If the execution is attempted, additional Set Clock Reports may be sent, at implementation-defined intervals, giving an indication that the execution is in progress but not yet completed. Finally, the report may be used to indicate the success or failure of the execution, once it has normally or abnormally completed.

The Set Clock Report (X'0075') major vector is built with the following subvectors:

- The Date/Time (X'01') subvector, which specifies the date and time that the report was built.

- The Reporting Set Clock (X'82') subvector, which includes the following subfield:

  - The Set Clock Status (X'10') subfield, used to indicate:

    - Acceptance or rejection of the request

    - the success (or failure) of the execution, if in fact the request was accepted earlier

    - An indication that, if the request has been accepted, execution is currently proceeding but has not yet completed.

  - The Detailed Data (X'98') subvector, which transports product-specific data concerning the acceptance/rejection (or success/failure, depending on the purpose of the report) of the set clock execution. The product-specific information is carried in one of two mutually exclusive subfields:

    - The Detailed Data Extended (X'85') subfield, which carries a 4-byte data ID code point identifying the type of detailed data that follows it.

    - The Detailed Data (X'82') subfield, which carries a 2-byte data ID code point identifying the type of detailed data that follows it.

In addition to the Set Clock Report major vector, the Routing and Targeting Instructions (X'154D') may be included within the CP-MSU. This GDS variable may specify the route that the fully intact MDS-MU takes to get to the intended target of the reply, after reaching the operations management focal point itself. The information in the Routing and Targeting Instructions is a reversal of origin and destination information carried in the original request. The origin and destination application names are reversed, as are the origin instance and destination instance identifiers, if present. If more than one application program name appears in the cascaded resource name list, then their order is reversed also. The Routing and Targeting Instructions is not required unless second-level application programs are involved as originators or destinations of requests, replies, or reports. See "The Routing and Targeting Instructions GDS Variable" in topic 1.7.3.3.1 for more information.

If the execution will not be attempted (or is attempted but is not successful), the SNA Condition Report (X'1532') GDS variable (X'1532'), containing sense data in the SNA_REPORT_CODE field, is also included within the CP-MSU to indicate the reason.

2.10.6.5.5 Operate Report

This structure is sent in response to an Operate command from the operator. The Operate Report indicates whether the command has been accepted for execution. If the execution is attempted, additional Operate Reports may be sent at implementation-defined intervals, giving an indication that the execution is in progress but not yet completed.

The report may also be used to indicate the success or failure of the execution, once it has normally or abnormally completed.

Additionally, an entry point operator may use this structure to send a message to the operations management focal point operator. Such a report would be a type of ancillary report. As a result of receiving (and successfully executing) an Operate command, the target of that command may be placed into a certain mode of operation. This mode may persist, even though the original command execution that caused it completed some time in the past (with success reports having been sent accordingly). For example, a target may be placed into a mode in which, at certain time intervals, display datastreams containing monitored data are sent to the Operate issuer. The Operate Report may be used to transport these "ancillary" reports. However, any unsolicited message (whether sent as a result of being in a certain mode or not) is sent as an ancillary report.

The Operate Report (X'0070') major vector is built with the following subvectors:

- The Date/Time (X'01') subvector, which specifies the date and time that the report was built.

| - The Reporting Operation (X'82') subvector, which may include the following subfield:

  - The Operation Status (X'10') subfield, which is used differently depending on what the target of the operate command is reporting:

    -- Acceptance or rejection of the request

    -- An indication that, if the request has been accepted, execution is currently proceeding but has not yet completed.

    -- The success (or failure) of the execution, if in fact the request was accepted

    -- An indication that the report is related to some persistent modal status that is active at the report issuer (ancillary). This is the default meaning if the Operation Status subfield is not present in the Reporting Operation subvector.

  - The Change Management Activation Use Status (X'20') subfield, which indicates, if the enveloped command resulted in an activation, whether:

    -- Both trial and production configuration components were searched for and used in the activation

    -- Only production components were searched for and used in the activation

    -- The components used were those used in the preceding activation

    -- Change management, or the Change Management Activation Use subfield of the request is not supported by the receiver of the request

    -- A special condition prevails at the reporting node. The special condition indication, if present, brings attention to the fact that one of two special conditions exist at the reporting node:

      - When the CMAU (change management activation use) parameter in the request was set to "trial and production" (or set to "last" with the previous activation having specified "trial and production"), only production components were found despite having searched for trial and production components.

      - When the CMAU parameter in the request was set to "production only," production activation was requested to override a previous failing trial activation.

- The Detailed Data (X'98') subvector, which transports product-specific data concerning the acceptance/rejection (or success/failure, depending on the purpose of the report) of the execution of the operate. The product-specific information is carried in one of two mutually exclusive subfields:

  - The Detailed Data Extended (X'85') subfield, which carries a 4-byte data id code point identifying the type of detailed data which follows it.

  - The Detailed Data (X'82') subfield, which carries a 2-byte data id code point identifying the type of detailed data which follows it.

- The Default Character Set ID (X'32') subvector, which establishes the character set and code page to be used for interpreting all character fields within the Operate major vector, unless such fields explicitly specify a character set and code page.

In addition to the Operate Report major vector, the following GDS variables and parameter major vectors may be included within the CP-MSU:

- The Partial Format Processing Method (X'1321') GDS variable specifies, in character string format, the format type of either a transparent coded datastream (X'1309') or a display datastream (X'1322') that is included within the report.

- The Transparent Coded Datastream (X'1309') parameter major vector may be included to transport a non-display datastream. The Partial Format Processing Method must be present to indicate the format of such a datastream. Transparent, in this sense, refers to the fact that the datastream meaning is transparent to MS, but understood by the target of the request.

- The Display Datastream (X'1322') GDS variable may be included to transport a display datastream. The Partial Format Processing Method must be present to include the format of such a datastream. The meaning of such a datastream is transparent to MS. The Default Character Set ID specifies the character set, code page, and a national language identifier that is used to encode the Display Datastream data.

- The Text Data (X'1300') parameter major vector may be included to tranport text data. The format of the data itself is not defined by MS.

- The Context-Identified Values (X'1323') GDS variable may be included to carry enveloped data that is not interpreted by EP_OPERATIONS_MGMT or the operations management focal point. See page 1.7.3.1 for more information concerning the use of this GDS variable.

- The Routing and Targeting Instructions (X'154D') GDS variable may be included within the CP-MSU. This GDS variable may specify the route that the fully intact MDS-MU takes to get to the intended target of the reply, after reaching the operations management focal point itself. The information in the Routing and Targeting Instructions is a reversal of origin and destination information carried in the original request. The origin and destination application names are reversed, as are the origin instance and destination instance identifiers, if present. If more than one application name appears in the cascaded resource name list, then their order is reversed also. The Routing and Targeting Instructions is not required unless second-level application programs are involved as originators or destinations of requests, replies, or reports. See "The Routing and Targeting Instructions GDS Variable" in topic 1.7.3.3.1 for more information.

- If the execution will not be attempted (or is attempted but is not successful), the SNA Condition Report (X'1532') GDS variable (X'1532'), containing sense data in the SNA_REPORT_CODE field, is included within the CP-MSU to indicate the reason.

2.10.6.5.6 Routing/Parsing Report

The Routing/Parsing Report is used by second-level application programs when:

- A second-level routing error occurs

- Zero or more than one major vectors are found when parsing the CP-MSU

- The major vector in the request is unrecognized or unsupported.

Examples of a second-level routing errors may include unsupported or unrecognized resource or application program names within the Routing and Targeting Instructions (X'154D') GDS variable. Also, the Routing and Targeting Instructions GDS variable may be malformed, thus triggering a Routing/Parsing Report to be sent. It is a violation of the architecture to have more than one major vector, or zero major vectors, present within an operations management CP-MSU. The Routing/Parsing Report, using the SNA Condition Report (X'1532') GDS variable, reports such errors.

The description of the error is carried in an SNA Condition Report (X'1532') GDS variable and returned to the application program (identified in the routing and targeting instructions) that issued the request. See "Operations Management Use of the SNA Condition Report (SNACR)" in topic 1.7.3.1 for more information.

As with the other reports, the routing and targeting instructions information in the original malformed request is reversed, with respect to origin and destination names (and instances, if they are present). The Routing/Parsing Report is sent back over the path in the opposite direction. If no routing and targeting instructions information was present in the request with the error, the Routing/Parsing Report is sent back to the application program identified as the origin in the MDS-MU. Although the MULTIPLE_DOMAIN_SUPPORT unit of work will be completed by doing this, the second-level application program unit of work will be left outstanding, therefore triggering the request-originator to re-send the command. Time-outs of second-level application programs are implementation-specific and are not defined by this architecture.

The Routing/Parsing Report (X'0077') major vector contains no subvectors. Its key identifies to the receiver that either a parsing or second-level application program (*not MDS*) routing error has occurred. The significant information is carried in the SNA Condition Report (X'1532') GDS variable.

The following GDS variables are contained within the CP-MSU, along with the Routing/Parsing Report major vector:

- The Routing and Targeting Instructions (X'154D') GDS variable may be included within the CP-MSU to specify the route that the fully intact MDS-MU should take to get to the intended target of the report, after reaching the operations management focal point itself. The information in the Routing and Targeting Instructions is a reversal of origin and destination information carried in the original request. The origin and destination application names are reversed, as are the origin instance and destination instance identifiers, if present. If more than one application name appears in the cascaded resource name list, then their order is reversed also. See "The Routing and Targeting Instructions GDS Variable" in topic 1.7.3.3.1 for more information.

- The SNA Condition Report (X'1532') GDS variable, containing sense data in the SNA_REPORT_CODE field, is included within the CP-MSU to indicate the second-level routing error or the parsing error that has occurred.

2.10.7 Common EP_XXXX Functions

Subtopics:

- 2.10.7.1 Building the Date/Time (X'01') and Relative Time (X'42') Subvectors
- 2.10.7.2 Building the SNA Address List (X'04') Subvector
- 2.10.7.3 Building the Product Set ID (X'10') Subvector
- 2.10.7.4 Building a Management Services Major Vector
- 2.10.7.5 Building an NMVT
- 2.10.7.6 Building an MDS-MU
- 2.10.7.7 Parsing of NMVTs

2.10.7.1 Building the Date/Time (X'01') and Relative Time (X'42') Subvectors

There are four electives available to implementations for time stamping management services major vectors:

- The choice of providing either a Date/Time (X'01') subvector or Relative Time (X'42') subvector in the major vector. This elective applies only to a major vector contained in an NMVT. If a major vector contained in a CP-MSU is required to have a time stamp at all, that time stamp must be the Date/Time subvector.

- If the implementation elects to provide a Relative Time subvector, it has a further choice of the increment of time measurement.

- If the implementation elects to provide a Date/Time subvector, it has a further choice of providing an indication of time at a greater precision than seconds, e.g., milliseconds or microseconds.

- If the implementation elects to provide a Date/Time subvector, it has a further choice of providing the offset between the local time sent in the Date/Time subvector and Greenwich Mean Time.

Depending on the capabilities of the node, the elective for either date/time or relative time will be selected.

- If the elective for providing a date/time subvector is selected, this process constructs a Date/Time (X'01') subvector. If the elective for providing time at a finer granularity than seconds is selected, the binary value providing finer granularity than seconds is placed in the subvector field denoted "Optional extension of time." If the elective for providing GMT offset is selected, the subfield containing this offset is included.

- If the elective for providing a relative time subvector is selected, this process constructs a Relative Time (X'42') subvector. The time increment of measure is specified according to the elective selected. Any of the units of measure listed in the definition of the Relative Time subvector are equally acceptable.

2.10.7.2 Building the SNA Address List (X'04') Subvector

A process building an SNA Address List (X'04') subvector starts with either a single local address, or a pair of local addresses. In the latter case one address is for a local resource, and the other is for its session partner. The process proceeds as follows:

- The Address format field in the subvector is filled in:

  - For a single local address, the value X'00' (one or more single local addresses) is placed in this field.

  - For a pair of local addresses, the value X'40' (one or more pairs of session partner local addresses, each pair identifying a session) is inserted.

- The address of the local resource is inserted into the subvector.

- If a session partner address is present, it is inserted in the subvector *after* the resource address.

- The Address Count field in the subvector is filled in: X'01' for a single address, X'02' for an address pair.

2.10.7.3 Building the Product Set ID (X'10') Subvector

A process building a Product Set ID (X'10') subvector performs the sequence of steps described below. The Product Set ID subvector contains one or more Product Identifier (X'11') MS subvectors. The following discussion provides details on how these subvectors are constructed.

In certain cases the EP_ALERT function set group constructs two Product Set IDs, one identifying the node sending the Alert, the other identifying the origin of the Alert condition. For the first of these two Product Set IDs, the term "resource" in the discussion below refers to the PU sending the Alert. For the second Product Set ID it refers to the the origin of the Alert condition.

X'10': This serves as an "envelope" for either:

  - One X'11' subvector for a resource that implements all of its product set in hardware; or

  - Two or more X'11' subvectors for a resource that implements part or all of its product set in software. One subvector identifies the hardware, the others identify the software. If the resource is comprised of more than one software product, a X'11' subvector is built for each product.

X'11'(hardware): The Product Identifier subvector identifying the hardware is constructed containing the following subfields:

- The Hardware Product Identifier (X'00') subfield is constructed. The machine type and serial number (or repair ID number) are required from all products; the model number is also required if it is applicable. The format type is determined as follows:

  - One factor in selecting the format type is the mechanism used for identifying instances of the product. Individual instances of a product may be identified by serial number, by repair ID number, or by both. All products are assigned serial numbers; some are assigned repair ID numbers as well.

    The repair ID number serves to identify a product instance for such purposes as service contracts. When a customer first acquires a product instance, the repair ID number and the serial number are typically the same. If, however, the original instance of the product fails and must be exchanged for a new one, the original repair ID number is carried over to the new instance; thus the repair ID number and the serial number for the replacement product are no longer identical.

    For products having numbers of both kinds, the one that is *more visible externally on the product* is the one sent in this subfield. For serial numbers, one of the X'10', X'11', or X'12' formats is chosen. For repair ID numbers, one of the X'20', X'21', or X'22' formats is chosen.

  - The other factor determining the format used is the amount of information needed to identify a product instance uniquely.

    -- The X'11' (or X'21') format is used when model number, machine type, and serial number (or repair ID number) are required to uniquely identify a product instance.

    -- The X'12' (or X'22') format is used when model number does not assist in uniquely identifying a product instance, but is provided for the purpose of additional information only.

    -- The X'10' (or X'20') format is used when machine type and a serial number (repair ID number) are required to uniquely identify a product instance and a model number is not provided.

- The Microcode EC Level (X'0B') subfield is constructed; inclusion of this subfield is optional for a product.

- The Hardware Product Common Name (X'0E') subfield is constructed; inclusion of this subfield is optional for a product.

- The Emulated Product Identifier (X'01') subfield is constructed when a product is emulating another hardware product.

- The Vendor ID (X'0F') subfield is constructed when the hardware product is manufactured by other than IBM; inclusion of this subfield is optional for a non-IBM product. This subfield is not included when the Product Set ID (X'10') common subvector is included in the Alert (X'0000') major vector.

X'11'(software): If any of the resource's product set is implemented in software, the Product Identifier subvector identifying the software is constructed containing the following subfields:

- The Software Product Serviceable Component Identifier (X'02') subfield is constructed if the product is supported by the IBM National Service Division (NSD).

- The Software Product Program Number (X'08') and Software Product Common Level (X'04') subfields are constructed for all software products that are not assigned a serviceable component identifier (are not supported by the National Service Division).

- The Software Product Common Name (X'06') subfield is always constructed. It provides a user-friendly means of identifying a software product.

- The Software Product Customization Identifier (X'07') or Software Product Customization Date and Time (X'09') subfield is constructed if the software product is customer modifiable. It provides unique identification of a particular set of instructions when multiple copies of the same software product exists in a system.

- The Vendor ID subfield is constructed when the software product is manufactured by other than IBM.

2.10.7.4 Building a Management Services Major Vector

A process constructs a major vector by enveloping the subvectors it has constructed earlier. Figure 2-3 in topic 1.2.1.1.1 shows the relationship between the 2-byte major vector length, the 2-byte major vector key, and the subvectors. There are four constraints on the placement of the subvectors within a major vector:

One constraint applies to all major vectors:

- If it is present, the SNA Address List (X'04') subvector must be placed first.

Three additional constraints are unique to the Alert (X'0000') major vector:

- Any Detail Qualifier (X'A0' or X'A1') subvectors must be placed in the major vector by EP_ALERT in the same order that they were received from the LMS. This is done to insure that the qualifiers can be displayed to the network operator at the control point in the same order they were passed from the Alerting component to EP_ALERT.

- If both a Hierarchy Name List (X'03') and a Hierarchy/Resource List (X'05') subvector are included, the Hierarchy/Resource List subvector is placed ahead of the Hierarchy Name List subvector in the major vector.

- If two Product Set ID (X'10') subvectors are included, the one identifying the PU sending the Alert appears before the one identifying the origin of the Alert condition.

2.10.7.5 Building an NMVT

A process building an NMVT proceeds as follows:

Note: Any fields not described below are set to 0.

- The NS Header (X'41038D') is placed in bytes 0-2.

- The PRID, bits 4-15 of bytes 5-6, is filled in:

  - For an unsolicited NMVT, the value X'000' is inserted.

  - For a reply NMVT, the PRID value from the request is inserted.

- The flags (byte 7) are set:

  - The solicitation indicator is set to 0 for an unsolicited NMVT, 1 for a reply.

  - For unsolicited NMVTs, the sequence field is filled with B'00' (only NMVT for this PRID).

  - For a reply, the process building the NMVT places the appropriate value in the sequence field. Different processes use different techniques for determining what value belongs in this field. EP_COMMON_OPERATIONS_SERVICES, for example, supports only a single reply per request, but the reply can span multiple NMVTs. So, in the case where the reply fits in a single NMVT, B'00' (only NMVT for this PRID) is used, while in the spanned NMVT case, B'10', B'11' or B'01' (first, middle or last NMVT for this PRID) are used accordingly. EP_QPI, on the other hand, is passed an explicit indication with each reply by the physical resources manager LMS.

  - The SNA Address List indicator is set to either 0 or 1, depending upon whether an SNA Address List subvector was included in the major vector.

- The major vector is inserted, beginning in byte 8.

If the process is building an NMVT to carry reply data solicited by either a command or request, the NMVT may not exceed 512 bytes in size. If it does, the process must build multiple NMVTs to carry the data after expanding the major vector, which always appears in the first NMVT, to carry the NMVT Count (X'44') subvector. The appropriate sequence field bits, as described above, must be set for each NMVT.

2.10.7.6 Building an MDS-MU

A process in an EP_XXXX application program building an MDS-MU constructs the elements of the MDS-MU as follows:

- *Origin Location:* The origin location has three elements: NETID, NAU name, and application program name. The first two of these pertain to the NAU containing the EP_XXXX application program building the MDS-MU. This information was passed to EP_XXXX at node initialization.

  The application program name is the name under which EP_XXXX has registered itself with MDS. See "Registering with MDS" in topic 2.9.2.9.1. MDS will use this name to route any MDS error messages that arise back to EP_XXXX.

- *Destination Location:* The destination location has the same three elements as the origin location. An EP_XXXX application program gets this information in one of three ways:

  - For a reply, it takes the information from the origin location field in the request.

  - For unsolicited data, e.g., an Alert, that is sent to a focal point, the EP_XXXX application program has registered itself with the MS_CAPS function set group, as an application program needing knowledge of the current focal point for a particular category of management services. See "Application Program Registration with MS_CAPS" in topic 2.9.3.10.

  - For unsolicited data not directed to a focal point, e.g., the common operations services Send Message to Operator, the EP_XXXX application program is passed the destination information along with the major vector.

- *Flags:* MDS message type, first MDS message indicator, and last MDS message indicator, are set by the EP_XXXX application program based on the nature of the message it is sending. The fourth flag, the routing verification indicator, is set by MDS itself: an application program using the MDS transport always sets this flag to 0.

- *Agent Unit of Work Correlator:* For a reply, this value is taken from the request. For unsolicited records, it is generated by the initiating application program. Figure 2-17 in topic 1.2.3.2.2 shows an example of this field.

- *Application Program GDS variable:* The CP-MSU that this process has built.

2.10.7.7 Parsing of NMVTs

The following table describes a parsing sequence that is common to all NMVT requests. This level of parsing is performed by the RECEIVE_REQUEST_SSCP_PU function set, before it passes the NMVT to an EP_XXXX function set. A discussion of the meaning of the table entries appears after the table.

Table 10-47. Sense Data Returned by PUMS After Receipt of an NMVT Request

Order of Checking	Mandatory / Optional	Sense Data	Condition	How Sent
1	Mandatory	X'1003 0001'	NMVTs not supported	-RSP
2	Optional	X'0815 0003'	NMVT being processed	-RSP
4	Optional	X'086F 0001'	MV length wrong for RU length	-RSP
5	Mandatory	X'080C 0005'	NMVT MV key not	-RSP

```
|          |            |               | supported          |          |
|__________|____________|_______________|____________________|__________|
```

Note: This explanation applies to tables for the individual major vectors that follow, as well as to the previous table.

The column headed "Order of Checking" in each table indicates the order in which the different checks on a request are to be made. A check marked "Optional" in the second column does not have to be done at all by a product implementing PUMS, but if it is done, then it must occur at the specified place in the sequence of checks. Checks that have the same number in the "Order of Checking" column may be done in any order. Checks that have multiple entries in this column will be done at different times for different parts of an NMVT. A length error, for example, may be detected either when scanning for the command subvector or when scanning later for a required subvector.

Checks shown as mandatory must be performed by all implementations of PUMS that receive the specified NMVT request. Those marked as optional may or may not be performed by an implementation, but if they are performed the indicated sense data must be used to report that the check has failed.

The statements of the conditions reported by the different sense data are very abbreviated in the tables. *SNA Formats* should be consulted for the full definition of each sense data.

An entry of "-RSP or X'7D'" in the "How Sent" column indicates that different implementations of PUMS may elect to send the sense data either in a negative response or in a Sense Data (X'7D') subvector within a reply NMVT. For requests that specify multiple resources, however, these sense data may be sent in a -RSP only if they apply to *all* of the resources specified on the request. If the sense data applies to some, but not all, of the specified resources, it must be sent in a X'7D' subvector along with an SNA Address List (X'04') subvector that identifies the resources to which it applies.

2.10.8 EP_XXXX Parsing of Individual Management Services Major Vectors

This section describes the general rules for parsing management services major vectors. The sections that follow it then list the individual parsing sequences for each of the management services major vectors that an EP_XXXX function set may receive. These parsing sequences begin after the common NMVT parsing, described in "Parsing of NMVTs" in topic 2.10.7.7, or the common MDS-MU parsing, described in "Common MDS-MU Parsing" in topic 2.9.2.9.2, has been completed.

Subtopics:

- 2.10.8.1 Subvector Formats
- 2.10.8.2 Parsing the Request RTM (X'8080') Major Vector
- 2.10.8.3 Parsing the Request Product Set ID (X'8090') Major Vector
- 2.10.8.4 Parsing the Change Management Major Vectors
- 2.10.8.5 Parsing the Operations Management Major Vectors

2.10.8.1 Subvector Formats

Management services subvectors fall into three categories, depending on the presence and placement of subfields within them. A *subfield*, like a subvector, consists of a length, a key, and data. Figure 10-69 depicts the three types of subvectors defined.

```
  Unformatted Subvector:

   ________ _____ ___________________
  | length | key || unformatted data  |
  |________|_____||___________________|

                    Example:  SNA Address List (X'04') MS common subvector


  Formatted Subvector:

                 |__subfield__ÿ|__subfield__ÿ| ... |__subfield__ÿ|
```

```
 ________ _____  ___ ___ ______ ___ ___ ______ _____ ___ ___ ______
| length | key || L | K | data | L | K | data | ... | L | K | data |
|________|_____||___|___|______|___|___|______|_____|___|___|______|

                    Example:  Failure Causes (X'96') Alert subvector




  Partially Formatted Subvector:

                  |___fixed size___ÿ|__subfield__ÿ|__subfield__ÿ| ... |__subfield__ÿ|
 ________ _____  __________________ ___ ___ ______ ___ ___ ______ _____ ___ ___ ______
| length | key || unformatted data | L | K | data | L | K | data | ... | L | K | data |
|________|_____||__________________|___|___|______|___|___|______|_____|___|___|______|

                    Example:  Product Identifier (X'11') MS common subvector

     Key:
          L: subfield length
          K: subfield key
```

Figure 10-69. Unformatted, Formatted, and Partially Formatted Subvectors

An *unformatted* subvector contains no subfields at all. A *formatted* subvector contains only subfields after its own length and key. A *partially formatted* subvector contains a fixed number of bytes of unformatted data after its length and key, with the remainder consisting only of subfields.

There are a few general rules for the parsing of management services request major vectors.

1. Unrecognized subvectors must be skipped over in a request major vector, but unrecognized subfields within a formatted subvector must be reported. (See Rule 3 below for a clarification of what subvectors qualify as unrecognized.)

2. The command subvector must be identified first, then the remaining subvectors, if any.

3. The subvectors that may be recognized on a request are limited by the major vector and the command subvector.

Rule 3 requires further elaboration: After the receiver has completed the common parsing, and thereby determined that the request contains a management services major vector that it is prepared to support, it searches for the *command* subvector (Rule 2). A receiver has, for each request major vector, a list of command subvector keys that it supports for that major vector. If it cannot find one of these, it reports X'080C 0006' (Command Subvector Not Recognized).

If the receiver does find a command subvector, it examines its contents to determine what *other* subvectors are required for the execution of the command. The subvectors indicated, explicitly or implicitly, in the command subvector are not merely the set of subvectors that a receiver is required to process; they are also the *only* subvectors that a receiver is *permitted* to process. Rule 1 states that a receiver skips over unrecognized subvectors in a request major vector, and acts according to the remaining subvectors in the request that it does recognize. The set of additional subvectors that a receiver is *allowed* to recognize on a request, however, is limited by the command subvector. If, for example, the command subvector in a request indicates, explicitly or implicitly, that the X'04' subvector is not included, then a receiver *must* ignore a X'04' that was included erroneously, even though it can in general recognize and process a X'04' subvector.

To summarize, a receiver looks first for a command subvector, then for other subvectors indicated by the command subvector. Any other subvectors that may be present *must* be ignored.

Sense data X'080C 0006' (Command Subvector Not Recognized) is not, as it might appear, simply a variant of X'086C nn00' (Required Subvector Missing). If a request major vector supports several different command subvectors, any one of which is valid, then no *one* of them is individually required, and so sense data X'086C nn00' is not appropriate for reporting the case in which none of the possible command subvectors is recognized. (If this sense data were used, there would be no non-arbitrary way to select the key nn of "the" missing subvector.) Instead, sense data X'080C 0006' is used to report that no appropriate command subvector was

recognized, even though the major vector key indicates that one must be present.

2.10.8.2 Parsing the Request RTM (X'8080') Major Vector

Since the Request RTM major vector is transported only in an NMVT on the SSCP-PU session, never in an MDS-MU on the CP-CP and LU-LU sessions used by MDS, the question of an MDS-MU error-reporting format does not arise for it.

Table 10-48. Sense Data Returned by PUMS after Receipt of a Request RTM (X'8080') Major Vector

Order of Checking	Mandatory / Optional	Sense Data	Condition	How Sent
1	Mandatory	X'080C 0006'	Command subvector not recognized	-RSP
2	Mandatory[1]	X'086C nn00'	Required subvector missing	-RSP
2	Mandatory[2]	X'086C 0400'	X'04' subvector not first	-RSP
1,2	Optional	X'086F 0002'	Length error	-RSP
3	Mandatory[3]	X'1005 0001'	Wrong address type	-RSP
3	Mandatory[3]	X'080C 000A'	Multiple addresses not supported by receiver	-RSP
3	Mandatory	X'0870 nnxx'	Invalid value (unformatted subvector)	-RSP
1,2,3	Optional	X'086F nn05'	Length error	-RSP
4	Mandatory[3]	X'0806 0001'	Resources unknown	-RSP or X'7D'

Notes:

1. The only value allowed is nn = X'94', indicating the RTM Control subvector. This check is mandatory if the RTM Request (X'92') subvector indicates that the X'94' subvector is present.

2. This check is mandatory if the RTM Request (X'92') subvector indicates that the SNA Address List (X'04') subvector is present.

3. This check is mandatory if the SNA Address List (X'04') subvector is present.

Sense data X'0870 nn00' is sent for an invalid value in an unformatted subvector. For example, if the RTM measurement definition X'04'is not supported, sense data X'0870 9408' is used (byte 8 of X'94' subvector is in error).

2.10.8.3 Parsing the Request Product Set ID (X'8090') Major Vector

Table 10-49. Sense Data Returned by PUMS After Receipt of a Request Product Set ID (X'8090') Major Vector

Order of Checking	Mandatory / Optional	Sense Data	Condition	How Sent
1	Mandatory	X'080C 0006'	Command subvector not recognized	-RSP
1	Optional	X'086F 0002'	Length error	-RSP
1,2	Optional[1]	X'086F nn05'	Length error	-RSP

Notes:

1. The only values allowed are nn = X'81' and nn = X'83' indicating the command subvectors for the Request Product Set ID (X'8090') major vector.

2.10.8.4 Parsing the Change Management Major Vectors

Parsing errors for the change management major vectors are reported using a variant of MDS-MU error-reporting Format 2, described in "Building Negative Replies for MDS-MUs" in topic 2.9.2.9.4: The SNA Condition Report is still enveloped by a CP-MSU, but the CP-MSU is transported as a SNA/DS agent object rather than as the application program GDS variable in an MDS-MU.

Table 10-50. Sense Data Returned by MS After Receipt of a Request Change Control (X'8050') Major Vector

Order of Checking	Mandatory / Optional	Sense Data	Condition	Alert Sent
1	Optional	X'080C 000B'	Automatic delayed acceptance not supported	No
1	Optional	X'080C 000D'	Post-test not supported	No
1	Optional	X'080C 000E'	Prohibition of automatic removal not supported	No
1	Optional	X'080C 0014'	Execution window timing not supported	No
1	Optional	X'084C 0008'	Volume not mounted	No
2	Mandatory	X'0838 0014'	Precluded combination of Install parameters	No
2	Mandatory	X'1005 0007'	SNA/MS command incompatible with SNA/FS server instruction	Yes
2	Optional	X'0838 0008'	Pre-test not applicable	No
2	Optional	X'0838 0009'	Execution window timing not applicable	No
2	Optional	X'0838 000A'	Automatic removal not applicable	No
2	Optional	X'0838 000B'	Post-test not applicable	No
2	Optional	X'0838 000C'	Automatic delayed acceptance not applicable	No
3	Mandatory	X'0838 0001'	State exception	No
3	Mandatory	X'0838 0002'	State exception	No
3	Mandatory	X'0838 0003'	State exception	No
3	Mandatory	X'0838 0004'	State exception	No
3	Mandatory	X'0838 0005'	State exception	No
3	Mandatory	X'0838 0006'	State exception	No
3	Mandatory	X'0838 0007'	State exception	No
3	Mandatory	X'0838 000D'	State exception	No

3	Mandatory	X'0838 000E'	State exception	No
3	Mandatory	X'0838 0011'	State exception	No
3	Mandatory	X'0838 0012'	State exception	No
3	Mandatory	X'0838 0013'	State exception	No
3	Mandatory	X'0838 0015'	State exception	No
3	Mandatory	X'0838 0016'	State exception	No
3	Mandatory	X'081D 0001'	NETID.LUNAME in token string does not identify the receiver	No
3	Optional	X'0838 000F'	State exception	No
3	Optional	X'0838 0010'	State exception	No

Table 10-51. Sense Data Returned by MS After Receipt of a Request Activation (X'8066') Major Vector

Order of Checking	Mandatory / Optional	Sense Data	Condition	Alert Sent
1	Mandatory	X'0872 0001'	Sessions are active	No
1	Optional	X'080C 000F'	Production-only activation not supported from focal point	No
1	Optional	X'080C 0015'	Change management activation use not supported	No
1	Optional	X'084C 0008'	Volume not mounted	No

Table 10-52. Sense Data Returned by MS After Receipt of a Request Cancelation (X'8076') Major Vector

Order of Checking	Mandatory / Optional	Sense Data	Condition	Alert Sent
1	Mandatory	X'0838 0017'	UOW has proceeded too far to cancel	No
1	Mandatory	X'0838 0018'	Cancel would dissrupt entry point resources	No
1	Mandatory	X'085B 0005'	UOW unknown or already completed	No

When a common operations services major vector is received in an MDS-MU on a CP-CP or LU-LU session used by MDS, all parsing errors are reported using MDS-MU error-reporting Format 2, described in "Building Negative Replies for MDS-MUs" in topic 2.9.2.9.4. The "How Sent" column in the table below applies only if the major vector is received in an NMVT.

Table 10-53. Sense Data Returned by PUMS after Receipt of a Common Operations Services Major Vector

Order of Checking	Mandatory / Optional	Sense Data	Condition	How Sent
Sense Data Created by EP_COMMON_OPERATIONS_SERVICES:				

1	Mandatory	X'086C nn00'[1]	Required subvector missing	-RSP or X'7D'
2	Mandatory	X'086D nnmm'[2]	Required subfield missing	-RSP or X'7D'
3	Mandatory	X'8018 0001'	Specified application unknown	-RSP or X'7D'
Sense Data Created by a Served Network Management Application[3]:				
--	--	X'084B 0003'	Application not available	X'7D'
--	--	X'1003 000D'	Request not supported by application	X'7D'
--	--	X'081C 0n0m'4	Execution error	X'7D'
--	--	X'086F 0001'	Invalid major vector length	X'7D'
--	--	X'086D nnmm'5	Required subfield missing	X'7D'
--	--	X'080C 0006'	Command subvector not recognized	X'7D'
--	--	X'080C nn00'6	Required subvector missing	X'7D'
--	--	X'0806 0001'	Resource unknown	X'7D'
--	--	X'086A nnmm'7	Subfield key invalid	X'7D'
--	--	X'086B nnmm'8	Subfield value invalid	X'7D'
--	--	X'086F nn05'9	Subvector length error	X'7D'
--	--	X'086F nn06'[10]	Subfield length error	X'7D'

Notes:

1. The only value allowed is nn = X'06', indicating the Name List subvector.

2. The only values allowed are nn = X'06', mm= X'50', indicating the Destination Application Name subfield in the Name List subvector.

3. Neither an order of checking nor a set of mandatory checks is specified for served applications.

4. The digits n and m are used as follows:

   - n represents the link connection status:

     n = X'A': The link connection status has not changed from the state previous to the execution.

     n = X'B': The link connection status was modified from the state existing previous to the execution.

   - m identifies the nature of the execution error:

     m = X'1': volatile storage error

     m = X'2': nonvolatile storage (e.g., file access error)

     m = X'3': link connection component (e.g., modem) interface error

m = X'4': unspecified software error.

5. The only values allowed are nn = X'06', mm= X'01', indicating the Resource Identifier subfield in the Name List subvector.

6. The only values allowed are nn = X'31' and nn = X'80', indicating, respectively, the Self-Defining Text Message subvector and the Test Setup Data subvector. Note that even though the architecture does not treat X'31' as a required subvector, some served network management applications do.

7. The key of the subvector containing the invalid subfield is identified in nn, and the key of the invalid subfield is identified in mm.

8. The key of the subvector containing the subfield with the invalid value is identified in nn, and the key of the subfield with the invalid value is identified in mm.

9. The key of the subvector with the invalid length is identified in nn.

10. The key of the subvector containing the subfield with the invalid length is identified in nn.

2.10.8.5 Parsing the Operations Management Major Vectors

All operations management parsing errors are reported using the SNA Condition Report (X'1532') GDS variable and MDS-MU error-reporting Format 2, described in "Building Negative Replies for MDS-MUs" in topic 2.9.2.9.4. See "Operations Management Use of the SNA Condition Report (SNACR)" in topic 1.7.3.1 for more information.

3.0 Part 3. Detailed Reference Material

A.0 Appendix A. Alerts Defined for Specific Environments

This appendix contains several groups of Alerts defined for specific environments. For each Alert, the following information is provided:

- The 32-bit Alert ID Number, calculated by means of the algorithm described in "Identification of Unique Alerts" in topic 2.10.1.19.3
- The Alert Type
- The Alert Description
- The Probable Causes
- The User, Install, and Failure Causes (as appropriate), together with their associated qualifiers
- A partial list of the additional subvectors to be included in the Alert; required subvectors such as the Product Set ID are not listed

For certain Alerts, the Recommended Actions associated with the User, Install, and Failure Causes are also included. For other Alerts, the actions cannot be specified, since they are dependent on characteristics of the particular Alert sender, e.g., how the product is serviced, whether it is attended, or whether it is locally or remotely attached to the Alert receiver.

Many of the Alerts contain one or more instances of the Detailed Data (X'82') subfield. Since it provides additional function, the Detailed Data Extended (X'85') subfield should be used instead of the X'82' subfield when possible. The X'85' and X'82' subfields are mutually exclusive in a subvector. For more information on these subfields see "The Detailed Data Extended (X'85') and Detailed Data (X'82') Common Subfields" in topic 2.10.1.19.5.

Subtopics:

- A.1 Alerts for Local Area Networks
- A.2 SDLC/LAN LLC Alerts
- A.3 Alerts for Switched Link Connections
- A.4 Alerts for X.25 Link Connections
- A.5 Alerts for APPN Nodes
- A.6 LU6.2 Alerts

A.1 Alerts for Local Area Networks

Subtopics:

- A.1.1 Token-Ring LAN Alerts
- A.1.2 CSMA/CD LAN Alerts
- A.1.3 Bridged LAN Alerts

A.1.1 Token-Ring LAN Alerts

This section documents the Alerts that should be sent by LAN managers and boundary-function-attached type 2.1 nodes. The following table defines which Alerts are sent by which nodes.

A "Yes" in a column means that the Alert with this number is always sent by this type of node. A "No" in a column means that the Alert with this number is never sent by this type of node.

Table A-1. Token-Ring Alert Sending Products

Alert Number	LAN Manager	BF-attached Type 2.1 Nodes
01	Note 1	Note 1
02	Note 1	Note 1
03	Note 1	Note 1
04	Note 1	Note 1
05	Note 1	Note 1
06	Yes	Yes
07	Yes	Yes
08	Yes	Yes
09	Note 3	Note 2
10	Note 3	No
11	No	Note 2
12	Note 3	Note 2
13	Note 3	Note 2

Notes:

1. This Alert flows if the sending product is unattended at the time of the error.

2. This Alert flows if a token-ring LAN manager is not present in the LAN to report errors on this ring.

3. If there are several token-ring LAN managers in a LAN, only one sends this Alert for each ring.

Subtopics:

- A.1.1.1 Token-Ring LAN Alert 1
- A.1.1.2 Token-Ring LAN Alert 2
- A.1.1.3 Token-Ring LAN Alert 3
- A.1.1.4 Token-Ring LAN Alert 4
- A.1.1.5 Token-Ring LAN Alert 5
- A.1.1.6 Token-Ring LAN Alert 6
- A.1.1.7 Token-Ring LAN Alert 7
- A.1.1.8 Token-Ring LAN Alert 8

- A.1.1.9 Token-Ring LAN Alert 9
- A.1.1.10 Token-Ring LAN Alert 10
- A.1.1.11 Token-Ring LAN Alert 11
- A.1.1.12 Token-Ring LAN Alert 12
- A.1.1.13 Token-Ring LAN Alert 13

A.1.1.1 Token-Ring LAN Alert 1

Alert Condition:

The adapter detected a problem on its lobe during the wrap-test portion of the insertion process. The insertion process did not complete.

Alert ID Number		X'55BF3E1C'
Alert Type	X'01'	Permanent
Alert Description	X'3211'	Open Failure
Probable Causes	X'3702'	Token-ring lobe
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3320'	Local token-ring adapter
	X'3711'	Local access unit
	X'3434'	Local lobe cables
Actions	X'1009'	Attempt to re-open the adapter after 30 seconds
	X'3301'	If problem persists then do the following:
	X'2010'	Review link detailed data
	X'3101'	Contact token-ring administrator responsible for this LAN
	X'32C0'	Report the following:
	X'82' SF	(Adapter Number)
	X'82' SF	(Error Code)
Additional SVs	X'51' SV	LAN LCS Data
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)

A.1.1.2 Token-Ring LAN Alert 2

Alert Condition:

The adapter detected a beaconing condition on the ring during the insertion process. The insertion process did not complete.

Alert ID Number		X'CAF3C58A'

Alert Type	X'01'	Permanent
Alert Description	X'3211'	Open Failure
Probable Causes	X'3703'	Token-ring fault domain
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3703'	Token-ring fault domain
Actions	X'1009'	Attempt to re-open the adapter after 30 seconds
	X'3301'	If problem persists then do the following:
	X'2010'	Review link detailed data
	X'3101'	Contact token-ring administrator responsible for this LAN
	X'32C0'	Report the following:
	X'82' SF	(Adapter Number)
	X'82' SF	(Error Code)
Additional SVs	X'51' SV	LAN LCS Data
	X'06' SF	Token-Ring Fault Domain Description
	X'26' SF	Fault Domain Names (Optional)
	X'07' SF	Beacon Data
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)
		Second resource below sender:
		Name=RINGhhhh or 'UNKNOWN'
		Type=X'2E' (RING)

A.1.1.3 Token-Ring LAN Alert 3

Alert Condition:

The adapter detected the presence of a station with its individual address on the ring during the insertion process. The insertion process did not complete.

Alert ID Number		X'D615A61E'
Alert Type	X'01'	Permanent
Alert Description	X'3211'	Open Failure
Probable Causes	X'3704'	Token-ring duplicate station address

User Causes	(none)	
Install Causes	X'3704'	Token-ring duplicate station addresses assigned
Actions	X'2010'	Review link detailed data
	X'3101'	Contact token-ring administrator responsible for this LAN
	X'32C0'	Report the following:
	X'82'SF	(Adapter Number)
	X'82'SF	(Error Code)
Failure Causes	(none)	
Additional SVs	X'51' SV	LAN LCS Data
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)

A.1.1.4 Token-Ring LAN Alert 4

Alert Condition:

The adapter received a Remove Ring Station MAC frame during the insertion process. The insertion process did not complete.

Alert ID Number		X'44D1AD86'
Alert Type	X'01'	Permanent
Alert Description	X'3211'	Open Failure
Probable Causes	X'3705'	Token-ring remove command received
User Causes	X'7101'	Token-ring remove adapter command received
Actions	X'2010'	Review link detailed data
	X'3101'	Contact token-ring administrator responsible for this LAN
	X'32C0'	Report the following
	X'82'SF	(Adapter Number)
	X'82'SF	(Error Code)
Install Causes	(none)	

Failure Causes	(none)	
Additional SVs	X'51' SV	LAN LCS Data
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)

A.1.1.5 Token-Ring LAN Alert 5

Alert Condition:

An error was detected during the insertion process that is not defined in "Token-Ring LAN Alert 1," "Token-Ring LAN Alert 2," "Token-Ring LAN Alert 3," or "Token-Ring LAN Alert 4." These conditions are not expected to occur, so they are included within one Alert definition. The insertion process did not complete.

Alert ID Number		X'016E5F4E'
Alert Type	X'01'	Permanent
Alert Description	X'3211'	Open failure
Probable Causes	X'3702'	Token-ring lobe
	X'3701'	Token-ring LAN component
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3712'	Local token-ring lobe
	X'3701'	Token-ring LAN component
	X'2600'	Interference
Actions	X'2010'	Review link detailed data
	X'3101'	Contact token-ring administrator responsible for this LAN
	X'32C0'	Report the following:
	X'82'SF	(Adapter Number)
	X'82'SF	(Error Code)
Additional SVs	X'51' SV	LAN LCS Data
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)
		Second resource below sender:
		Name=RINGhhhh or 'UNKNOWN'
		Type=X'2E' (RING)

A.1.1.6 Token-Ring LAN Alert 6

Alert Condition:

The reporting station's adapter detected a wire-fault condition on the ring.

Alert ID Number		X'A676B230'
Alert Type	X'01'	Permanent
Alert Description	X'3212'	Wire fault
Probable Causes	X'3702'	Token-ring lobe
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3711'	Local access unit
	X'3434'	Local lobe cables
	X'3320'	Local token-ring adapter
Actions	X'2010'	Review link detailed data
	X'3101'	Contact token-ring administrator responsible for this LAN
	X'0105'	Request verification of management server reporting
		links (1)
	X'32C0'	Report the following:
	X'82'	(Adapter Number)
	X'82'	(Error Code)
Additional SVs	X'51' SV	LAN LCS Data
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)
		Second resource below sender:
		Name=RINGhhhh (Hex ring ID)
		Type=X'2E' (RING)

Notes:

1. This code point is present if the sending product is a LAN manager and has reporting links with remote management servers.

A.1.1.7 Token-Ring LAN Alert 7

Alert Condition:

The reporting station's adapter has left the ring as part of the beacon automatic-recovery process. That is, the reporting station's adapter was a member of the beacon fault domain and removed itself from the ring to perform a self test, which was unsuccessful.

Alert ID Number		X'EB61E14F'
Alert Type	X'01'	Permanent
Alert Description	X'3213'	Auto-removal
Probable Causes	X'3702'	Token-ring lobe
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3320'	Local token-ring adapter
	X'3711'	Local access unit
	X'3434'	Local lobe cables
Actions	X'2010'	Review link detailed data
	X'3101'	Contact token-ring administrator responsible for this LAN
	X'0105'	Request verification of management server reporting
		links (1)
	X'32C0'	Report the following:
	X'82'	(Adapter Number)
	X'82'	(Error Code)
Additional SVs	X'51' SV	LAN LCS Data
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)
		Second resource below sender:
		Name=RINGhhhh (Hex ring ID)
		Type=X'2E' (RING)

Notes:

1. This code point is present if the sending product is a LAN manager and has reporting links with remote management servers.

A.1.1.8 Token-Ring LAN Alert 8

Alert Condition:

The reporting station's adapter received a Remove Adapter command from a LAN manager and, as a result, left the LAN.

Alert ID Number		X'59F32622'
Alert Type	X'01'	Permanent
Alert Description	X'3214'	Remove adapter command received
Probable Causes	X'7013'	LAN manager operator
User Causes	X'7101'	Token-ring remove adapter command received
Actions	X'2010'	Review link detailed data
	X'3101'	Contact token-ring administrator responsible for this LAN
	X'0105'	Request verification of management server reporting
		links (1)
	X'32C0'	Report the following:
	X'82'	(Adapter Number)
	X'82'	(Error Code)
Install Causes	(none)	
Failure Causes	(none)	
Additional SVs	X'51' SV	LAN LCS Data
	X'02' SF	Ring/Segment Identifier
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)

Notes:

1. This code point is present if the sending product is a LAN manager and has reporting links with remote management servers.

A.1.1.9 Token-Ring LAN Alert 9

Alert Condition:

The ring has been beaconing for a time longer than the hard-error detection timer. Manual intervention is necessary to recover the ring.

Alert ID Number		X'2102FCEB'
Alert Type	X'01'	Permanent
Alert Description	X'3215'	Token-ring inoperative

Probable Causes	X'3703'	Token-ring fault domain
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3703'	Token-ring fault domain
Actions	X'2010'	Review link detailed data
	X'3101'	Contact token-ring administrator responsible for this LAN
	X'0105'	Request verification of management server reporting
		links (1)
	X'32A0'	Report the following:
	X'82'SF	(Ring Status)
Additional SVs	X'51' SV	LAN LCS Data
	X'06' SF	Token-Ring Fault Domain Description
	X'26' SF	Fault Domain Names (Optional)
	X'07' SF	Beacon Data
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)
		Second resource below sender:
		Name=RINGhhhh (Hex ring ID)
		Type=X'2E' (RING)

Notes:

1. This code point is present if the sending product is a LAN manager and has reporting links with remote management servers.

A.1.1.10 Token-Ring LAN Alert 10

Alert Condition:

The ring was in a beaconing condition for a time shorter than the hard-error detection timer. When the stations in the beacon fault domain were queried, one or both of them had left the ring.

Alert ID Number		X'698CCD51'
Alert Type	X'01'	Permanent
Alert Description	X'3213'	Auto removal
Probable Causes	X'3714'	Remote token-ring lobe

User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3321'	Remote token-ring adapter
	X'3713'	Remote access unit
	X'3435'	Remote lobe cables
Actions	X'2010'	Review link detailed data
	X'3101'	Contact token-ring administrator responsible for this LAN
	X'0105'	Request verification of management server reporting
		links (1)
Additional SVs	X'51' SV	LAN LCS Data
	X'06' SF	Fault Domain Description (CP) (2)
	X'26' SF	Fault Domain Names (CP) (3)
	X'08' SF	Single Individual MAC Address (CP) (4)
	X'28' SF	Single Individual MAC Name (CP) (5)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)
		Second resource below sender:
		Name=RINGhhhh (Hex ring ID)
		Type=X'2E' (RING)

Notes:

1. This code point is present if the sending product is a LAN manager and has reporting links with remote management servers.

2. This subfield is present if the sending product has determined that both beacon fault domain stations left the ring as part of the automatic recovery process.

3. This subfield is optionally present, but only present if the sending product has determined that both beacon fault domain stations left the ring as part of the automatic recovery process.

4. This subfield is present if the sending product has determined that only one of the beacon fault domain stations left the ring as part of the automatic recovery process.

5. This subfield is optionally present, but only present if the sending product has determined only one of the beacon fault domain stations left the ring as part of the automatic recovery process.

A.1.1.11 Token-Ring LAN Alert 11

Alert Condition:

The ring was in a beaconing condition for less than 52 seconds and then recovered. The sender of this Alert either knows that neither station in the fault domain left the ring, or has no knowledge about whether a station removed itself from the ring in order to bypass the fault.

Alert ID Number		X'2F36696E'
Alert Type	X'01'	Permanent
Alert Description	X'3216'	Token-ring temporary error
Probable Causes	X'3703'	Token-ring fault domain
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3703'	Token-ring fault domain
Actions	X'2010'	Review link detailed data
	X'3101'	Contact token-ring administrator responsible for this LAN
	X'0105'	Request verification of management server reporting
		links (1)
Additional SVs	X'51' SV	LAN LCS Data
	X'06' SF	Token-Ring Fault Domain Description
	X'26' SF	Fault Domain Names (Optional)
	X'07' SF	Beacon Data
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)
		Second resource below sender:
		Name=RINGhhhh (Hex ring ID)
		Type=X'2E' (RING)

Notes:

1. This code point is present if the sending product is a LAN manager and has reporting links with remote management servers.

A.1.1.12 Token-Ring LAN Alert 12

Alert Condition:

The ring error monitor (REM) has detected excessive soft errors for the ring.

Alert ID Number		X'A9998C16'
Alert Type	X'11'	Impending problem
Alert Description	X'4001'	Excessive token-ring errors

Probable Causes	X'3703'	Token-ring fault domain
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3703'	Token-ring fault domain
Actions	X'2010'	Review link detailed data
	X'3101'	Contact token-ring administrator responsible for this LAN
Additional SVs	X'51' SV	LAN LCS Data
	X'06' SF	Token-Ring Fault Domain Description
	X'26' SF	Fault Domain Names (Optional)
	X'09' SF	Fault Domain Error Weight Pair
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)
		Second resource below sender:
		Name=RINGhhhh (Hex ring ID)
		Type=X'2E' (RING)

A.1.1.13 Token-Ring LAN Alert 13

Alert Condition:

The ring error monitor (REM) has detected that an adapter is experiencing excessive congestion and is discarding a significant number of frames.

Alert ID Number		X'57D16A21'
Alert Type	X'03'	Performance
Alert Description	X'5011'	Communication overrun
Probable Causes	X'3223'	Token-ring adapter interface
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1022'	Communication program
	X'3324'	Token-ring adapter
Actions	X'2010'	Review link detailed data

	X'3101'	Contact token-ring administrator responsible for this LAN
Additional SVs	X'51' SV	LAN LCS Data
	X'02' SF	Ring or Bus Identifier
	X'08' SF	Single Individual MAC Address
	X'28' SF	Single Individual MAC Name
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)

A.1.2 CSMA/CD LAN Alerts

This section documents the Alerts that should be sent by LAN managers and boundary-function attached T2.1 nodes for problems detected on CSMA/CD LANs. The following table defines which Alerts are sent by LAN managers and boundary-function attached T2.1 nodes.

A 'Yes' in a column means that the Alert with this number is always sent by this type of station. A 'No' in a column means that the Alert with this number is never sent by this type of station.

Table A-2. CSMA/CD LAN Alert Sending Products

Alert Number	**LAN Manager**	**BF-Attached Type 2.1 Nodes**
01	Note 1	Note 1
02	Note 1	Note 1
03	Note 1	Note 1
04	Note 1	Note 1
05	Yes	Yes
06	Note 3	Note 2
07	Yes	Yes
08	Yes	Yes
09	Yes	Yes
10	Yes	No
11	Yes	No

Notes:

1. This Alert flows if the sending product is unattended at the time of the error.

2. This Alert flows if a CSMA/CD LAN manager is not present in the LAN to report errors on this bus.

3. If there are several CSMA/CD LAN managers in a LAN, only one sends this Alert for each bus.

Subtopics:

- A.1.2.1 CSMA/CD LAN Alert 1
- A.1.2.2 CSMA/CD LAN Alert 2
- A.1.2.3 CSMA/CD LAN Alert 3
- A.1.2.4 CSMA/CD LAN Alert 4
- A.1.2.5 CSMA/CD LAN Alert 5
- A.1.2.6 CSMA/CD LAN Alert 6
- A.1.2.7 CSMA/CD LAN Alert 7
- A.1.2.8 CSMA/CD LAN Alert 8
- A.1.2.9 CSMA/CD LAN Alert 9

- A.1.2.10 CMSA/CD LAN Alert 10
- A.1.2.11 CSMA/CD LAN Alert 11

A.1.2.1 CSMA/CD LAN Alert 1

Alert Condition:

The adapter could not detect a carrier signal on the CSMA/CD network during the insertion process. The insertion process did not complete.

Alert ID Number		X'75CB6673'
Alert Type	X'01'	Permanent
Alert Description	X'3211'	Open failure
Probable Causes	X'3436'	Local CSMA/CD LAN cable
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3436'	Local CSMA/CD LAN cable
	X'3721'	CSMA/CD LAN component
Actions	X'1320'	Check cable connection and retry
	X'2010'	Review link detailed data
	X'3102'	Contact CSMA/CD administrator responsible for this LAN
	X'32A0'	Report the following:
	X'82'SF	(Adapter Number)
Additional SVs	X'51' SV	LAN LCS Data
	X'02' SF	Ring/Bus Identifier
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)
		Second resource below sender:
		Name=CBUShhhh (Hex bus number) or 'UNKNOWN'
		Type=X'32' (CBUS)

A.1.2.2 CSMA/CD LAN Alert 2

Alert Condition:

The adapter detected the presence of a station with its address on the CSMA/CD LAN bus during the insertion process. The insertion process did not complete.

Alert ID Number		X'DD8A0144'
Alert Type	X'01'	Permanent
Alert Description	X'3211'	Open failure
Probable Causes	X'3724'	CSMA/CD duplicate station address
User Causes	(none)	
Install Causes	X'3724'	Duplicate station address assigned
Actions	X'2010'	Review link detailed data
	X'3102'	Contact CSMA/CD administrator responsible for this LAN
	X'32A0'	Report the following:
	X'82'	(Adapter Number)
Failure Causes	(none)	
Additional SVs	X'51' SV	LAN LCS Data
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)
		Second resource below sender:
		Name=CBUShhhh (Hex bus number) or 'UNKNOWN'
		Type=X'32' (CBUS)

A.1.2.3 CSMA/CD LAN Alert 3

Alert Condition:

The adapter received a Remove Station command during the insertion process. The insertion process did not complete.

Alert ID Number		X'C6B5D6A5'
Alert Type	X'01'	Permanent
Alert Description	X'3211'	Open failure
Probable Causes	X'3725'	CSMA/CD remove adapter command received
User Causes	X'7107'	CSMA/CD remove adapter command received

Actions	X'2010'	Review link detailed data
	X'3102'	Contact CSMA/CD administrator responsible for this LAN
	X'32A0'	Report the following:
	X'82'	(Adapter Number)
Install Causes	(none)	
Failure Causes	(none)	
Additional SVs	X'51' SV	LAN LCS Data
	X'02' SF	Ring/Bus Identifier (1)
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)

Notes:

1. This subfield is present only if the station has completed the portion of the open process during which it learns the ring identifier.

A.1.2.4 CSMA/CD LAN Alert 4

Alert Condition:

An error was detected during the insertion process that is not defined in "CSMA/CD LAN Alert 1" in topic A.1.2.1, "CSMA/CD LAN Alert 2" in topic A.1.2.2, or "CSMA/CD LAN Alert 3" in topic A.1.2.3. The insertion process did not complete.

Alert ID Number		X'8B1836C5'
Alert Type	X'01'	Permanent
Alert Description	X'3211'	Open failure
Probable Causes	X'3721'	CSMA/CD LAN component
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3721'	CSMA/CD LAN component
Actions	X'1320'	Check cable connection and retry
	X'2010'	Review link detailed data
	X'3102'	Contact CSMA/CD administrator responsible for this LAN
	X'32C0'	Report the following:
	X'82'	(Adapter Number)
	X'82'	(Error Code)
Additional SVs	X'51' SV	LAN LCS Data

	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)
		Second resource below sender:
		Name=CBUShhhh (Hex Bus Name) or 'UNKNOWN'
		Type=X'32' (CBUS)

A.1.2.5 CSMA/CD LAN Alert 5

Alert Condition:

The reporting node's adapter received a Remove Adapter command from a LAN manager and, as a result, left the CSMA/CD LAN.

Alert ID Number		X'EB1D6ABB'
Alert Type	X'01'	Permanent
Alert Description	X'3214'	Remove adapter command received
Probable Causes	X'7013'	LAN manager operator
User Causes	X'7107'	CSMA/CD remove adapter command received
Actions	X'2010'	Review link detailed data
	X'3102'	Contact CSMA/CD administrator responsible for this LAN
	X'32A0'	Report the following:
	X'82'	(Adapter Number)
Install Causes	(none)	
Failure Causes	(none)	
Additional SVs	X'51' SV	LAN LCS Data
	X'02' SF	Ring/Bus Identifier
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)

A.1.2.6 CSMA/CD LAN Alert 6

Alert Condition:

A continuous-carrier condition has been detected on the CSMA/CD bus and the sending product's adapter is not the cause of the error.

Alert ID Number		X'3CDEF8F5' (1)
Alert Type	X'01'	Permanent
Alert Description	X'3220'	CSMA/CD bus inoperative
Probable Causes	X'3330'	Adapter hardware
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3323'	Remote CSMA/CD adapter
Actions	X'2010'	Review link detailed data
	X'3102'	Contact CSMA/CD administrator responsible for this LAN
Additional SVs	X'51' SV	LAN LCS Data
	X'02' SF	Ring/Bus Identifier
	X'08' SF	Single MAC Address (2)
	X'28' SF	Single MAC Name (3)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)
		Second resource below sender:
		Name=CBUShhhh (Hex bus number)
		Type=X'32' (CBUS)

Notes:

1. This Alert was previously documented with the incorrect Alert ID value of X'43AAE16E'. Alert receivers should be able to accept Alerts containing either the current, correct value or the old, incorrect value.

2. This subfield is present if the sending product has isolated the continuous carrier to a single station address (not its own).

3. This subfield is optionally present, but only present if the sending product has isolated the continuous carrier to a single station address.

A.1.2.7 CSMA/CD LAN Alert 7

Alert Condition:

A no-carrier condition was detected on the CSMA/CD bus.

Alert ID Number		X'668E036D'

Alert Type	X'01'	Permanent
Alert Description	X'3221'	CSMA/CD LAN communications lost
Probable Causes	X'3426'	CSMA/CD LAN cables
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3436'	Local CSMA/CD adapter cable
	X'3426'	CSMA/CD LAN cables
	X'3721'	CSMA/CD LAN component
Actions	X'2010'	Review link detailed data
	X'3102'	Contact CSMA/CD administrator responsible for this LAN
	X'32A0'	Report the following:
	X'82'	(Adapter Number)
Additional SVs	X'51' SV	LAN LCS Data
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)
		Second resource below sender:
		Name=CBUShhhh (Hex bus number)
		Type=X'32' (CBUS)

A.1.2.8 CSMA/CD LAN Alert 8

Alert Condition:

A continuous-carrier condition has been detected on the CSMA/CD bus and the source of the condition has been isolated to the local adapter.

Alert ID Number		X'176D9CDF'
Alert Type	X'01'	Permanent
Alert Description	X'3220'	CSMA/CD bus inoperative
Probable Causes	X'3322'	Local CSMA/CD adapter
User Causes	(none)	

Install Causes	(none)	
Failure Causes	X'3322'	Local CSMA/CD adapter
	X'3722'	CSMA/CD LAN translator unit
Actions	X'2010'	Review link detailed data
	X'3102'	Contact CSMA/CD administrator responsible for this LAN
Additional SVs	X'51' SV	LAN LCS Data
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)
		Second resource below sender:
		Name=CBUShhhh (Hex bus number)
		Type=X'32' (CBUS)

A.1.2.9 CSMA/CD LAN Alert 9

Alert Condition:

A continuous-carrier condition has been detected on the CSMA/CD bus and the source of the condition can not be isolated.

Alert ID Number		X'F7A377AE'
Alert Type	X'01'	Permanent
Alert Description	X'3220'	CSMA/CD bus inoperative
Probable Causes	X'3325'	CSMA/CD adapter
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3325'	CSMA/CD adapter
	X'3722'	CSMA/CD LAN translator unit
Actions	X'2010'	Review link detailed data
	X'3102'	Contact CSMA/CD administrator responsible for this LAN
Additional SVs	X'51' SV	LAN LCS Data
	X'02' SF	Ring or Bus Identifier
	X'05' SV	Hierarchy/Resource List

	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)
		Second resource below sender:
		Name=CBUShhhh (Hex bus number)
		Type=X'32' (CBUS)

A.1.2.10 CMSA/CD LAN Alert 10

Alert Condition:

The LAN manager has detected that an adapter is experiencing excessive congestion and is discarding a significant number of frames.

Alert ID Number		X'A48865FD' (1)
Alert Type	X'03'	Performance
Alert Description	X'5011'	Communication overrun
Probable Causes	X'3221'	CSMA/CD Adapter Interface
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1022'	Communication program
	X'3325'	CSMA/CD adapter
Actions	X'2010'	Review link detailed data
	X'3102'	Contact CSMA/CD administrator responsible for this LAN
Additional SVs	X'51' SV	LAN LCS Data
	X'02' SF	Ring or Bus Identifier
	X'08' SF	Single Individual MAC Address
	X'28' SF	Single Individual MAC Name
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)

Notes:

1. This Alert was previously documented with the incorrect Alert ID value of X'1A56BAAE'. Alert receivers should be able to accept Alerts containing either the current, correct value or the old, incorrect value.

A.1.2.11 CSMA/CD LAN Alert 11

Alert Condition:

A continuous-carrier condition has been detected on the CSMA/CD bus and the source of the condition has been isolated to a remote adapter. The remote adapter automatically removed itself from the LAN and the error has been bypassed.

Alert ID Number		X'C1A08052'
Alert Type	X'01'	Permanent
Alert Description	X'3213'	Auto removal
Probable Causes	X'3323'	Remote CSMA/CD adapter
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3325'	CSMA/CD adapter
Actions	X'2010'	Review link detailed data
	X'3102'	Contact CSMA/CD administrator responsible for this LAN
Additional SVs	X'51' SV	LAN LCS Data
	X'04' SF	Remote Individual MAC Address
	X'24' SF	Remote Individual MAC Name
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)
		Second resource below sender:
		Name=CBUShhhh (Hex bus number)
		Type=X'32' (CBUS)

A.1.3 Bridged LAN Alerts

This section defines the Alerts sent for problems associated with bridged LANs.

The Alerts described below are those sent by nodes receiving management services support on an SSCP-PU session. Alerts sent in a peer environment by nodes receiving management services support over the CP-CP and LU-LU sessions used by multiple-domain support, are the same as those described below except for the Hierarchy/Resource List (X'05') subvector. In those Alerts described below in which an X'05' is not present, the Alert sender includes an X'05' subvector containing its own CP name in the Hierarchy Name List (X'10') subfield. In those Alerts containing a X'05' subvector, the X'05' also includes as the first name in the Hierarchy Name List (X'10') subfield, the CP name of the Alert sender. In addition, in the peer environment, the hierarchy complete indicator in the Hierarchy Name List subfield is set to 0 indicating the hierarchy is complete. It is set to 1 in the hierarchical environment. These Alerts are sent only by LAN managers.

Subtopics:

- A.1.3.1 Bridged LAN Alert 1
- A.1.3.2 Bridged LAN Alert 2
- A.1.3.3 Bridged LAN Alert 3
- A.1.3.4 Bridged LAN Alert 4
- A.1.3.5 Bridged LAN Alert 5
- A.1.3.6 Bridged LAN Alert 6
- A.1.3.7 Bridged LAN Alert 7
- A.1.3.8 Bridged LAN Alert 8
- A.1.3.9 Bridged LAN Alert 9
- A.1.3.10 Bridged LAN Alert 10
- A.1.3.11 Bridged LAN Alert 11
- A.1.3.12 Bridged LAN Alert 12
- A.1.3.13 Bridged LAN Alert 13
- A.1.3.14 Bridged LAN Alert 14
- A.1.3.15 Bridged LAN Alert 15
- A.1.3.16 Bridged Alert 16

A.1.3.1 Bridged LAN Alert 1

Alert Condition:

An abnormally large percentage of frames are being discarded at a bridge. The bridge server has calculated the ratio of the number of frames it discarded to the number of frames it copied for forwarding and this ratio exceeded a threshold. Only the controlling LAN manager sends this Alert.

Alert ID Number		X'EE64FB52'
Alert Type	X'01'	Permanent
Alert Description	X'4010'	Error-to-traffic ratio exceeded
Probable Causes	X'3740'	LAN bridge
User Causes	X'7109'	LAN bridge operator took bridge offline
	X'710A'	LAN manager operator took bridge offline
Install Causes	(none)	
Failure Causes	X'3700'	LAN component
	X'3741'	Congestion in LAN bridge
	X'2007'	LAN communications error
Actions	X'2010'	Review link detailed data
	X'3103'	Contact LAN administrator responsible for this LAN
Additional SVs	X'51' SV	LAN LCS Data
	X'0A' SF	Bridge Identifier
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)
		Second resource below sender:
		Name=(bridge name)
		Type=X'3A' (BRDG)

A.1.3.2 Bridged LAN Alert 2

Alert Condition:

The LAN bridge was taken off-line by an operator. Shut-down was orderly and the LAN bridge server issued a frame warning the LAN managers that it was being removed from the LAN. Bridge frame forwarding functions were terminated. That is, either an operator at the bridge or at a LAN manager issued a Set Bridge Parameters frame to set the Route Active parameter of the bridge server to cause the bridge not to forward frames. Only the controlling LAN manager sends this Alert.

Alert ID Number		X'355A1EC7' (1)
Alert Type	X'01'	Permanent
Alert Description	X'B003'	LAN bridge taken offline
Probable Causes	X'7012'	LAN bridge operator
	X'7013'	LAN manager operator
User Causes	X'7109'	LAN bridge operator took bridge offline
	X'710A'	LAN manager operator took bridge offline
Actions	X'2010'	Review link detailed data
	X'3103'	Contact LAN administrator responsible for this LAN
Install Causes	(none)	
Failure Causes	(none)	
Additional SVs	X'51' SV	LAN LCS Data
	X'0A' SF	Bridge Identifier
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)
		Second resource below sender:
		Name=(bridge name)
		Type=X'3A' (BRDG)

Notes:

1. This Alert was previously documented with the incorrect Alert ID value of X'608A29AF'. Alert receivers should be able to accept Alerts containing either the current, correct value or the old, incorrect value.

A.1.3.3 Bridged LAN Alert 3

Alert Condition:

A LAN reporting link to remote management servers has been lost. The remote link station does not respond. The inactivity timer (Ti) has expired, causing the remote station to be polled. The remote station does not respond to the poll.

The controlling LAN manager sends this Alert when it loses reporting links and it's local ring was not in a beaconing condition for a pre-determined period before the link was lost. The default for this pre-determined period is 1 minute, but it is configurable. Only the controlling LAN manager sends this Alert.

Alert ID Number		X'841206FE'
Alert Type	X'01'	Permanent
Alert Description	X'3330'	Management server reporting link error
Probable Causes	X'2107'	LAN LLC communications/remote node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2107'	LAN LLC communications/remote node
	X'F017'	Poll count exhausted
Actions	X'2010'	Review link detailed data
	X'3103'	Contact LAN administrator responsible for this LAN
	X'32C0'	Report the following
	X'82' SF	(Adapter Number)
	X'82' SF	(Reference Code)
Additional SVs	X'51' SV	LAN LCS Data
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'04' SF	Remote Individual MAC Address
	X'24' SF	Remote Individual MAC Name (Optional)
	X'05' SF	LAN Routing Information (1)
	X'52' SV	LCS configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count

	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)
		Second resource below sender:(2)
		Name=(bridge name)
		Type=X'3A' (BRDG)
		Second resource below sender:(3)
		Name=(management server name)
		Type=X'3C' (MSVR)

Notes:

1. This subfield is present if the lost link traversed a LAN bridge.

2. This subfield is present if the management servers with which the link was lost were located in a LAN bridge.

3. This subfield is present if the management servers with which the link was lost were not located in a LAN bridge.

A.1.3.4 Bridged LAN Alert 4

Alert Condition:

A LAN reporting link to remote management servers has been lost. The remote link station sent a Disconnect Mode to the local link station. The LAN manager tried to re-establish the link after a pre-determined time and the attempt was unsuccessful. Only the controlling LAN manager sends this Alert.

Alert ID Number		X'B1D9A4C5'
Alert Type	X'01'	Permanent
Alert Description	X'3300'	Management server reporting link error
Probable Causes	X'2007'	LAN LLC communications
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2007'	LAN LLC communications
	X'F01A'	DM received
Actions	X'2010'	Review link detailed data
	X'3103'	Contact LAN administrator responsible for this LAN
	X'32C0'	Report the following
	X'82' SF	(Adapter Number)
	X'82'	(Reference Code)

	SF	
Additional SVs	X'51' SV	LAN LCS Data
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'04' SF	Remote Individual MAC Address
	X'24' SF	Remote Individual MAC Name (Optional)
	X'05' SF	LAN Routing Information (1)
	X'52' SV	LCS configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)
		Second resource below sender:(2)
		Name=(bridge name)
		Type=X'3A' (BRDG)
		Second resource below sender:(3)
		Name=(management server name)
		Type=X'3C' (MSVR)

Notes:

1. This subfield is present if the lost link traversed a LAN bridge.

2. This subfield is present if the management servers with which the link was lost were located in a LAN bridge.

3. This subfield is present if the management servers with which the link was lost were not located in a LAN bridge.

A.1.3.5 Bridged LAN Alert 5

Alert Condition:

A LAN reporting link to remote management servers has been lost. The local link station sent an invalid or unsupported frame to the remote link station. This resulted in the remote link station returning a Frame Reject. The LAN manager tried to re-establish the link after a pre-determined time and the attempt was unsuccessful.

Alert ID Number		X'8A5B2D2C'
Alert Type	X'01'	Permanent
Alert Description	X'2100'	Software program error
Probable Causes	X'2007'	LAN LLC communications
	X'1000'	Software program
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1000'	Software program
	X'F010'	Frame reject received:
		Invalid/unsupported command or response sent
Actions	X'3301'	If problem persists then do the following:
	X'2010'	Review link detailed data
	X'3103'	Contact LAN administrator responsible for this LAN
	X'32C0'	Report the following
	X'82' SF	(Adapter Number)
	X'82' SF	(Reference Code)
Additional SVs	X'51' SV	LAN LCS Data
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'04' SF	Remote Individual MAC Address
	X'24' SF	Remote Individual MAC Name (Optional)
	X'05' SF	LAN Routing Information (1)
	X'52' SV	LCS configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)

		Type=X'39' (LAN)
		Second resource below sender:(2)
		Name=(bridge name)
		Type=X'3A' (BRDG)
		Second resource below sender:(3)
		Name=(management server name)
		Type=X'3C' (MSVR)

Notes:

1. This subfield is present if the lost link traversed a LAN bridge.

2. This subfield is present if the management servers with which the link was lost were located in a LAN bridge.

3. This subfield is present if the management servers with which the link was lost were not located in a LAN bridge.

A.1.3.6 Bridged LAN Alert 6

Alert Condition:

A LAN reporting link to remote management servers has been lost. The local link station sent an I-frame when not permitted to the remote link station. This resulted in the remote link station returning a Frame Reject. The LAN manager tried to re-establish the link after a pre-determined time and the attempt was unsuccessful.

Alert ID Number		X'8E9A309B'
Alert Type	X'01'	Permanent
Alert Description	X'2100'	Software program error
Probable Causes	X'2007'	LAN LLC communications
	X'1000'	Software program
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1000'	Software program
	X'F011'	Frame reject received:
		I-frame sent when not permitted
Actions	X'3301'	If problem persists then do the following:
	X'2010'	Review link detailed data
	X'3103'	Contact LAN administrator responsible for this LAN
	X'32C0'	Report the following
	X'82' SF	(Adapter Number)
	X'82' SF	(Reference Code)

Additional SVs	X'51' SV	LAN LCS Data
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'04' SF	Remote Individual MAC Address
	X'24' SF	Remote Individual MAC Name (Optional)
	X'05' SF	LAN Routing Information (1)
	X'52' SV	LCS configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)
		Second resource below sender:(2)
		Name=(bridge name)
		Type=X'3A' (BRDG)
		Second resource below sender:(3)
		Name=(management server name)
		Type=X'3C' (MSVR)

Notes:

1. This subfield is present if the lost link traversed a LAN bridge.

2. This subfield is present if the management servers with which the link was lost were located in a LAN bridge.

3. This subfield is present if the management servers with which the link was lost were not located in a LAN bridge.

A.1.3.7 Bridged LAN Alert 7

Alert Condition:

A LAN reporting link to remote management servers has been lost. The local link station sent a frame with an invalid N(r). This resulted in the remote link station returning a Frame Reject. The LAN manager tried to re-establish the link after a pre-determined time and the attempt was unsuccessful.

Alert ID Number		X'83D91642'
Alert Type	X'01'	Permanent
Alert Description	X'2100'	Software program error
Probable Causes	X'2007'	LAN LLC communications
	X'1000'	Software program
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1000'	Software program
	X'F012'	Frame reject received: invalid N(r) sent
Actions	X'3301'	If problem persists then do the following:
	X'2010'	Review link detailed data
	X'3103'	Contact LAN administrator responsible for this LAN
	X'32C0'	Report the following
	X'82' SF	(Adapter Number)
	X'82' SF	(Reference Code)
Additional SVs	X'51' SV	LAN LCS Data
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'04' SF	Remote Individual MAC Address
	X'24' SF	Remote Individual MAC Name (Optional)
	X'05' SF	LAN Routing Information (1)
	X'52' SV	LCS configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)
		Second resource below

		sender:(2)
		Name=(bridge name)
		Type=X'3A' (BRDG)
		Second resource below
		sender:(3)
		Name=(management server name)
		Type=X'3C' (MSVR)

Notes:

1. This subfield is present if the lost link traversed a LAN bridge.

2. This subfield is present if the management servers with which the link was lost were located in a LAN bridge.

3. This subfield is present if the management servers with which the link was lost were not located in a LAN bridge.

A.1.3.8 Bridged LAN Alert 8

Alert Condition:

A LAN reporting link to remote management servers has been lost. The local link station sent a frame with an I-field that was too long. This resulted in the remote link station returning a Frame Reject.

Alert ID Number		X'87180BF5'
Alert Type	X'01'	Permanent
Alert Description	X'2100'	Software program error
Probable Causes	X'2007'	LAN LLC communications
	X'1000'	Software program
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1000'	Software program
	X'F013'	Frame reject received:
		Maximum I-field length exceeded
Actions	X'3301'	If problem persists then do the following:
	X'2010'	Review link detailed data
	X'3103'	Contact LAN administrator responsible for this LAN
	X'32C0'	Report the following
	X'82' SF	(Adapter Number)
	X'82' SF	(Reference Code)
Additional SVs	X'51' SV	LAN LCS Data

	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'04' SF	Remote Individual MAC Address
	X'24' SF	Remote Individual MAC Name (Optional)
	X'05' SF	LAN Routing Information (1)
	X'52' SV	LCS configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)
		Second resource below sender:(2)
		Name=(bridge name)
		Type=X'3A' (BRDG)
		Second resource below sender:(3)
		Name=(management server name)
		Type=X'3C' (MSVR)

Notes:

1. This subfield is present if the lost link traversed a LAN bridge.

2. This subfield is present if the management servers with which the link was lost were located in a LAN bridge.

3. This subfield is present if the management servers with which the link was lost were not located in a LAN bridge.

A.1.3.9 Bridged LAN Alert 9

Alert Condition:

A LAN reporting link to remote management servers has been lost. The remote link station sent an invalid or unsupported frame to the local link station. This resulted in the local link station returning a Frame Reject. The LAN manager tried to re-establish the link after a pre-determined time and the attempt was unsuccessful.

Alert ID Number		X'28EF2B5D'
Alert Type	X'01'	Permanent

Alert Description	X'2100'	Software program error
Probable Causes	X'2007'	LAN LLC communications
	X'1023'	Communications program in remote node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1023'	Communications program in remote node
	X'F020'	Invalid/unsupported command or response received
Actions	X'3301'	If problem persists then do the following:
	X'2010'	Review link detailed data
	X'3103'	Contact LAN administrator responsible for this LAN
	X'32C0'	Report the following
	X'82' SF	(Adapter Number)
	X'82' SF	(Reference Code)
Additional SVs	X'51' SV	LAN LCS Data
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'04' SF	Remote Individual MAC Address
	X'24' SF	Remote Individual MAC Name (Optional)
	X'05' SF	LAN Routing Information (1)
	X'52' SV	LCS configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)
		Second resource below sender:(2)
		Name=(bridge name)
		Type=X'3A' (BRDG)

		Second resource below sender:(3)
		Name=(management server name)
		Type=X'3C' (MSVR)

Notes:

1. This subfield is present if the lost link traversed a LAN bridge.

2. This subfield is present if the management servers with which the link was lost were located in a LAN bridge.

3. This subfield is present if the management servers with which the link was lost were not located in a LAN bridge.

A.1.3.10 Bridged LAN Alert 10

Alert Condition:

A LAN reporting link to remote management servers has been lost. The remote link station sent an I-frame when not permitted to the local link station. This resulted in the local link station returning a Frame Reject. The LAN manager tried to re-establish the link after a pre-determined time and the attempt was unsuccessful.

Alert ID Number		X'2C2E36EA'
Alert Type	X'01'	Permanent
Alert Description	X'2100'	Software program error
Probable Causes	X'2007'	LAN LLC communications
	X'1023'	Communications program in remote node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1023'	Communications program in remote node
	X'F021'	I-Frame received when not permitted
Actions	X'3301'	If problem persists then do the following:
	X'2010'	Review link detailed data
	X'3103'	Contact LAN administrator responsible for this LAN
	X'32C0'	Report the following
	X'82' SF	(Adapter Number)
	X'82' SF	(Reference Code)
Additional SVs	X'51' SV	LAN LCS Data
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)

	X'04' SF	Remote Individual MAC Address
	X'24' SF	Remote Individual MAC Name (Optional)
	X'05' SF	LAN Routing Information (1)
	X'52' SV	LCS configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)
		Second resource below sender:(2)
		Name=(bridge name)
		Type=X'3A' (BRDG)
		Second resource below sender:(3)
		Name=(management server name)
		Type=X'3C' (MSVR)

Notes:

1. This subfield is present if the lost link traversed a LAN bridge.

2. This subfield is present if the management servers with which the link was lost were located in a LAN bridge.

3. This subfield is present if the management servers with which the link was lost were not located in a LAN bridge.

A.1.3.11 Bridged LAN Alert 11

Alert Condition:

A LAN reporting link to remote management servers has been lost. The remote link station sent a frame with an invalid N(r). This resulted in the local link station returning a Frame Reject. The LAN manager tried to re-establish the link after a pre-determined time and the attempt was unsuccessful.

Alert ID Number		X'216D1033'
Alert Type	X'01'	Permanent
Alert Description	X'2100'	Software program error

Probable Causes	X'2007'	LAN LLC communications
	X'1023'	Communications program in remote node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1023'	Communications program in remote node
	X'F022'	Invalid N(r) received
Actions	X'3301'	If problem persists then do the following:
	X'2010'	Review link detailed data
	X'3103'	Contact LAN administrator responsible for this LAN
	X'32C0'	Report the following
	X'82' SF	(Adapter Number)
	X'82' SF	(Reference Code)
Additional SVs	X'51' SV	LAN LCS Data
LAN LCS Data		
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'04' SF	Remote Individual MAC Address
	X'24' SF	Remote Individual MAC Name (Optional)
	X'05' SF	LAN Routing Information (1)
	X'52' SV	LCS configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)
		Second resource below sender:(2)
		Name=(bridge name)
		Type=X'3A' (BRDG)
		Second resource below sender:(3)

		Name=(management server name)
		Type=X'3C' (MSVR)

Notes:

1. This subfield is present if the lost link traversed a LAN bridge.

2. This subfield is present if the management servers with which the link was lost were located in a LAN bridge.

3. This subfield is present if the management servers with which the link was lost were not located in a LAN bridge.

A.1.3.12 Bridged LAN Alert 12

Alert Condition:

A LAN reporting link to remote management servers has been lost. The remote link station sent a frame with an I-field that was too long. This resulted in the local link station returning a Frame Reject. The LAN manager tried to re-establish the link after a pre-determined time and the attempt was unsuccessful.

Alert ID Number		X'25AC0D84'
Alert Type	X'01'	Permanent
Alert Description	X'2100'	Software program error
Probable Causes	X'2007'	LAN LLC communications
	X'1023'	Communications program in remote node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1023'	Communications program in remote node
	X'F023'	Received I-field exceeded maximum length
Actions	X'3301'	If problem persists then do the following:
	X'2010'	Review link detailed data
	X'3103'	Contact LAN administrator responsible for this LAN
	X'32C0'	Report the following
	X'82' SF	(Adapter Number)
	X'82' SF	(Reference Code)
Additional SVs	X'51' SV	LAN LCS Data
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'04' SF	Remote Individual MAC Address
	X'24' SF	Remote Individual MAC Name

		(Optional)
	X'05' SF	LAN Routing Information (1)
	X'52' SV	LCS configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' (LAN)
		Second resource below sender:(2)
		Name=(bridge name)
		Type=X'3A' (BRDG)
		Second resource below sender:(3)
		Name=(management server name)
		Type=X'3C' (MSVR)

Notes:

1. This subfield is present if the lost link traversed a LAN bridge.

2. This subfield is present if the management servers with which the link was lost were located in a LAN bridge.

3. This subfield is present if the management servers with which the link was lost were not located in a LAN bridge.

A.1.3.13 Bridged LAN Alert 13

Alert Condition:

The remote management server has been unable to receive data on the link and has been sending Receiver Not Ready frames contiguously for more than 30 seconds.

Alert ID Number		X'6CEB625D'
Alert Type	X'01'	Permanent
Alert Description	X'5004'	Out of resources
Probable Causes	X'1023'	Communications program in remote node
	X'1031'	LAN management server

User Causes	X'710B'	User incapacitated LAN management server program
Actions	X'1405'	Reactivate LAN management server program
Install Causes	(none)	
Failure Causes	X'0020'	Excessive load on processor
	X'0111'	Number of LAN mgmt. frames received exceeds buffer capacity
	X'1031'	LAN management server
Actions	X'2010'	Review link detailed data
	X'3103'	Contact LAN administrator responsible for this LAN
Additional SVs	X'51' SV	LAN LCS Data
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'04' SF	Remote Individual MAC Address
	X'24' SF	Remote Individual MAC Name (Optional)
	X'05' SF	LAN Routing Information (1)
	X'05' SV	Hierarchy/Resource List (CP)
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' LAN
		Second resource below sender:(2)
		Name=(bridge name)
		Type=X'3A' BRDG
		Second resource below sender:(3)
		Name=(management server name)
		Type=X'3C' MSVR

Notes:

1. This subfield is present if the link about which this Alert pertains traversed a LAN bridge.

2. This subfield is present if the management servers with which the link identified in this Alert were located in a LAN bridge.

3. This subfield is present if the management servers with which the link identified in this Alert were not located in a LAN bridge.

A.1.3.14 Bridged LAN Alert 14

Alert Condition:

The remote management server has been congested and, as a result, has

probably discarded management data.

Alert ID Number		X'68882FCE'
Alert Type	X'01'	Permanent
Alert Description	X'3230'	LAN management data lost
Probable Causes	X'1031'	LAN management server
	X'1023'	Communications program in remote node
User Causes	X'710B'	User incapacitated LAN management server program
Actions	X'1405'	Reactivate LAN management server program
Install Causes	(none)	
Failure Causes	X'0020'	Excessive load on processor
	X'0111'	Number of LAN mgmt. frames received exceeds buffer capacity
	X'1031'	LAN management server
Actions	X'2010'	Review link detailed data
	X'3103'	Contact LAN administrator responsible for this LAN
Additional SVs	X'51' SV	LAN LCS Data
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'04' SF	Remote Individual MAC Address
	X'24' SF	Remote Individual MAC Name (Optional)
	X'05' SF	LAN Routing Information (1)
	X'05' SV	Hierarchy/Resource List (CP)
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' LAN
		Second resource below sender:(2)
		Name=(bridge name)
		Type=X'3A' BRDG
		Second resource below sender:(3)
		Name=(management server name)
		Type=X'3C' MSVR

Notes:

1. This subfield is present if the link about which this Alert pertains traversed a LAN bridge.

2. This subfield is present if the management servers with which the link identified in this Alert were located in a LAN bridge.

3. This subfield is present if the management servers with which the link identified in this Alert were not located in a LAN bridge.

A.1.3.15 Bridged LAN Alert 15

Alert Condition:

A LAN manager attempted to establish a reporting link with management servers, but was rejected because it used an invalid password. Only the controlling LAN manager sends this Alert.

Alert ID Number		X'91C0A9A7'
Alert Type	X'12'	Unknown
Alert Description	X'C001'	Invalid reporting link password
Probable Causes	X'7013'	Remote LAN manager operator
User Causes	X'7103'	LAN manager operator entered incorrect password
	X'7104'	Unauthorized access to LAN management server attempted
Actions	X'3301'	If problem persists then do the following:
	X'2010'	Review link detailed data
	X'3103'	Contact LAN administrator responsible for this LAN
Install Causes	(none)	
Failure Causes	(none)	
Additional SVs	X'51' SV	LAN LCS Data
	X'08' SF	Single MAC Address
	X'28' SF	Single MAC Name (Optional)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Type=X'39' LAN

A.1.3.16 Bridged Alert 16

Alert Condition:

The LAN bridge has detected an error that caused it to terminate the communication link with the LAN manager. Only the controlling LAN manager sends this Alert.

Alert ID Number		X'F431E3D4'
Alert Type	X'01'	Permanent
Alert Description	X'B003'	LAN bridge taken offline
Probable Causes	X'3340'	Local LAN adapter
	X'3600'	Modem
	X'1022'	Communications program
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3340'	Local LAN adapter
	X'3600'	Modem
	X'1031'	LAN management server
	X'3400'	Cable loose or defective
Actions	X'2010'	Review link detail data
	X'3103'	Contact LAN administrator responsible for this LAN
	X'32A0'	Report the following (1) , (2)
	X'82' SF	(Modem control line status) (3)
	X'82' SF	(X.21 state) (3) , (4)
Additional SVs	X'51' SV	LAN LCS Data
	X'0A' SF	Bridge Identifier
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(LAN name)
		Resource Type=X'39' (LAN)
		Second resource below sender:
		Name=(bridge name)
		Resource Type=X'39' (BRDG)

Notes:

1. The 'Report the following' sequence is only present if communication with a split bridge has been lost.

2. The 'Report the following' code point will be X'32C0' if the split bridge interface is X.21.

3. New code point to be assigned.

4. This field is present if the split bridge interface is X.21.

A.2 SDLC/LAN LLC Alerts

Subtopics:

- A.2.1 LAN LLC Alerts
- A.2.2 SDLC Alerts

A.2.1 LAN LLC Alerts

Subtopics:

- A.2.1.1 LAN LLC Alert 1
- A.2.1.2 LAN LLC Alert 2
- A.2.1.3 LAN LLC Alert 3
- A.2.1.4 LAN LLC Alert 4
- A.2.1.5 LAN LLC Alert 5
- A.2.1.6 LAN LLC Alert 6
- A.2.1.7 LAN LLC Alert 7
- A.2.1.8 LAN LLC Alert 8
- A.2.1.9 LAN LLC Alert 9
- A.2.1.10 LAN LLC Alert 10
- A.2.1.11 LAN LLC Alert 11

A.2.1.1 LAN LLC Alert 1

Alert Condition:

A LAN logical link has been lost. The remote link station does not respond. The inactivity timer (Ti) or acknowledgement timer (T1) has expired, causing the remote station to be polled. The remote station does not respond to the poll.

Alert ID Number		X'5B8F5BA7'
Alert Type	X'01'	Permanent
Alert Description	X'3300'	Link error
Probable Causes	X'2107'	LAN LLC communications/remote node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2107'	LAN LLC communications/remote node
	X'F017'	Poll count exhausted
Actions	X'3301'	If problem persists then do the following
	X'2010'	Review link detail data
	X'3103'	Contact LAN administrator responsible for this LAN
	X'32C0'	Report the following
	X'82' SF	(Adapter Number)
	X'82' SF	(Reference Code)
Additional SVs	X'51' SV	LAN LCS Data
	X'02' SF	Ring/Segment Identifier
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)

	X'04' SF	Remote Individual MAC Address
	X'24' SF	Remote Individual MAC Name (Optional)
	X'05' SF	LAN Routing Information (1)
	X'52' SV	LCS Configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Resource Name=(LAN name)
		Resource Type=X'39' (LAN)
		Second resource below sender:
		Resource Name=(CP name)
		Resource Type=X'F4' (CP)

Notes:

1. This subfield is present if the lost logical link traversed a MAC bridge.

A.2.1.2 LAN LLC Alert 2

Alert Condition:

A LAN logical link has been lost. The remote link station sent a Disconnect Mode response to the local link station.

Alert ID Number		X'B1D9A4C5'
Alert Type	X'01'	Permanent
Alert Description	X'3300'	Link error
Probable Causes	X'2007'	LAN LLC communications
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2007'	LAN LLC communications
	X'F01A'	DM received

Actions	X'3303'	If problem persists then do the following
	X'2010'	Review link detail data
	X'3101'	Contact LAN administrator responsible for this LAN
	X'32C0'	Report the following
	X'82' SF	(Adapter Number)
	X'82' SF	(Reference Code)
Additional SVs	X'51' SV	LAN LCS Data
	X'02' SF	Ring/Segment Identifier
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'04' SF	Remote Individual MAC Address
	X'24' SF	Remote Individual MAC Name (Optional)
	X'05' SF	LAN Routing Information (1)
	X'52' SV	LCS Configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Resource Name=(LAN name)
		Resource Type=X'39' (LAN)
		Second resource below sender:
		Resource Name=(CP name)
		Resource Type=X'F4' (CP)

Notes:

1. This subfield is present if the lost logical link traversed a MAC bridge.

A.2.1.3 LAN LLC Alert 3

Alert Condition:

A LAN logical link has been lost. The remote link station sent a SABME command to the local link station which was already open (had been initialized via a SABME-UA exchange).

Alert ID Number		X'E65B0B7F'
Alert Type	X'01'	Permanent
Alert Description	X'2100'	Software program error
Probable Causes	X'2007'	LAN LLC communications
	X'1023'	Communications program in remote node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1023'	Communications program in remote node
	X'F016'	SABME received while in ABME
Actions	X'3301'	If problem persists then do the following
	X'2010'	Review link detail data
	X'3103'	Contact LAN administrator responsible for this LAN
	X'32C0'	Report the following
	X'82' SF	(Adapter Number)
	X'82' SF	(Reference Code)
Additional SVs	X'51' SV	LAN LCS Data
	X'02' SF	Ring/Segment Identifier
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'04' SF	Remote Individual MAC Address
	X'24' SF	Remote Individual MAC Name (Optional)
	X'05' SF	LAN Routing Information (1)
	X'52' SV	LCS Configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Resource Name=(LAN name)

		Resource Type=X'39' (LAN)
		Second resource below sender:
		Resource Name=(CP name)
		Resource Type=X'F4' (CP)

Notes:

1. This subfield is present if the lost logical link traversed a MAC bridge.

A.2.1.4 LAN LLC Alert 4

Alert Condition:

A LAN logical link has been lost. The local link station sent an invalid or unsupported command or response to the remote link station. This resulted in the remote link station returning a Frame Reject response.

Alert ID Number		X'8A5B2D2C'
Alert Type	X'01'	Permanent
Alert Description	X'2100'	Software program error
Probable Causes	X'2007'	LAN LLC communications
	X'1000'	Software program
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1000'	Software program
	X'F010'	Frame reject received:
		Invalid/unsupported command or response sent
Actions	X'3301'	If problem persists then do the following:
	X'2010'	Review link detail data
	X'3103'	Contact LAN administrator responsible for this LAN
	X'32C0'	Report the following
	X'82' SF	(Adapter Number)
	X'82' SF	(Reference Code)
Additional SVs	X'51' SV	LAN LCS Data
	X'02' SF	Ring/Segment Identifier
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'04' SF	Remote Individual MAC Address
	X'24' SF	Remote Individual MAC Name (Optional)

	X'05' SF	LAN Routing Information (1)
	X'52' SV	LCS Configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Resource Name=(LAN name)
		Resource Type=X'39' (LAN)
		Second resource below sender:
		Resource Name=(CP name)
		Resource Type=X'F4' (CP)

Notes:

1. This subfield is present if the lost logical link traversed a MAC bridge.

A.2.1.5 LAN LLC Alert 5

Alert Condition:

A LAN logical link has been lost. The local link station sent an I-field when not permitted to the remote link station. This resulted in the remote link station returning a Frame Reject response.

Alert ID Number		X'8E9A309B'
Alert Type	X'01'	Permanent
Alert Description	X'2100'	Software program error
Probable Causes	X'2007'	LAN LLC communications
	X'1000'	Software program
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1000'	Software program
	X'F011'	Frame reject received:
		I-field sent when not permitted

Actions	X'3301'	If problem persists then do the following:
	X'2010'	Review link detail data
	X'3103'	Contact LAN administrator responsible for this LAN
	X'32C0'	Report the following
	X'82' SF	(Adapter Number)
	X'82' SF	(Reference Code)
Additional SVs	X'51' SV	LAN LCS Data
	X'02' SF	Ring/Segment Identifier
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'04' SF	Remote Individual MAC Address
	X'24' SF	Remote Individual MAC Name (Optional)
	X'05' SF	LAN Routing Information (1)
	X'52' SV	LCS Configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Resource Name=(LAN name)
		Resource Type=X'39' (LAN)
		Second resource below sender:
		Resource Name=(CP name)
		Resource Type=X'F4' (CP)

Notes:

1. This subfield is present if the lost logical link traversed a MAC bridge.

A.2.1.6 LAN LLC Alert 6

Alert Condition:

A LAN logical link has been lost. The local link station sent a frame with an invalid N(r). This resulted in the remote link station returning a Frame Reject response.

Alert ID Number		X'83D91642'
Alert Type	X'01'	Permanent
Alert Description	X'2100'	Software program error
Probable Causes	X'2007'	LAN LLC communications
	X'1000'	Software program
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1000'	Software program
	X'F012'	Frame reject received: invalid N(r) sent
Actions	X'3301'	If problem persists then do the following:
	X'2010'	Review link detail data
	X'3103'	Contact LAN administrator responsible for this LAN
	X'32C0'	Report the following
	X'82' SF	(Adapter Number)
	X'82' SF	(Reference Code)
Additional SVs	X'51' SV	LAN LCS Data
	X'02' SF	Ring/Segment Identifier
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'04' SF	Remote Individual MAC Address
	X'24' SF	Remote Individual MAC Name (Optional)
	X'05' SF	LAN Routing Information (1)
	X'52' SV	LCS Configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Resource Name=(LAN name)
		Resource Type=X'39' (LAN)

		Second resource below sender:
		Resource Name=(CP name)
		Resource Type=X'F4' (CP)

Notes:

1. This subfield is present if the lost logical link traversed a MAC bridge.

A.2.1.7 LAN LLC Alert 7

Alert Condition:

A LAN logical link has been lost. The local link station sent a frame with an I-field that was too long. This resulted in the remote link station returning a Frame Reject response.

Alert ID Number		X'87180BF5'
Alert Type	X'01'	Permanent
Alert Description	X'2100'	Software program error
Probable Causes	X'2007'	LAN LLC communications
	X'1000'	Software program
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1000'	Software program
	X'F013'	Frame reject received:
		maximum I-field length exceeded
Actions	X'3301'	If problem persists then do the following:
	X'2010'	Review link detail data
	X'3103'	Contact LAN administrator responsible for this LAN
	X'32C0'	Report the following
	X'82' SF	(Adapter Number)
	X'82' SF	(Reference Code)
Additional SVs	X'51' SV	LAN LCS Data
	X'02' SF	Ring/Segment Identifier
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'04' SF	Remote Individual MAC Address
	X'24' SF	Remote Individual MAC Name (Optional)
	X'05' SF	LAN Routing Information (1)

	X'52' SV	LCS Configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Resource Name=(LAN name)
		Resource Type=X'39' (LAN)
		Second resource below sender:
		Resource Name=(CP name)
		Resource Type=X'F4' (CP)

Notes:

1. This subfield is present if the lost logical link traversed a MAC bridge.

A.2.1.8 LAN LLC Alert 8

Alert Condition:

A LAN logical link has been lost. The remote link station sent an invalid or unsupported command or response to the local link station. This resulted in the local link station returning a Frame Reject response.

Alert ID Number		X'28EF2B5D'
Alert Type	X'01'	Permanent
Alert Description	X'2100'	Software program error
Probable Causes	X'2007'	LAN LLC communications
	X'1023'	Communications program in remote node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1023'	Communications program in remote node
	X'F020'	Invalid/unsupported command or response received
Actions	X'3301'	If problem persists then do the following:

	X'2010'	Review link detail data
	X'3103'	Contact LAN administrator responsible for this LAN
	X'32C0'	Report the following
	X'82' SF	(Adapter Number)
	X'82' SF	(Reference Code)
Additional SVs	X'51' SV	LAN LCS Data
	X'02' SF	Ring/Segment Identifier
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'04' SF	Remote Individual MAC Address
	X'24' SF	Remote Individual MAC Name (Optional)
	X'05' SF	LAN Routing Information (1)
	X'52' SV	LCS Configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Resource Name=(LAN name)
		Resource Type=X'39' (LAN)
		Second resource below sender:
		Resource Name=(CP name)
		Resource Type=X'F4' (CP)

Notes:

1. This subfield is present if the lost logical link traversed a MAC bridge.

A.2.1.9 LAN LLC Alert 9

Alert Condition:

A LAN logical link has been lost. The remote link station sent an I-field when not permitted to the local link station. This resulted in the local link station returning a Frame Reject response.

Alert ID Number		X'2C2E36EA'

Alert Type	X'01'	Permanent
Alert Description	X'2100'	Software program error
Probable Causes	X'2007'	LAN LLC communications
	X'1023'	Communications program in remote node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1023'	Communications program in remote node
	X'F021'	I-field received when not permitted
Actions	X'3301'	If problem persists then do the following:
	X'2010'	Review link detail data
	X'3103'	Contact LAN administrator responsible for this LAN
	X'32C0'	Report the following
	X'82' SF	(Adapter Number)
	X'82' SF	(Reference Code)
Additional SVs	X'51' SV	LAN LCS Data
	X'02' SF	Ring/Segment Identifier
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'04' SF	Remote Individual MAC Address
	X'24' SF	Remote Individual MAC Name (Optional)
	X'05' SF	LAN Routing Information (1)
	X'52' SV	LCS Configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Resource Name=(LAN name)
		Resource Type=X'39' (LAN)
		Second resource below sender:
		Resource Name=(CP name)

		Resource Type=X'F4' (CP)

Notes:

1. This subfield is present if the lost logical link traversed a MAC bridge.

A.2.1.10 LAN LLC Alert 10

Alert Condition:

A LAN logical link has been lost. The remote link station sent a frame with an invalid N(r). This resulted in the local link station returning a Frame Reject response.

Alert ID Number		X'216D1033'
Alert Type	X'01'	Permanent
Alert Description	X'2100'	Software program error
Probable Causes	X'2007'	LAN LLC communications
	X'1023'	Communications program in remote node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1023'	Communications program in remote node
	X'F022'	Invalid N(r) received
Actions	X'3301'	If problem persists then do the following:
	X'2010'	Review link detail data
	X'3103'	Contact LAN administrator responsible for this LAN
	X'32C0'	Report the following
	X'82' SF	(Adapter Number)
	X'82' SF	(Reference Code)
Additional SVs	X'51' SV	LAN LCS Data
	X'02' SF	Ring/Segment Identifier
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'04' SF	Remote Individual MAC Address
	X'24' SF	Remote Individual MAC Name (Optional)
	X'05' SF	LAN Routing Information (1)
	X'52' SV	LCS Configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address

	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Resource Name=(LAN name)
		Resource Type=X'39' (LAN)
		Second resource below sender:
		Resource Name=(CP name)
		Resource Type=X'F4' (CP)

Notes:

1. This subfield is present if the lost logical link traversed a MAC bridge.

A.2.1.11 LAN LLC Alert 11

Alert Condition:

A LAN logical link has been lost. The remote link station sent a frame with an I-field that was too long. This resulted in the local link station returning a Frame Reject response.

Alert ID Number		X'25AC0D84'
Alert Type	X'01'	Permanent
Alert Description	X'2100'	Software program error
Probable Causes	X'2007'	LAN LLC communications
	X'1023'	Communications program in remote node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1023'	Communications program in remote node
	X'F023'	Received I-field exceeded maximum length
Actions	X'3301'	If problem persists then do the following:
	X'2010'	Review link detail data
	X'3103'	Contact LAN administrator responsible for this LAN

	X'32C0'	Report the following
	X'82' SF	(Adapter Number)
	X'82' SF	(Reference Code)
Additional SVs	X'51' SV	LAN LCS Data
	X'02' SF	Ring/Segment Identifier
	X'03' SF	Local Individual MAC Address
	X'23' SF	Local Individual MAC Name (Optional)
	X'04' SF	Remote Individual MAC Address
	X'24' SF	Remote Individual MAC Name (Optional)
	X'05' SF	LAN Routing Information (1)
	X'52' SV	LCS Configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Resource Name=(LAN name)
		Resource Type=X'39' (LAN)
		Second resource below sender:
		Resource Name=(CP name)
		Resource Type=X'F4' (CP)

Notes:

1. This subfield is present if the lost logical link traversed a MAC bridge.

A.2.2 SDLC Alerts

Subtopics:

- A.2.2.1 SDLC Alert 1
- A.2.2.2 SDLC Alert 2
- A.2.2.3 SDLC Alert 3
- A.2.2.4 SDLC Alert 4
- A.2.2.5 SDLC Alert 5
- A.2.2.6 SDLC Alert 6
- A.2.2.7 SDLC Alert 7
- A.2.2.8 SDLC Alert 8
- A.2.2.9 SDLC Alert 9
- A.2.2.10 SDLC Alert 10
- A.2.2.11 SDLC Alert 11
- A.2.2.12 SDLC Alert 12
- A.2.2.13 SDLC Alert 13
- A.2.2.14 SDLC Alert 14

A.2.2.1 SDLC Alert 1

Alert Condition:

An SDLC logical link has been lost. The secondary link station does not respond to poll frames sent by the primary station.

Alert ID Number		X'32A37F1B'
Alert Type	X'01'	Permanent
Alert Description	X'3300'	Link error
Probable Causes	X'2104'	SDLC communications/remote node
	X'2031'	Line
User Causes	X'0209'	Remote device power off
Actions	X'0200'	Check power
Install Causes	(none)	
Failure Causes	X'2104'	SDLC communications/remote node
	X'3511'	Line
	X'F017'	Poll count exhausted
Actions		(Sender-specific actions)
Additional SVs	X'52' SV	LCS Configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'06' SF	Link Station Attributes
	X'07' SF	Link Attributes
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Resource Name=(Adapter number)
		Resource Type=X'21' (Adapter)
		Second resource below sender:
		Resource Name=(CP name)
		Resource Type=X'F4' (CP)

A.2.2.2 SDLC Alert 2

Alert Condition:

An SDLC logical link has been lost. The secondary link station sent a Disconnect Mode response to the primary link station.

Alert ID Number		X'BD84C4C9'
Alert Type	X'01'	Permanent
Alert Description	X'3300'	Link error
Probable Causes	X'2004'	SDLC communications
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2004'	SDLC communications
	X'F01A'	DM received
Actions		(Sender-specific actions)
Additional SVs	X'52' SV	LCS Configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'06' SF	Link Station Attributes
	X'07' SF	Link Attributes
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Resource Name=(Adapter number)
		Resource Type=X'21' (Adapter)
		Second resource below sender:
		Resource Name=(CP name)
		Resource Type=X'F4' (CP)

A.2.2.3 SDLC Alert 3

Alert Condition:

An SDLC logical link has been lost. The primary link station sent a SNRM command to the secondary link station while it was in NRM.

Alert ID Number		X'D635CA1E'
Alert Type	X'01'	Permanent
Alert Description	X'2100'	Software program error
Probable Causes	X'2004'	SDLC communications
	X'1023'	Communications program in remote node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1023'	Communications program in remote node
	X'F015'	SNRM received while in NRM
Actions		(Sender-specific actions)
Additional SVs	X'52' SV	LCS Configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'06' SF	Link Station Attributes
	X'07' SF	Link Attributes
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Resource Name=(Adapter number)
		Resource Type=X'21' (Adapter)
		Second resource below sender:
		Resource Name=(CP name)
		Resource Type=X'F4' (CP)

A.2.2.4 SDLC Alert 4

Alert Condition:

An SDLC logical link has been lost. The primary link station sent an invalid or unsupported command to the secondary link station. This resulted in the secondary link station returning a Frame reject response.

Alert ID Number		X'B776CA94'
Alert Type	X'01'	Permanent
Alert Description	X'2100'	Software program error
Probable Causes	X'2004'	SDLC communications
	X'1000'	Software program
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1000'	Software program
	X'F010'	Frame reject received:
		invalid/unsupported command or response sent
Actions		(Sender-specific actions)
Additional SVs	X'52' SV	LCS Configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'06' SF	Link Station Attributes
	X'07' SF	Link Attributes
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Resource Name=(Adapter number)
		Resource Type=X'21' (Adapter)
		Second resource below sender:
		Resource Name=(CP name)
		Resource Type=X'F4' (CP)

A.2.2.5 SDLC Alert 5

Alert Condition:

An SDLC logical link has been lost. The primary link station sent an I-field when not permitted to the secondary link station. This resulted in the secondary link station returning a Frame reject response.

Alert ID Number		X'B3B7D723'
Alert Type	X'01'	Permanent
Alert Description	X'2100'	Software program error
Probable Causes	X'2004'	SDLC communications
	X'1000'	Software program
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1000'	Software program
	X'F011'	Frame reject received:
		I-field sent when not permitted
Actions		(Sender-specific actions)
Additional SVs	X'52' SV	LCS Configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'06' SF	Link Station Attributes
	X'07' SF	Link Attributes
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Resource Name=(Adapter number)
		Resource Type=X'21' (Adapter)
		Second resource below sender:
		Resource Name=(CP name)
		Resource Type=X'F4' (CP)

A.2.2.6 SDLC Alert 6

Alert Condition:

An SDLC logical link has been lost. The primary link station sent a frame with an invalid N(r). This resulted in the secondary link station returning a Frame reject response.

Alert ID Number		X'BEF4F1FA'
Alert Type	X'01'	Permanent
Alert Description	X'2100'	Software program error
Probable Causes	X'2004'	SDLC communications
	X'1000'	Software program
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1000'	Software program
	X'F012'	Frame reject received: invalid N(r) sent
Actions		(Sender-specific actions)
Additional SVs	X'52' SV	LCS Configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'06' SF	Link Station Attributes
	X'07' SF	Link Attributes
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Resource Name=(Adapter number)
		Resource Type=X'21' (Adapter)
		Second resource below sender:
		Resource Name=(CP name)
		Resource Type=X'F4' (CP)

A.2.2.7 SDLC Alert 7

Alert Condition:

An SDLC logical link has been lost. The primary link station sent a frame with an I-field that was too long. This resulted in the secondary link station returning a Frame reject response.

Alert ID Number		X'BA35EC4D'
Alert Type	X'01'	Permanent
Alert Description	X'2100'	Software program error
Probable Causes	X'2004'	SDLC communications
	X'1000'	Software program
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1000'	Software program
	X'F013'	Frame reject received:
		maximum I-field length exceeded
Actions		(Sender-specific actions)
Additional SVs	X'52' SV	LCS Configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'06' SF	Link Station Attributes
	X'07' SF	Link Attributes
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Resource Name=(Adapter number)
		Resource Type=X'21' (Adapter)
		Second resource below sender:
		Resource Name=(CP name)
		Resource Type=X'F4' (CP)

A.2.2.8 SDLC Alert 8

Alert Condition:

An SDLC logical link has been lost. The secondary link station sent an invalid or unsupported command to the primary link station. This resulted in the primary link station returning a Frame reject response.

Alert ID Number		X'15C2CCE5'
Alert Type	X'01'	Permanent
Alert Description	X'2100'	Software program error
Probable Causes	X'2004'	SDLC communications
	X'1023'	Communications program in remote node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1023'	Communications program in remote node
	X'F020'	Invalid/unsupported command or response received
Actions		(Sender-specific actions)
Additional SVs	X'52' SV	LCS Configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'06' SF	Link Station Attributes
	X'07' SF	Link Attributes
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Resource Name=(Adapter number)
		Resource Type=X'21' (Adapter)
		Second resource below sender:
		Resource Name=(CP name)
		Resource Type=X'F4' (CP)

A.2.2.9 SDLC Alert 9

Alert Condition:

An SDLC logical link has been lost. The secondary link station sent an I-field when not permitted to the primary link station. This resulted in the primary link station returning a Frame reject response.

Alert ID Number		X'1103D152'
Alert Type	X'01'	Permanent
Alert Description	X'2100'	Software program error
Probable Causes	X'2004'	SDLC communications
	X'1023'	Communications program in remote node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1023'	Communications program in remote node
	X'F021'	I-field received when not permitted
Actions		(Sender-specific actions)
Additional SVs	X'52' SV	LCS Configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'06' SF	Link Station Attributes
	X'07' SF	Link Attributes
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Resource Name=(Adapter number)
		Resource Type=X'21' (Adapter)
		Second resource below sender:
		Resource Name=(CP name)
		Resource Type=X'F4' (CP)

A.2.2.10 SDLC Alert 10

Alert Condition:

```
An SDLC logical link has been lost.  The primary link station sent a frame
with an invalid N(r).  This resulted in the secondary link station
returning a Frame reject response.
```

Alert ID Number		X'1C40F78B'
Alert Type	X'01'	Permanent
Alert Description	X'2100'	Software program error
Probable Causes	X'2004'	SDLC communications
	X'1023'	Communications program in remote node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1023'	Communications program in remote node
	X'F022'	Invalid N(r) received
Actions		(Sender-specific actions)
Additional SVs	X'52' SV	LCS Configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'06' SF	Link Station Attributes
	X'07' SF	Link Attributes
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Resource Name=(Adapter number)
		Resource Type=X'21' (Adapter)
		Second resource below sender:
		Resource Name=(CP name)
		Resource Type=X'F4' (CP)

A.2.2.11 SDLC Alert 11

Alert Condition:

```
An SDLC  logical link has been lost.  The number of I-frames transmitted
```

by the remote link station has exceeded the local link station's receive window size.

Alert ID Number		X'EABB6A14'
Alert Type	X'01'	Permanent
Alert Description	X'2100'	Software program error
Probable Causes	X'2004'	SDLC communications
	X'1023'	Communications program in remote node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1023'	Communications program in remote node
	X'F01B'	Receive window size exceeded
Actions		(Sender-specific actions)
Additional SVs	X'52' SV	LCS Configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'06' SF	Link Station Attributes
	X'07' SF	Link Attributes
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Resource Name=(Adapter number)
		Resource Type=X'21' (Adapter)
		Second resource below sender:
		Resource Name=(CP name)
		Resource Type=X'F4' (CP)

A.2.2.12 SDLC Alert 12

Alert Condition:

An SDLC logical link has been lost. The secondary link station's inactivity timer has expired.

Alert ID Number		X'0E2DDF11'
Alert Type	X'01'	Permanent
Alert Description	X'3300'	Link error
Probable Causes	X'2104'	SDLC communications/remote node
	X'2031'	Line
User Causes	X'0209'	Remote device power off
Actions	X'0200'	Check power
Install Causes	(none)	
Failure Causes	X'2104'	SDLC communications/remote node
	X'3511'	Line
	X'F019'	Inactivity timer expired
Actions		(Sender-specific actions)
Additional SVs	X'52' SV	LCS Configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'06' SF	Link Station Attributes
	X'07' SF	Link Attributes
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Resource Name=(Adapter number)
		Resource Type=X'21' (Adapter)
		Second resource below sender:
		Resource Name=(CP name)
		Resource Type=X'F4' (CP)

A.2.2.13 SDLC Alert 13

Alert Condition:

Link establishment has failed. The local link station's retry limit for XID has been exceeded.

Alert ID Number		X'0AECC2A6'
Alert Type	X'01'	Permanent
Alert Description	X'3300'	Link error
Probable Causes	X'2104'	SDLC communications/remote node
	X'2031'	Line
User Causes	X'0209'	Remote device power off
Actions	X'0200'	Check power
Install Causes	(none)	
Failure Causes	X'2104'	SDLC communications/remote node
	X'3511'	Line
	X'F018'	XID poll count exhausted
Actions		(Sender-specific actions)
Additional SVs	X'52' SV	LCS Configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'06' SF	Link Station Attributes
	X'07' SF	Link Attributes
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Resource Name=(Adapter number)
		Resource Type=X'21' (Adapter)
		Second resource below sender:
		Resource Name=(CP name)
		Resource Type=X'F4' (CP)

A.2.2.14 SDLC Alert 14

Alert Condition:

An SDLC logical link has been lost. The secondary link station send a Frame reject response, but it contained no I-field indicating the reason for the rejection.

Alert ID Number		X'A472BC48'
Alert Type	X'01'	Permanent
Alert Description	X'2100'	Software program error

Probable Causes	X'2004'	SDLC communications
	X'1000'	Software program
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1000'	Software program
	X'F014'	Frame reject received: no reason specified
Actions		(Sender-specific actions)
Additional SVs	X'52' SV	LCS Configuration
	X'02' SF	Remote Device Address
	X'04' SF	Local Device Address
	X'06' SF	Link Station Attributes
	X'07' SF	Link Attributes
	X'8C' SV	Link Station Data
	X'01' SF	Current Ns/Nr Counts
	X'02' SF	Outstanding Frame Count
	X'03' SF	Last Control Field Received
	X'04' SF	Last Control Field Sent
	X'05' SF	Sequence Number Modulus
	X'06' SF	Link Station State
	X'07' SF	LLC Reply Timer Expiration Count
	X'08' SF	Last Received Nr Count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Resource Name=(Adapter number)
		Resource Type=X'21' (Adapter)
		Second resource below sender:
		Resource Name=(CP name)
		Resource Type=X'F4' (CP)

A.3 Alerts for Switched Link Connections

Subtopics:

- A.3.1 X.21 and X.21 Short-Hold Mode Alerts

A.3.1 X.21 and X.21 Short-Hold Mode Alerts

This section documents the Alerts sent by nodes using X.21 and X.21 Short Hold Mode. These Alerts are sent by type 2 or boundary-function-attached type 2.1 nodes. These stations do not send Alerts for errors detected on the initial call, but only send Alerts for errors detected on reconnections. These Alerts are sent as delayed Alerts, and are sent at all only if the same type 4 node to which the reconnection attempt was

directed is eventually reached.

Subtopics:

- A.3.1.1 X.21 Secondary Alert 1
- A.3.1.2 X.21 Peer-to-Peer Alert 1
- A.3.1.3 X.21 Secondary Alert 2
- A.3.1.4 X.21 Peer-to-Peer Alert 2
- A.3.1.5 X.21 Secondary Alert 3
- A.3.1.6 X.21 Peer-to-Peer Alert 3
- A.3.1.7 X.21 Secondary Alert 4
- A.3.1.8 X.21 Peer-to-Peer Alert 4
- A.3.1.9 X.21 Secondary Alert 5
- A.3.1.10 X.21 Peer-to-Peer Alert 5
- A.3.1.11 X.21 Secondary Alert 6
- A.3.1.12 X.21 Peer-to-Peer Alert 6
- A.3.1.13 X.21 Secondary Alert 7
- A.3.1.14 X.21 Peer-to-Peer Alert 7
- A.3.1.15 X.21 Secondary Alert 8
- A.3.1.16 X.21 Peer-to-Peer Alert 8
- A.3.1.17 X.21 Secondary Alert 9
- A.3.1.18 X.21 Peer-to-Peer Alert 9
- A.3.1.19 X.21 Secondary Alert 10
- A.3.1.20 X.21 Peer-to-Peer Alert 10
- A.3.1.21 X.21 Secondary Alert 11
- A.3.1.22 X.21 Peer-to-Peer Alert 11
- A.3.1.23 X.21 Secondary Alert 12
- A.3.1.24 X.21 Peer-to-Peer Alert 12
- A.3.1.25 X.21 Secondary Alert 13
- A.3.1.26 X.21 Peer-to-Peer Alert 13
- A.3.1.27 X.21 Secondary Alert 14
- A.3.1.28 X.21 Peer-to-Peer Alert 14
- A.3.1.29 X.21 Secondary Alert 15
- A.3.1.30 X.21 Peer-to-Peer Alert 15
- A.3.1.31 X.21 Secondary Alert 16
- A.3.1.32 X.21 Peer-to-Peer Alert 16
- A.3.1.33 X.21 Secondary Alert 17
- A.3.1.34 X.21 Peer-to-Peer Alert 17
- A.3.1.35 X.21 Secondary Alert 18
- A.3.1.36 X.21 Peer-to-Peer Alert 18
- A.3.1.37 X.21 Secondary Alert 19
- A.3.1.38 X.21 Peer-to-Peer Alert 19
- A.3.1.39 X.21 Secondary Alert 20
- A.3.1.40 X.21 Peer-to-Peer Alert 20
- A.3.1.41 X.21 Secondary Alert 21
- A.3.1.42 X.21 Peer-to-Peer Alert 21
- A.3.1.43 X.21 Secondary Alert 22
- A.3.1.44 X.21 Peer-to-Peer Alert 22
- A.3.1.45 X.21 Secondary Alert 23
- A.3.1.46 X.21 Peer-to-Peer Alert 23
- A.3.1.47 X.21 Secondary Alert 24
- A.3.1.48 X.21 Peer-to-Peer Alert 24
- A.3.1.49 X.21 Secondary Alert 25
- A.3.1.50 X.21 Peer-to-Peer Alert 25
- A.3.1.51 X.21 Secondary Alert 26
- A.3.1.52 X.21 Peer-to-Peer Alert 26
- A.3.1.53 X.21 Secondary Alert 27
- A.3.1.54 X.21 Peer-to-Peer Alert 27
- A.3.1.55 X.21 Secondary Alert 28
- A.3.1.56 X.21 Peer-to-Peer Alert 28
- A.3.1.57 X.21 Secondary Alert 29
- A.3.1.58 X.21 Peer-to-Peer Alert 29
- A.3.1.59 X.21 Secondary Alert 30
- A.3.1.60 X.21 Peer-to-Peer Alert 30
- A.3.1.61 X.21 Secondary Alert 31
- A.3.1.62 X.21 Peer-to-Peer Alert 31
- A.3.1.63 X.21 Secondary Alert 32
- A.3.1.64 X.21 Peer-to-Peer Alert 32
- A.3.1.65 X.21 Secondary Alert 33
- A.3.1.66 X.21 Peer-to-Peer Alert 33
- A.3.1.67 X.21 Secondary Alert 34
- A.3.1.68 X.21 Peer-to-Peer Alert 34

A.3.1.1 X.21 Secondary Alert 1

Alert Condition:

The secondary station received a CPS01, but the X.21 network did not signal ready for data within a specified period of time (See GA27-3287 *IBM Implementation of X.21 Interface General Information Manual* for the value of the timer.) This indicates the outgoing call was signalled to the DTE, but no further response was received. This is sent as a delayed Alert by the secondary station.

Alert ID Number		X'861061E5'
Alert Type	X'01'	Permanent (Delayed)

Alert Description	X'3311'	X.21 error -- SNA secondary
Probable Causes	X'2201'	Called DTE
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2201'	Called DTE
	X'2300'	Connection not established
Actions	X'3302'	If problem continues to occur then do the following
	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

A.3.1.2 X.21 Peer-to-Peer Alert 1

Alert Condition:

The reporting station received a CPS01, but the X.21 network did not signal ready for data within a specified period of time (See GA27-3287 *IBM Implementation of X.21 Interface General Information Manual* for the value of the timer.) This indicates the outgoing call was signalled to the DTE, but no further response was received.

Alert ID Number		X'6EB437E0'
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error
Probable Causes	X'2201'	Called DTE
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2201'	Called DTE
	X'2300'	Connection not established
Actions	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

A.3.1.3 X.21 Secondary Alert 2

Alert Condition:

The secondary station received a CPS02, CPS03 or CPS05, but the X.21 network did not signal ready for data within a specified period of time (See GA27-3287 *IBM Implementation of X.21 Interface General Information Manual* for the value of the timer.) This indicates the outgoing call was received by the DTE, but no further response was received. This is sent as a delayed Alert by the secondary station.

Alert ID Number		X'D974044D'
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3311'	X.21 error -- SNA secondary
Probable Causes	X'2005'	X.21 network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2005'	X.21 network
	X'2300'	Connection not established
Actions	X'3302'	If problem continues to occur then do the following
	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

A.3.1.4 X.21 Peer-to-Peer Alert 2

Alert Condition:

The reporting station received a CPS02, CPS03, or CPS05, but the X.21 network did not signal ready for data within a specified period of time (See GA27-3287 *IBM Implementation of X.21 Interface General Information Manual* for the value of the timer.) This indicates the outgoing call was received by the DTE, but no further response was received.

Alert ID Number		X'31D05248'
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error
Probable Causes	X'2005'	X.21 network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2005'	X.21 network
	X'2300'	Connection not established
Actions	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

A.3.1.5 X.21 Secondary Alert 3

Alert Condition:

The secondary station received a CPS04, but the X.21 network did not signal ready for data within a specified period of time (See GA27-3287 *IBM Implementation of X.21 Interface General Information Manual* for the value of the timer.) This indicates the outgoing call reached a private network, but no further response was received. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'5C1722E8' (1)
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3311'	X.21 error -- SNA secondary
Probable Causes	X'2401'	Private network reached
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2041'	Private network reached
	X'2300'	Connection not established
Actions	X'3302'	If problem continues to occur then do the following
	X'3104'	Contact network information service for private network called
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

Notes:

1. This Alert was previously documented with the incorrect Alert ID value of X'C05198EA'. Alert receivers should be able to accept Alerts containing either the current, correct value or the old, incorrect value.

A.3.1.6 X.21 Peer-to-Peer Alert 3

Alert Condition:

The reporting station received a CPS04, but the X.21 network did not signal ready for data within a specified period of time (See GA27-3287 *IBM Implementation of X.21 Interface General Information Manual* for the value of the timer.) This indicates the outgoing call reached a private network, but no further response was received.

Alert ID Number		X'28F5CEEF'
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error
Probable Causes	X'2041'	Private network reached
User Causes	(none)	

Install Causes	(none)	
Failure Causes	X'2041'	Private network Called
	X'2300'	Connection not established
Actions	X'3104'	Contact network information service for private network called
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

A.3.1.7 X.21 Secondary Alert 4

Alert Condition:

Retries have been exhausted and a CPS20 was received by the secondary station indicating the called number could not be connected. No cause was specified. This is sent as a delayed Alert by the secondary station.

Alert ID Number		X'B511541E'
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3311'	X.21 error -- SNA secondary
Probable Causes	X'FE00'	X.21 network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2005'	X.21 network

Actions	X'3302'	If problem continues to occur then do the following
	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

A.3.1.8 X.21 Peer-to-Peer Alert 4

Alert Condition:

Retries have been exhausted and a CPS20 was received by the reporting station indicating the called number could not be connected. No cause was specified.

Alert ID Number		X'4CBDD3F0'
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error
Probable Causes	X'FE00'	Undetermined
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2005'	X.21 network
Actions	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)

		Type=X'F4' (CP)

A.3.1.9 X.21 Secondary Alert 5

Alert Condition:

The retries have been exhausted when trying to connect or reconnect a link and a CPS21 was received by the secondary station indicating the called number was busy. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'F230538E'
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3311'	X.21 error -- SNA secondary
Probable Causes	X'2301'	Called number busy
User Causes	X'2301'	Called number busy
Actions	X'3302'	If problem continues to occur then do the following
	X'3122'	Contact called DTE's operator
	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)
Install Causes	(none)	
Failure Causes	(none)	

A.3.1.10 X.21 Peer-to-Peer Alert 5

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and a CPS21 was received by the reporting station indicating the called number was busy.

Alert ID Number		X'0B9CD460' (1)
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error
Probable Causes	X'2301'	Called number busy
User Causes	X'2301'	Called number busy

Actions	X'1206'	Wait Then Retry
	X'3301'	If Problem Persists Then Do the Following
	X'3122'	Contact called DTE's operator
	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)
Install Causes	(none)	
Failure Causes	(none)	
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

Notes:

1. This Alert was previously documented with the incorrect Alert ID value of X'3CD24384'. Alert receivers should be able to accept Alerts containing either the current, correct value or the old, incorrect value.

A.3.1.11 X.21 Secondary Alert 6

Alert Condition:

The retries have been exhausted when trying to connect or reconnect a link and a CPS22 was received by the secondary station indicating there was a procedure error in the selection signals. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'C2E9214F'
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3311'	X.21 error -- SNA secondary

Probable Causes	X'3309'	Line adapter
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3309'	Line adapter
Actions	X'3302'	If problem continues to occur then do the following
	X'30E1'	Contact the service representative for
	X'83' SF	(Product Identifier)
	X'3503'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

A.3.1.12 X.21 Peer-to-Peer Alert 6

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and a CPS22 was received by the reporting station indicating there was a procedure error with the selection signals.

Alert ID Number		X'3B45A6A1'
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error
Probable Causes	X'3309'	Line adapter
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3309'	Line adapter
Actions	X'30E1'	Contact Service Representative for
	X'83' SF	(Product Identifier)
	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data

	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

A.3.1.13 X.21 Secondary Alert 7

Alert Condition:

The retries have been exhausted when trying to connect or reconnect a link and a CPS23 was received by the secondary station indicating there was a transmission error in the selection signals. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'9B1BDB20'
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3311'	X.21 error -- SNA secondary
Probable Causes	X'2036'	DCE-DSE connection
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3512'	The connection between the calling DCE and its DSE
Actions	X'3302'	If problem continues to occur then do the following
	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

A.3.1.14 X.21 Peer-to-Peer Alert 7

Alert Condition:

```
The retries have been exhausted trying to connect or reconnect a link and
a CPS23 was received by the reporting station indicating there was a
transmission error in the selection signals.
```

Alert ID Number		X'62B75CCE'
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error
Probable Causes	X'2036'	DCE-DSE connection
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3512'	The connection between the calling DCE and its DSE
Actions	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

A.3.1.15 X.21 Secondary Alert 8

Alert Condition:

```
The retries have been exhausted when trying to connect or reconnect a link
and a CPS41 was received by the secondary station.  This indicates the
calling DTE is not permitted connection to the called DTE.  This Alert is
sent as a delayed Alert by the secondary station.
```

Alert ID Number		X'033AC8C8'

Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3311'	X.21 error -- SNA secondary
Probable Causes	X'2308'	Access barred
User Causes	X'2304'	Incorrect telephone number specified
Actions	X'0103'	Verify telephone number
Install Causes	(none)	
Failure Causes	X'2308'	Access barred
Actions	X'3302'	If problem continues to occur then do the following
	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

A.3.1.16 X.21 Peer-to-Peer Alert 8

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and a CPS41 was received by the reporting station. This indicates the calling DTE is not permitted connection to the called DTE.

Alert ID Number		X'EB9E9ECD'
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error
Probable Causes	X'2308'	Access barred
User Causes	X'2304'	Incorrect telephone number specified
Actions	X'0103'	Verify telephone number
Install Causes	(none)	
Failure Causes	X'2308'	Access barred
Actions	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)

	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

A.3.1.17 X.21 Secondary Alert 9

Alert Condition:

The retries have been exhausted when trying to connect or reconnect a link and a CPS42 was received by the secondary station. This indicates the called DTE has been assigned a new number. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'F0FE2CBD'
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3311'	X.21 error -- SNA secondary
Probable Causes	X'2306'	Changed Number
User Causes	X'2304'	Incorrect telephone number specified
Actions	X'0103'	Verify telephone number
Install Causes	(none)	
Failure Causes	X'2306'	New Telephone Number Assigned to Called DTE
Actions	X'3302'	If problem continues to occur then do the following
	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

A.3.1.18 X.21 Peer-to-Peer Alert 9

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and a CPS42 was received by the reporting station. This indicates the called DTE has been assigned a new number.

Alert ID Number		X'185A7AB8'
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error
Probable Causes	X'2306'	Changed Number
User Causes	X'2304'	Incorrect telephone number specified
Actions	X'0103'	Verify telephone number
Install Causes	(none)	
Failure Causes	X'2306'	New Telephone Number Assigned to Called DTE
Actions	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

A.3.1.19 X.21 Secondary Alert 10

Alert Condition:

The retries have been exhausted when trying to connect or reconnect a link and a CPS43 was received by the secondary station. This indicates the called DTE address is out of the numbering plan or not assigned to any DTE. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'E8561A00'
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3311'	X.21 error -- SNA secondary
Probable Causes	X'2300'	Connection not established
User Causes	X'2304'	Incorrect telephone number specified
Actions	X'0103'	Verify telephone number
Install Causes	(none)	
Failure Causes	X'2307'	Called number outside numbering plan or unknown by network
Actions	X'3302'	If problem continues to occur then do the following
	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

A.3.1.20 X.21 Peer-to-Peer Alert 10

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and a CPS43 was received by the reporting station. This indicates the called DTE address is out of the numbering plan or not assigned to any DTE.

Alert ID Number		X'00F24C05'
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error
Probable Causes	X'2300'	Connection not established
User Causes	X'2304'	Incorrect telephone number specified
Actions	X'0103'	Verify telephone number

Install Causes	(none)	
Failure Causes	X'2307'	Called number outside numbering plan or unknown by network
Actions	X'3105'	Retry
	X'32D0'	Contact X.21 network information service
	X'82' SF	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
		(Telephone Number Called)
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

A.3.1.21 X.21 Secondary Alert 11

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and a CPS44 was received by the secondary station. This indicates the called number is out of order. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'987DFF4D'
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3311'	X.21 error -- SNA secondary
Probable Causes	X'2201'	Called DTE
	X'3510'	Called DCE
	X'2005'	X.21 network
User Causes	X'0212'	Called DTE power off
	X'2510'	Line not enabled at called DTE
	X'0211'	Called DCE power off

Actions	X'1200'	Retry
	X'0200'	Check power
	X'1331'	Enable line then retry
Install Causes	(none)	
Failure Causes	X'2203'	Called DTE signalling uncontrolled not ready
	X'3513'	Local loop associated with the called DTE
Actions	X'3302'	If problem continues to occur then do the following
	X'3122'	Contact called DTE's operator
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

A.3.1.22 X.21 Peer-to-Peer Alert 11

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and a CPS44 was received by the reporting station. This indicates the called number is out of order.

Alert ID Number		X'9E31D69D'
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error
Probable Causes	X'2201'	Called DTE
	X'3510'	Called DCE
	X'2005'	X.21 network
User Causes	X'0212'	Called DTE power off
	X'2510'	Line not enabled at called DTE
	X'0211'	Called DCE power off
Actions	X'1200'	Retry
	X'0200'	Check power
	X'1331'	Enable line then retry
Install Causes	(none)	
Failure Causes	X'2203'	Called DTE signalling uncontrolled not ready
	X'3513'	Local loop associated with the called DTE

Actions	X'3301'	If problem persists then do the following
	X'3122'	Contact called DTE's operator
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

A.3.1.23 X.21 Secondary Alert 12

Alert Condition:

The retries have been exhausted when trying to connect or reconnect a link and a CPS45 was received by the secondary station. This indicates the called DTE is signalling controlled not ready. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'9E6C217A'
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3311'	X.21 error -- SNA secondary
Probable Causes	X'2201'	Called DTE
User Causes	X'2511'	Port deactivated at called DTE
Actions	X'1330'	Activate port then retry
Install Causes	(none)	
Failure Causes	X'2202'	Called DTE signalling controlled not ready
Actions	X'3302'	If problem continues to occur then do the following

	X'3122'	Contact called DTE's operator
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

A.3.1.24 X.21 Peer-to-Peer Alert 12

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and a CPS45 was received by the reporting station. This indicates the called DTE is signalling controlled not ready.

Alert ID Number		X'76C8777F'
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error
Probable Causes	X'2201'	Called DTE
User Causes	X'2511'	Port deactivated at called DTE
Actions	X'1330'	Activate port then retry
Install Causes	(none)	
Failure Causes	X'2202'	Called DTE signalling controlled not ready
Actions	X'1200'	Retry
	X'3301'	If problem persists then do the following
	X'3122'	Contact called DTE's operator
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)

		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

A.3.1.25 X.21 Secondary Alert 13

Alert Condition:

A CPS45 with a date and time was received by the secondary station. This indicates the called DTE is signalling controlled not ready. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'FEBADD77'
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3311'	X.21 error -- SNA secondary
Probable Causes	X'2201'	Called DTE
User Causes	X'2201'	Called DTE taken out of service
Actions	X'12C0'	Retry after
	X'82' SF	(Year/Month/Day)
	X'82' SF	(Time)
Install Causes	(none)	
Failure Causes	X'2202'	Called DTE signalling controlled not ready
Actions	X'3301'	If problem persists then do the following
	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

A.3.1.26 X.21 Peer-to-Peer Alert 13

Alert Condition:

A CPS45 with a date and time was received by the reporting station. This indicates the called DTE is signalling controlled not ready.

Alert ID Number		X'161E8B72'
Alert Type	X'01'	Permanent

Alert Description	X'3310'	X.21 error
Probable Causes	X'2201'	Called DTE
User Causes	X'2201'	Called DTE taken out of service
Actions	X'12C0'	Retry after
	X'82' SF	(Year/Month/Day)
	X'82' SF	(Time)
Install Causes	(none)	
Failure Causes	X'2202'	Called DTE signalling controlled not ready
Actions	X'3301'	If problem persists then do the following
	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

A.3.1.27 X.21 Secondary Alert 14

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and a CPS46 was received by the secondary station. This indicates the called DTE is signalling uncontrolled not ready. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'7B66A32A'
Alert Type	X'01'	Permanent (Delayed)

Alert Description	X'3311'	X.21 error -- SNA secondary
Probable Causes	X'2201'	Called DTE
User Causes	X'0212'	Called DTE
	X'2510'	Line not enabled at called DTE
Actions	X'0200'	Check power
	X'1331'	Enable line then retry
Install Causes	(none)	
Failure Causes	X'2203'	Called DTE signalling uncontrolled not ready
Actions	X'3302'	If problem continues to occur then do the following
	X'3122'	Contact called DTE's operator
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

A.3.1.28 X.21 Peer-to-Peer Alert 14

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and a CPS46 was received by the reporting station. This indicates the called DTE is signalling uncontrolled not ready.

Alert ID Number		X'A6694258'
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error
Probable Causes	X'2201'	Called DTE
User Causes	X'0212'	Called DTE power off
	X'2510'	Line not enabled at called DTE
Actions	X'0200'	Check power
	X'1331'	Enable line then retry
Install Causes	(none)	
Failure Causes	X'2203'	Called DTE signalling uncontrolled not ready

Actions	X'1200'	Retry
	X'3301'	If problem persists then do the following
	X'3122'	Contact called DTE's operator
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

A.3.1.29 X.21 Secondary Alert 15

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and a CPS47 was received by the secondary station. This indicates the called DCE does not have power. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'1630A02C' (1)
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3311'	X.21 error -- SNA secondary
Probable Causes	X'3510'	Called DCE
User Causes	X'0211'	Called DCE power off
Actions	X'0200'	Check power
Install Causes	(none)	
Failure Causes	X'3510'	Called DCE
Actions	X'3105'	If problem continues to occur then do the following

	X'32D0'	Contact X.21 network information service
	X'82' SF	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
		(Telephone Number Called)

Notes:

1. This Alert was previously documented with the incorrect Alert ID value of X'ECECAA6D'. Alert receivers should be able to accept Alerts containing either the current, correct value or the old, incorrect value.

A.3.1.30 X.21 Peer-to-Peer Alert 15

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and a CPS47 was received by the reporting station. This indicates the called DCE does not have power.

Alert ID Number		X'FE94F629'
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error
Probable Causes	X'3510'	Called DCE
User Causes	X'0211'	Called DCE power off
Actions	X'0200'	Check power
Install Causes	(none)	
Failure Causes	X'3510'	Called DCE
Actions	X'3105'	Retry
	X'32D0'	Contact X.21 network information service
	X'82' SF	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
		(Telephone Number Called)
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)

		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

A.3.1.31 X.21 Secondary Alert 16

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and a CPS48 was received by the secondary station. This indicates the requested facility is detected as invalid by the DCE at the local DTE/DCE interface. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'9EBCF239' (1)
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3311'	X.21 error -- SNA secondary
Probable Causes	X'2307'	Invalid request
User Causes	X'2300'	Calling DTE does not subscribe to this facility
Actions	X'3302'	If problem continues to occur then do the following
	X'3105'	Contact X.21 network information service
	X'32C0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
Install Causes	(none)	
Failure Causes	(none)	

Notes:

1. This Alert was previously documented with the incorrect Alert ID value of X'DF55F53F'. Alert receivers should be able to accept Alerts containing either the current, correct value or the old, incorrect value.

A.3.1.32 X.21 Peer-to-Peer Alert 16

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and a CPS48 was received by the reporting station. This indicates the requested facility is detected as invalid by the DCE at the local DTE/DCE interface.

Alert ID Number		X'26F972D1' (1)
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error
Probable Causes	X'2307'	Invalid request
User Causes	X'2308'	Calling DTE does not subscribe to this facility
Actions	X'3105'	Contact X.21 network information service
	X'32C0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
Install Causes	(none)	
Failure Causes	(none)	
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

Notes:

1. This Alert was previously documented with the incorrect Alert ID value of X'1CD57B79'. Alert receivers should be able to accept Alerts containing either the current, correct value or the old, incorrect value.

A.3.1.33 X.21 Secondary Alert 17

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and a CPS49 was received by the primary station. This indicates an error with the local loop associated with the called DTE. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'A5D509ED'

Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3311'	X.21 error -- SNA secondary
Probable Causes	X'2005'	X.21 network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3513'	Local loop associated with the called DTE
Actions	X'3302'	If problem continues to occur then do the following
	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

A.3.1.34 X.21 Peer-to-Peer Alert 17

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and a CPS49 was received by the reporting station. This indicates an error with the local loop associated with the called DTE.

Alert ID Number		X'5C798E03'
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error
Probable Causes	X'2005'	X.21 network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3513'	Local loop associated with the called DTE
Actions	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

A.3.1.35 X.21 Secondary Alert 18

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and a CPS51 was received by the primary station. This indicates the called number was busy and the network information service should be contacted for details. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'F230538E'
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3311'	X.21 error -- SNA secondary
Probable Causes	X'2301'	Called number busy
User Causes	X'2301'	Called number busy
Actions	X'3302'	If problem continues to occur then do the following
	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)
Install Causes	(none)	
Failure Causes	(none)	

A.3.1.36 X.21 Peer-to-Peer Alert 18

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and a CPS51 was received by the reporting station. This indicates the called number is temporarily busy and the network information service should be contacted for details.

Alert ID Number		X'0B9CD460' (1)
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error
Probable Causes	X'2301'	Called number busy
User Causes	X'2301'	Called number busy
Actions	X'3301'	If problem persists then do the following
	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)
Install Causes	(none)	
Failure Causes	(none)	
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

Notes:

1. This Alert was previously documented with the incorrect Alert ID value of X'3CD24384'. Alert receivers should be able to accept Alerts containing either the current, correct value or the old, incorrect value.

A.3.1.37 X.21 Secondary Alert 19

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and a CPS52 was received by the primary station. This indicates the called DTE belongs to a user class of service which is incompatible with that of the calling DTE. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'367EF253'
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3311'	X.21 error -- SNA secondary
Probable Causes	X'2304'	Incorrect number called
User Causes	X'2304'	Incorrect telephone number specified
Actions	X'0103'	Verify telephone number
Install Causes	(none)	
Failure Causes	X'2309'	Speed classes incompatible
	X'230A'	User classes of service incompatible
Actions	X'3302'	If problem continues to occur then do the following
	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

A.3.1.38 X.21 Peer-to-Peer Alert 19

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and a CPS52 was received by the reporting station. This indicates the called DTE belongs to a user class of service which is incompatible with that of the calling DTE.

Alert ID Number		X'EB711321'
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error
Probable Causes	X'2304'	Incorrect number called
User Causes	X'2304'	Incorrect telephone number specified
Actions	X'0103'	Verify telephone number

Install Causes	(none)	
Failure Causes	X'2309'	Speed classes incompatible
	X'230A'	User classes of service incompatible
Actions	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

A.3.1.39 X.21 Secondary Alert 20

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and a CPS61 was received by the station. This indicates the network is temporarily congested. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'D55C05B3'
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'5001'	Network congestion
Probable Causes	X'2005'	X.21 network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3521'	Temporary lack of resources in X.21 network

Actions	X'3302'	If problem continues to occur then do the following
	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

A.3.1.40 X.21 Peer-to-Peer Alert 20

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and a CPS61 was received by the station. This indicates the network is temporarily congested.

Alert ID Number		X'D55C05B3'
Alert Type	X'01'	Permanent
Alert Description	X'5001'	Network congestion
Probable Causes	X'2005'	X.21 network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3521'	Temporary lack of resources in X.21 network
Actions	X'3301'	If problem persists then do the following
	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)

		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

A.3.1.41 X.21 Secondary Alert 21

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and a CPS71 was received by the station. This indicates a shortage of network resources. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'D81F236A'
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'5001'	Network congestion
Probable Causes	X'2005'	X.21 network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3522'	Long term lack of resources in X.21 network
Actions	X'3302'	If problem continues to occur then do the following
	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

A.3.1.42 X.21 Peer-to-Peer Alert 21

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and a CPS71 was received by the station. This indicates a shortage of network resources.

Alert ID Number		X'D81F236A'
Alert Type	X'01'	Permanent
Alert Description	X'5001'	Network congestion

Probable Causes	X'2005'	X.21 network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3522'	Long term lack of resources in X.21 network
Actions	X'3301'	If problem persists then do the following
	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

A.3.1.43 X.21 Secondary Alert 22

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and a CPS72 was received by the station. This indicates the RPOA nominated by the calling DTE is unable to forward the call. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'0138B506'
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3311'	X.21 error -- SNA secondary
Probable Causes	X'2040'	Inter-exchange network
User Causes	(none)	

Install Causes	(none)	
Failure Causes	X'2040'	Inter-exchange network
Actions	X'3302'	If problem continues to occur then do the following
	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

A.3.1.44 X.21 Peer-to-Peer Alert 22

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and a CPS72 was received by the station. This indicates the RPOA nominated by the calling DTE is unable to forward the call.

Alert ID Number		X'F89432E8'
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error
Probable Causes	X'2040'	Inter-exchange network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2040'	Inter-exchange network
Actions	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)

		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

A.3.1.45 X.21 Secondary Alert 23

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and an invalid CPS was received by the secondary station. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'802A9670'
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3311'	X.21 error -- SNA secondary
Probable Causes	X'2036'	DCE-DSE connection
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3512'	The connection between the calling DCE and its DSE
	X'F059'	Invalid CPS received from network
Actions	X'3302'	If problem continues to occur then do the following
	X'3105'	Contact X.21 network information service
	X'32C0'	Report the following
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

A.3.1.46 X.21 Peer-to-Peer Alert 23

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and an invalid CPS was received by the reporting station.

Alert ID Number		X'688EC075'
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error

Probable Causes	X'2036'	DCE-DSE connection
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3512'	The connection between the calling DCE and its DSE
	X'F059'	Invalid CPS received from network
Actions	X'3105'	Contact X.21 network information service
	X'32C0'	Report the following
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

A.3.1.47 X.21 Secondary Alert 24

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and an unexpected CPS was received by the secondary station. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'EEEFE65A'
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3311'	X.21 error -- SNA secondary
Probable Causes	X'2005'	X.21 network
User Causes	(none)	
Install Causes	(none)	

Failure Causes	X'F058'	Unrecognized CPS received from the network
Actions	X'3302'	If problem continues to occur then do the following
	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

A.3.1.48 X.21 Peer-to-Peer Alert 24

Alert Condition:

The retries have been exhausted trying to connect or reconnect a link and an unexpected CPS was received by the reporting station.

Alert ID Number		X'174361B4'
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error
Probable Causes	X'2005'	X.21 network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'F058'	Unrecognized CPS received from the network
Actions	X'3105'	Contact X.21 network information service
	X'32D0'	Report the following
	X'82' SF	(Call Progress Signal)
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:

		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

A.3.1.49 X.21 Secondary Alert 25

Alert Condition:

This Alert indicates one of the timers, T1, T2, or T3, expired. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'92C4D92C'
Alert Type	X'01'	Permanent
Alert Description	X'3311'	X.21 error -- SNA secondary
Probable Causes	X'2005'	X.21 network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'20A0'	No response from X.21 network (sf82) expired
	X'82' SF	(Timer)
Actions	X'3302'	If problem continues to occur then do the following
	X'3105'	Contact X.21 network information service
	X'32C0'	Report the following
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

A.3.1.50 X.21 Peer-to-Peer Alert 25

Alert Condition:

This Alert indicates one of the timers, T1, T2, or T3, expired.

Alert ID Number		X'6B685EC2'
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error
Probable Causes	X'2005'	X.21 network

User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'20A0'	No response from X.21 network (sf82) expired
	X'82' SF	(Timer)
Actions	X'3105'	Contact X.21 network information service
	X'32C0'	Report the following
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

A.3.1.51 X.21 Secondary Alert 26

Alert Condition:

This Alert indicates timer T5 expired without receiving an expected response from the network. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'6B685EC2'
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3310'	X.21 error
Probable Causes	X'2005'	X.21 network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'20A0'	No response from X.21 network (sf82)

		expired
	X'82' SF	(Timer)
Actions	X'3302'	If problem continues to occur then do the following
	X'3105'	Contact X.21 network information service
	X'32A0'	Report the following
	X'82' SF	(Reporting Telephone Number)

A.3.1.52 X.21 Peer-to-Peer Alert 26

Alert Condition:

This Alert indicates timer T5 expired without receiving an expected response from the network.

Alert ID Number		X'6B685EC2'
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error
Probable Causes	X'2005'	X.21 network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'20A0'	No response from X.21 network (sf82) expired
	X'82' SF	(Timer)
Actions	X'3105'	If problem persists then do the following
	X'32A0'	Report the following
	X'82' SF	(Reporting Telephone Number)
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)

		Type=X'F4' (CP)

A.3.1.53 X.21 Secondary Alert 27

Alert Condition:

This Alert indicates the calling DCE is signalling not ready. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'333F4124'
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3311'	X.21 error -- SNA secondary
Probable Causes	X'2005'	X.21 network
User Causes	X'0210'	Calling DCE power off
Actions	X'0200'	Check power
Install Causes	(none)	
Failure Causes	X'2050'	X.21 network has initiated test loop
	X'3520'	X.21 network component
	X'F050'	DCE not ready
Actions	X'3302'	If problem continues to occur then do the following
	X'3105'	Contact X.21 network information service
	X'32C0'	Report the following
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

A.3.1.54 X.21 Peer-to-Peer Alert 27

Alert Condition:

This Alert indicates the calling DCE is signalling not ready.

Alert ID Number		X'24EC7079'
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error
Probable Causes	X'2005'	X.21 network
User Causes	X'0210'	Calling DCE power off

Actions	X'0200'	Check power
Install Causes	(none)	
Failure Causes	X'2050'	X.21 network has initiated test loop
	X'3520'	X.21 network component
	X'F050'	DCE not ready
Actions	X'3105'	Contact X.21 network information service
	X'32C0'	Report the following
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

A.3.1.55 X.21 Secondary Alert 28

Alert Condition:

A DCE clear indication was received by the secondary station during call establishment. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'3D061D5E'
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3311'	X.21 error -- SNA secondary
Probable Causes	X'2105'	X.21 communications/called node
User Causes	(none)	
Install Causes	(none)	

Failure Causes	X'3520'	X.21 network component
	X'2201'	Called DTE
	X'F051'	DCE not ready
Actions	X'3302'	If problem continues to occur then do the following
	X'3105'	Contact X.21 network information service
	X'32C0'	Report the following
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

A.3.1.56 X.21 Peer-to-Peer Alert 28

Alert Condition:

A DCE clear indication was received by the primary station during call establishment.

Alert ID Number		X'E009FC2C'
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error
Probable Causes	X'2105'	X.21 communications/called node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3520'	X.21 network component
	X'2201'	Called DTE
	X'F051'	DCE not ready
Actions	X'3105'	Contact X.21 network information service
	X'32C0'	Report the following
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:

		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

A.3.1.57 X.21 Secondary Alert 29

Alert Condition:

A persistent DCE clear indication was received by the secondary station during call establishment. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'6332915A'
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3311'	X.21 error -- SNA secondary
Probable Causes	X'2005'	X.21 network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3520'	X.21 network component
	X'F052'	Persistent DCE clear indication during call estab. (T6 exp)
Actions	X'3302'	If problem continues to occur then do the following
	X'3105'	Contact X.21 network information service
	X'32C0'	Report the following
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

A.3.1.58 X.21 Peer-to-Peer Alert 29

Alert Condition:

A persistent DCE clear indication was received by the reporting station during call establishment.

Alert ID Number		X'8B96C75F'
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error

Probable Causes	X'2005'	X.21 network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3520'	X.21 network component
	X'F052'	Persistent DCE clear indication during call estab. (T6 exp)
Actions	X'3105'	Contact X.21 network information service
	X'32C0'	Report the following
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

A.3.1.59 X.21 Secondary Alert 30

Alert Condition:

A DCE controlled not ready was received by the secondary station during call establishment. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'34842630'
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3311'	X.21 error -- SNA secondary
Probable Causes	X'2105'	X.21 communications/called node
User Causes	(none)	
Install Causes	(none)	

Failure Causes	X'3520'	X.21 network component
	X'2201'	Called DTE
	X'F053'	DCE controlled not ready during call Estab.
Actions	X'3302'	If problem continues to occur then do the following
	X'3105'	Contact X.21 network information service
	X'32C0'	Report the following
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

A.3.1.60 X.21 Peer-to-Peer Alert 30

Alert Condition:

A DCE controlled not ready was received by the reporting station during call establishment.

Alert ID Number		X'E98BC742' (1)
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error
Probable Causes	X'2105'	X.21 communications/called node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3520'	X.21 network component
	X'2201'	Called DTE
	X'F053'	DCE controlled not ready during call estab.
Actions	X'3105'	Contact X.21 network information service
	X'32C0'	Report the following
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)

		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

Notes:

1. This Alert was previously documented with the incorrect Alert ID value of X'ED79EE47'. Alert receivers should be able to accept Alerts containing either the current, correct value or the old, incorrect value.

A.3.1.61 X.21 Secondary Alert 31

Alert Condition:

A persistent DCE controlled not ready was received by the secondary station during call establishment. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'79B4DCE8'
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3311'	X.21 error -- SNA secondary
Probable Causes	X'2005'	X.21 network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3520'	X.21 network component
	X'F054'	Persistent DCE CNR during call estab. (T6 exp.)
Actions	X'3302'	If problem continues to occur then do the following
	X'3105'	Contact X.21 network information service
	X'32C0'	Report the following
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

A.3.1.62 X.21 Peer-to-Peer Alert 31

Alert Condition:

A persistent DCE controlled not ready was received by the reporting station during call establishment.

Alert ID Number		X'91108AED'
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error
Probable Causes	X'2005'	X.21 network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3520'	X.21 network component
	X'F054'	Persistent DCE CNR during call estab. (T6 exp.)
Actions	X'3105'	Contact X.21 network information service
	X'32C0'	Report the following
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

A.3.1.63 X.21 Secondary Alert 32

Alert Condition:

A DCE fault condition was received by the secondary station during call establishment. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'B5260D14' (1)
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3311'	X.21 error -- SNA secondary

Probable Causes	X'2036'	DCE-DSE connection
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3512'	The connection between the calling DCE and its DSE
	X'F055'	DCE fault condition during call establishment
Actions	X'3302'	If problem continues to occur then do the following
	X'3105'	Contact X.21 network information service
	X'32C0'	Report the following
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

Notes:

1. This Alert was previously documented with the incorrect Alert ID value of X'5E74704E'. Alert receivers should be able to accept Alerts containing either the current, correct value or the old, incorrect value.

A.3.1.64 X.21 Peer-to-Peer Alert 32

Alert Condition:

A DCE fault condition was received by the reporting station during call establishment.

Alert ID Number		X'5D825B11'
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error
Probable Causes	X'2036'	DCE-DSE connection
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3512'	The connection between the calling DCE and its DSE
	X'F055'	DCE fault condition during call establishment
Actions	X'3105'	Contact X.21 network information service
	X'32C0'	Report the following
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

A.3.1.65 X.21 Secondary Alert 33

Alert Condition:

A persistent DCE clear indication was received by the secondary station during data phase. This Alert is sent as a delayed Alert by the secondary station.

Alert ID Number		X'74F7FA31' (1)
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3311'	X.21 error -- SNA secondary
Probable Causes	X'2005'	X.21 network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3520'	X.21 network component
	X'F057'	Persistent DCE clear indication received in data phase (T6 exp)
Actions	X'3302'	If problem continues to occur then do the following
	X'3105'	Contact X.21 network information service
	X'32C0'	Report the following
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)

Notes:

1. This Alert was previously documented with the incorrect Alert ID value of X'49DA1D89'. Alert receivers should be able to accept Alerts

```
|     containing either the current, correct value or the old, incorrect
|     value.
```

A.3.1.66 X.21 Peer-to-Peer Alert 33

Alert Condition:

A persistent DCE clear indication was received by the primary station during data phase.

Alert ID Number		X'9C53AC34'
Alert Type	X'01'	Permanent
Alert Description	X'3310'	X.21 error
Probable Causes	X'2005'	X.21 network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3520'	X.21 network component
	X'F057'	Persistent DCE clear indication received in data phase (T6 exp)
Actions	X'3105'	Contact X.21 network information service
	X'32C0'	Report the following
	X'82' SF	(Calling Telephone Number)
	X'82' SF	(Telephone Number Called)
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

A.3.1.67 X.21 Secondary Alert 34

Alert Condition:

This a delayed Alert sent by the secondary station to indicate that the terminal control unit was reset.

Alert ID Number		X'C14B8A4F'
Alert Type	X'01'	Permanent (Delayed)
Alert Description	X'3313'	X.21 connection cleared
Probable Causes	X'7011'	Terminal control unit operator
User Causes	X'2310'	X.21 connection intentionally cleared by term. control operator
Actions	X'3121'	Contact terminal control unit operator
Install Causes	(none)	
Failure Causes	(none)	

A.3.1.68 X.21 Peer-to-Peer Alert 34

Alert Condition:

All available resources are inactive at the reporting station.

Alert ID Number		X'F72A9045'
Alert Type	X'01'	Permanent
Alert Description	X'7001'	Resource not active
Probable Causes	X'3381'	X.21 rotary group
User Causes	X'3381'	An op. deact. ports in rotary group used X.21 SHM
Actions	X'13A0'	Activate ports in rotary group associated with
	X'82' SF	(Telephone Number)
Install Causes	(none)	
Failure Causes	(none)	
Additional SVs	X'52' SV	Link Connection Subsystem Configuration Data
	X'01' SF	Port Address
	X'06' SF	Modem LCS Link Station Attributes
	X'05' SV	Hierarchy/Resource List
		Hierarchy Name List

		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(link name)
		Type=X'F9' (LINK)
		Second resource below sender:
		Name=(remote CP name)
		Type=X'F4' (CP)

A.4 Alerts for X.25 Link Connections

This section defines the Alerts sent by X.25 nodes.

Subtopics:

- A.4.1 Packet Layer Control (PLC)
- A.4.2 Link Access Protocol Balanced (LAPB)
- A.4.3 Logical Link Control (LLC)
- A.4.4 ISDN Alerts

A.4.1 Packet Layer Control (PLC)

Subtopics:

- A.4.1.1 X.25 PLC Alert 1
- A.4.1.2 X.25 PLC Alert 2
- A.4.1.3 X.25 PLC Alert 3
- A.4.1.4 X.25 PLC Alert 4
- A.4.1.5 X.25 PLC Alert 5
- A.4.1.6 X.25 PLC Alert 6
- A.4.1.7 X.25 PLC Alert 7
- A.4.1.8 X.25 PLC Alert 8
- A.4.1.9 X.25 PLC Alert 9
- A.4.1.10 X.25 PLC Alert 10
- A.4.1.11 X.25 PLC Alert 11

A.4.1.1 X.25 PLC Alert 1

Alert Condition:

A CLEAR_INDICATION packet containing a DCE-Originated cause code and a diagnostic code was received by the DTE.

Alert ID Number		X'B5B412E5' (1)
Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error
Probable Causes	X'2050'	Packet Layer Control
	X'2008'	X.25 Communications
	X'2006'	X.25 Network
	X'2200'	Remote Node

User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'20C1'	X.25 Communications Error -- The following indication packet was received from the network
	X'82' SF	(packet type and cause code)
	X'82' SF	(diagnostic code)
	X'2006'	X.25 Communications Error
	X'2200'	Remote Node
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32D0'	Report the following
	X'82' SF	(DTE address called)
	X'82' SF	(DTE address calling)
	X'82' SF	(local DTE address)
	X'F0A0'	For
	X'82' SF	(locally-initiated logical channel)
		or
	X'82' SF	(remotely-initiated logical channel)
	X'3123'	Contact remote DTE's operator
Additional SVs	X'52' SV	LCS Configuration Data
	X'07' SF	LCS Link Attributes
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(adapter name)
		Type=X'21' (adapter)
		Second resource below sender:
		Name=(port name)
		Type=X'3F' (port)

Notes:

1. This Alert was previously documented with the incorrect Alert ID value of X'D484ED27'. Alert receivers should be able to accept Alerts containing either the current, correct value or the old, incorrect value.

A.4.1.2 X.25 PLC Alert 2

Alert Condition:

A RESTART_INDICATION packet containing a DCE-Originated cause code and a diagnostic code was received by the DTE.

Alert ID Number		X'CDA515B8'

Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error
Probable Causes	X'2050'	Packet Layer Control
	X'2008'	X.25 Communications
	X'2006'	X.25 Network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'20C1'	X.25 Communications Error -- The
		following indication packet was
		received from the network
	X'82' SF	(packet type and cause code)
	X'82' SF	(diagnostic code)
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32A0'	Report the following
	X'82' SF	(local DTE address)
Additional SVs	X'52' SV	LCS Configuration Data
	X'07' SF	LCS Link Attributes
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(adapter name)
		Type=X'21' (adapter)
		Second resource below sender:
		Name=(port name)
		Type=X'3F' (port)

A.4.1.3 X.25 PLC Alert 3

Alert Condition:

A RESET_REQUEST packet containing a DTE-Originated cause code and a diagnostic code was sent by the DTE.

Alert ID Number		X'D3A1B295' (1)
Alert Type	X'02'	Temporary
Alert Description	X'3320'	X.25 Error
Probable Causes	X'2050'	Packet Layer Control

	X'2008'	X.25 Communications
	X'2200'	Remote Node
	X'2006'	X.25 Network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'20C2'	X.25 Communications Error -- The DTE sent the following request packet to the network
	X'82' SF	(packet type and cause code)
	X'82' SF	(diagnostic code)
	X'2200'	Remote Node
	X'2006'	X.25 Communications Error
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32D0'	Report the following
	X'82' SF	(DTE address called)
	X'82' SF	(DTE address calling)
	X'82' SF	(local DTE address)
	X'F0A0'	For
	X'82' SF	(locally-initiated logical channel)
		or
	X'82' SF	(remotely-initiated logical channel)
	X'3123'	Contact remote DTE's operator
Additional SVs	X'52' SV	LCS Configuration Data
	X'07' SF	LCS Link Attributes
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(adapter name)
		Type=X'21' (adapter)
		Second resource below sender:
		Name=(port name)
		Type=X'3F' (port)

Notes:

1. This Alert was previously documented with the incorrect Alert ID value of X'6A837F72'. Alert receivers should be able to accept Alerts containing either the current, correct value or the old, incorrect value.

A.4.1.4 X.25 PLC Alert 4

Alert Condition:

A CLEAR_REQUEST packet containing a DTE-Originated cause code and a diagnostic code was sent by the DTE.

Alert ID Number		X'056A9521'
Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error
Probable Causes	X'2050'	Packet Layer Control
	X'2008'	X.25 Communications
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'20C2'	X.25 Communications Error -- The DTE sent the following request packet to the network
	X'82' SF	(packet type and cause code)
	X'82' SF	(diagnostic code)
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32D0'	Report the following
	X'82' SF	(DTE address called)
	X'82' SF	(DTE address calling)
	X'82' SF	(local DTE address)
	X'F0A0'	For
	X'82' SF	(locally-initiated logical channel)
		or
	X'82' SF	(remotely-initiated logical channel)
Additional SVs	X'52' SV	LCS Configuration Data
	X'07' SF	LCS Link Attributes
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(adapter name)
		Type=X'21' (adapter)
		Second resource below sender:
		Name=(port name)
		Type=X'3F' (port)

A.4.1.5 X.25 PLC Alert 5

Alert Condition:

A RESTART_REQUEST packet containing a DTE-Originated cause code and a diagnostic code was sent by the DTE.

Alert ID Number		X'056A9521'
Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error
Probable Causes	X'2050'	Packet Layer Control
	X'2008'	X.25 Communications
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'20C2'	X.25 Communications Error -- The DTE
		sent the following request packet to
		the network
	X'82' SF	(packet type and cause code)
	X'82' SF	(diagnostic code)
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32A0'	Report the following
	X'82' SF	(local DTE address)
Additional SVs	X'52' SV	LCS Configuration Data
	X'07' SF	LCS Link Attributes
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(adapter name)
		Type=X'21' (adapter)
		Second resource below sender:
		Name=(port name)
		Type=X'3F' (port)

A.4.1.6 X.25 PLC Alert 6

Alert Condition:

Time-limit T20 expired at the DTE prior to receipt of a RESTART_CONFIRMATION packet following transfer of a RESTART_REQUEST packet.

Alert ID Number		X'F50A02F0'
Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error

Probable Causes	X'2050'	Packet Layer Control
	X'2008'	X.25 Communications
	X'2006'	X.25 Network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'20D1'	No response from the X.25 network --
	X'82' SF	(timer) expired
	X'82' SF	(retry count)
	X'82' SF	(timer setting)
	X'2006'	X.25 Communications Error
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32A0'	Report the following
	X'82' SF	(local DTE address)
Additional SVs	X'52' SV	LCS Configuration Data
	X'07' SF	LCS Link Attributes
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(adapter name)
		Type=X'21' (adapter)
		Second resource below sender:
		Name=(port name)
		Type=X'3F' (port)

A.4.1.7 X.25 PLC Alert 7

Alert Condition:

Time-limit T22 expired at the DTE prior to receipt of a RESET_CONFIRMATION packet following transfer of a RESET_REQUEST packet.

Alert ID Number		X'F50A02F0'
Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error
Probable Causes	X'2050'	Packet Layer Control
	X'2008'	X.25 Communications
	X'2006'	X.25 Network
User Causes	(none)	

Install Causes	(none)	
Failure Causes	X'20D1'	No response from the X.25 network --
	X'82' SF	(timer) expired
	X'82' SF	(retry count)
	X'82' SF	(timer setting)
	X'2006'	X.25 Communications Error
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32D0'	Report the following
	X'82' SF	(DTE address called)
	X'82' SF	(DTE address calling)
	X'82' SF	(local DTE address)
	X'F0A0'	For
	X'82' SF	(locally-initiated logical channel)
		or
	X'82' SF	(remotely-initiated logical channel)
Additional SVs	X'52' SV	LCS Configuration Data
	X'07' SF	LCS Link Attributes
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(adapter name)
		Type=X'21' (adapter)
		Second resource below sender:
		Name=(port name)
		Type=X'3F' (port)

A.4.1.8 X.25 PLC Alert 8

Alert Condition:

Time-limit T21 expired at the DTE prior to receipt of a CALL_CONNECTED or CLEAR_INDICATION packet following transfer of a CALL_REQUEST packet.

Alert ID Number		X'F50A02F0'
Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error
Probable Causes	X'2050'	Packet Layer Control
	X'2008'	X.25 Communications
	X'2006'	X.25 Network

User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'20D1'	No response from the X.25 network --
	X'82' SF	(timer) expired
	X'82' SF	(retry count)
	X'82' SF	(timer setting)
	X'2006'	X.25 Communications Error
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32D0'	Report the following
	X'82' SF	(DTE address called)
	X'82' SF	(DTE address calling)
	X'82' SF	(local DTE address)
	X'F0A0'	For
	X'82' SF	(locally-initiated logical channel)
		or
	X'82' SF	(remotely-initiated logical channel)
Additional SVs	X'52' SV	LCS Configuration Data
	X'07' SF	LCS Link Attributes
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(adapter name)
		Type=X'21' (adapter)
		Second resource below sender:
		Name=(port name)
		Type=X'3F' (port)

A.4.1.9 X.25 PLC Alert 9

Alert Condition:

Time-limit T23 expired at the DTE prior to receipt of a CLEAR_CONFIRMATION packet following transfer of a CLEAR_REQUEST packet.

Alert ID Number		X'F50A02F0'
Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error
Probable Causes	X'2050'	Packet Layer Control
	X'2008'	X.25 Communications
	X'2006'	X.25 Network

User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'20D1'	No response from the X.25 network --
	X'82' SF	(timer) expired
	X'82' SF	(retry count)
	X'82' SF	(timer setting)
	X'2006'	X.25 Communications Error
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32D0'	Report the following
	X'82' SF	(DTE address called)
	X'82' SF	(DTE address calling)
	X'82' SF	(local DTE address)
	X'F0A0'	For
	X'82' SF	(locally-initiated logical channel)
		or
	X'82' SF	(remotely-initiated logical channel)
Additional SVs	X'52' SV	LCS Configuration Data
	X'07' SF	LCS Link Attributes
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(adapter name)
		Type=X'21' (adapter)
		Second resource below sender:
		Name=(port name)
		Type=X'3F' (port)

A.4.1.10 X.25 PLC Alert 10

Alert Condition:

A packet level protocol violation, on the part of the PSDN access node, was detected by the DTE. A specific diagnostic code, indicating the source and the reason for the exception, has been reported to a higher layer function.

Alert ID Number		X'BA5D4659'
Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error
Probable Causes	X'2050'	Packet Layer Control

	X'2008'	X.25 Communications
	X'2006'	X.25 Network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'20B2'	X.25 Protocol Violation Detected
	X'82' SF	(diagnostic code)
	X'2006'	X.25 Communications Error
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32D0'	Report the following
	X'82' SF	(DTE address called)
	X'82' SF	(DTE address calling)
	X'82' SF	(local DTE address)
	X'F0A0'	For
	X'82' SF	(locally-initiated logical channel)
		or
	X'82' SF	(remotely-initiated logical channel)
Additional SVs	X'52' SV	LCS Configuration Data
	X'07' SF	LCS Link Attributes
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(adapter name)
		Type=X'21' (adapter)
		Second resource below sender:
		Name=(port name)
		Type=X'3F' (port)

A.4.1.11 X.25 PLC Alert 11

Alert Condition:

A DIAGNOSTIC packet, indicating a protocol violation on the part of the DTE, was sent by the PSDN or received by the DTE, or both.

Alert ID Number		X'4C323FE5'
Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error
Probable Causes	X'2050'	Packet Layer Control
	X'2008'	X.25 Communications

	X'2006'	X.25 Network
User Causes	(none)	
Install Causes	X'8000'	Configuration Error
Actions	X'1503'	Correct Configuration
Failure Causes	X'20C3'	X.25 Communications Error -- The following diagnostic packet was received from the network
	X'82' SF	(diagnostic code)
	X'82' SF	(diagnostic explanation)
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32A0'	Report the following
	X'82' SF	(local DTE address)
Additional SVs	X'52' SV	LCS Configuration Data
	X'07' SF	LCS Link Attributes
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(adapter name)
		Type=X'21' (adapter)
		Second resource below sender:
		Name=(port name)
		Type=X'3F' (port)

A.4.2 Link Access Protocol Balanced (LAPB)

Subtopics:

- A.4.2.1 X.25 LAPB Alert 1
- A.4.2.2 X.25 LAPB Alert 2
- A.4.2.3 X.25 LAPB Alert 3
- A.4.2.4 X.25 LAPB Alert 4
- A.4.2.5 X.25 LAPB Alert 5
- A.4.2.6 X.25 LAPB Alert 6
- A.4.2.7 X.25 LAPB Alert 7
- A.4.2.8 X.25 LAPB Alert 8
- A.4.2.9 X.25 LAPB Alert 9
- A.4.2.10 X.25 LAPB Alert 10
- A.4.2.11 X.25 LAPB Alert 11

A.4.2.1 X.25 LAPB Alert 1

Alert Condition:

The local station received a frame with an I-field which was too long and sent an FRMR.

Alert ID Number		X'07B1E788'

Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error
Probable Causes	X'2051'	Link Access Protocol Balanced
	X'8003'	Communication Configuration
	X'3500'	Communication Equipment
User Causes	(none)	
Install Causes	X'80C4'	Communication Configuration Error
	X'82' SF	(configuration object/record)
	X'82' SF	(configuration parameter)
Actions	X'1503'	Correct Configuration
Failure Causes	X'F023'	Received I-field exceeded maximum length
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32A0'	Report the following
	X'82' SF	(local DTE address)
Additional SVs	X'52' SV	LCS Configuration Data
	X'07' SF	LCS Link Attributes
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(adapter name)
		Type=X'21' (adapter)
		Second resource below sender:
		Name=(port name)
		Type=X'3F' (port)

A.4.2.2 X.25 LAPB Alert 2

Alert Condition:

The local station received a frame with an invalid N(R) value and sent an FRMR.

Alert ID Number		X'C0E4E919'
Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error

Probable Causes	X'2051'	Link Access Protocol Balanced
	X'3500'	Communication Equipment
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'F022'	Invalid N(R) received
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32A0'	Report the following
	X'82' SF	(local DTE address)
Additional SVs	X'52' SV	LCS Configuration Data
	X'07' SF	LCS Link Attributes
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(adapter name)
		Type=X'21' (adapter)
		Second resource below sender:
		Name=(port name)
		Type=X'3F' (port)

A.4.2.3 X.25 LAPB Alert 3

Alert Condition:

The local station has retried a command frame the maximum number of times without receiving the appropiate response. The local station enters disconnected state immediately.

Alert ID Number		X'A596712C' (1)
Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error
Probable Causes	X'2051'	Link Access Protocol Balanced
	X'8003'	Communication Configuration
	X'3401'	Local DCE Interface Cable
	X'3541'	Local DCE
	X'3500'	Communication Equipment
User Causes	(none)	
Install Causes	X'80C4'	Communication Configuration Error
	X'82' SF	(configuration object/record)

	X'82' SF	(configuration parameter)
Actions	X'1503'	Correct Configuration
Failure Causes	X'2006'	X.25 Communications Error
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32A0'	Report the following
	X'82' SF	(local DTE address)
Additional SVs	X'52' SV	LCS Configuration Data
	X'07' SF	LCS Link Attributes
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(adapter name)
		Type=X'21' (adapter)
		Second resource below sender:
		Name=(port name)
		Type=X'3F' (port)

Notes:

1. This Alert was previously documented with the incorrect Alert ID value of X'CEA222A9'. Alert receivers should be able to accept Alerts containing either the current, correct value or the old, incorrect value.

A.4.2.4 X.25 LAPB Alert 4

Alert Condition:

An unexpected DISC command frame was received during information transfer.

Alert ID Number		X'985806E2'
Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error
Probable Causes	X'2051'	Link Access Protocol Balanced
	X'3500'	Communication Equipment
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2006'	X.25 Communications Error

Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32A0'	Report the following
	X'82' SF	(local DTE address)
Additional SVs	X'52' SV	LCS Configuration Data
	X'07' SF	LCS Link Attributes
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(adapter name)
		Type=X'21' (adapter)
		Second resource below sender:
		Name=(port name)
		Type=X'3F' (port)

A.4.2.5 X.25 LAPB Alert 5

Alert Condition:

The local station sent SABM to initialize the link and received a DM frame.

Alert ID Number		X'E13004C9'
Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error
Probable Causes	X'2051'	Link Access Protocol Balanced
	X'230B'	Link Set Up Failure
	X'3500'	Communication Equipment
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2006'	X.25 Communications Error
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32A0'	Report the following
	X'82' SF	(local DTE address)
Additional SVs	X'52' SV	LCS Configuration Data
	X'07' SF	LCS Link Attributes
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List

		First resource below sender:
		Name=(adapter name)
		Type=X'21' (adapter)
		Second resource below sender:
		Name=(port name)
		Type=X'3F' (port)

A.4.2.6 X.25 LAPB Alert 6

Alert Condition:

A FRMR frame was received during information transfer state. The W-bit was on in the FRMR indicating receipt of a frame from the local station which had an invalid or unknown control field.

Alert ID Number		X'00891F75'
Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error
Probable Causes	X'2051'	Link Access Protocol Balanced
	X'3500'	Communication Equipment
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2006'	X.25 Communications Error
	X'F010'	Frame Reject Received: Invalid/unsupported command or response sent
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32A0'	Report the following
	X'82' SF	(local DTE address)
Additional SVs	X'52' SV	LCS Configuration Data
	X'07' SF	LCS Link Attributes
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(adapter name)
		Type=X'21' (adapter)
		Second resource below sender:
		Name=(port name)
		Type=X'3F' (port)

A.4.2.7 X.25 LAPB Alert 7

Alert Condition:

A FRMR frame was received during information transfer state. The X-bit was on in the FRMR indicating receipt of a frame from the local station which had an I-field which was not permitted.

Alert ID Number		X'CF6F806D'
Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error
Probable Causes	X'2051'	Link Access Protocol Balanced
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2006'	X.25 Communications Error
	X'F011'	Frame Reject Received: I-field sent when not permitted
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32A0'	Report the following
	X'82' SF	(local DTE address)
Additional SVs	X'52' SV	LCS Configuration Data
	X'07' SF	LCS Link Attributes
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(adapter name)
		Type=X'21' (adapter)
		Second resource below sender:
		Name=(port name)
		Type=X'3F' (port)

A.4.2.8 X.25 LAPB Alert 8

Alert Condition:

A FRMR frame was received during information transfer state. The Y-bit was on in the FRMR indicating receipt of a frame which had an oversized I-field.

Alert ID Number		X'F5E40347'
Alert Type	X'01'	Permanent

Alert Description	X'3320'	X.25 Error
Probable Causes	X'2051'	Link Access Protocol Balanced
User Causes	(none)	
Install Causes	X'80C4'	Communication Configuration Error
	X'82' SF	(configuration object/record)
	X'82' SF	(configuration parameter)
Actions	X'1503'	Correct Configuration
Failure Causes	X'2006'	X.25 Communications Error
	X'F013'	Frame Reject Received: Maximum I-field length exceeded
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32A0'	Report the following
	X'82' SF	(local DTE address)
Additional SVs	X'52' SV	LCS Configuration Data
	X'07' SF	LCS Link Attributes
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(adapter name)
		Type=X'21' (adapter)
		Second resource below sender:
		Name=(port name)
		Type=X'3F' (port)

A.4.2.9 X.25 LAPB Alert 9

Alert Condition:

A FRMR frame was received during information transfer state. The Z-bit was on in the FRMR indicating receipt of a frame which had an invalid N(R) specified.

Alert ID Number		X'C22CA6B4'
Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error
Probable Causes	X'2051'	Link Access Protocol Balanced

User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2006'	X.25 Communications Error
	X'F012'	Frame Reject Received: Invalid N(R) sent
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32A0'	Report the following
	X'82' SF	(local DTE address)
Additional SVs	X'52' SV	LCS Configuration Data
	X'07' SF	LCS Link Attributes
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(adapter name)
		Type=X'21' (adapter)
		Second resource below sender:
		Name=(port name)
		Type=X'3F' (port)

A.4.2.10 X.25 LAPB Alert 10

Alert Condition:

The local station received a frame with an invalid or unknown control field and sent an FRMR.

Alert ID Number		X'1F9CF04A'
Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error
Probable Causes	X'2051'	Link Access Protocol Balanced
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2006'	X.25 Communications Error
	X'F020'	Invalid/unsupported command or response received
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service

	X'32A0'	Report the following
	X'82' SF	(local DTE address)
Additional SVs	X'52' SV	LCS Configuration Data
	X'07' SF	LCS Link Attributes
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(adapter name)
		Type=X'21' (adapter)
		Second resource below sender:
		Name=(port name)
		Type=X'3F' (port)

A.4.2.11 X.25 LAPB Alert 11

Alert Condition:

The local station received a frame which had an I-field which was not permitted and sent an FRMR.

Alert ID Number		X'3FAE0180'
Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error
Probable Causes	X'2051'	Link Access Protocol Balanced
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2006'	X.25 Communications Error
	X'F021'	I-field received when not permitted
	X'1021'	Communication Controller Control Program
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32A0'	Report the following
	X'82' SF	(local DTE address)
Additional SVs	X'52' SV	LCS Configuration Data
	X'07' SF	LCS Link Attributes
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(adapter name)

		Type=X'21' (adapter)
		Second resource below sender:
		Name=(port name)
		Type=X'3F' (port)

A.4.3 Logical Link Control (LLC)

Note: For X.25 LLC protocol, protocol data units (PDUs) are the basic elements of information that are exchanged between logical link stations. For QLLC, a PDU = LLU (logical link unit), while for ELLC, a PDU = LPDU (LLC protocol data unit). When a FRMR response is mentioned in the following Alerts, it refers to an error condition of a PDU that is not recoverable, at the X.25 LLC layer, by retransmission of the identical PDU.

Subtopics:

- A.4.3.1 X.25 LLC Alert 1
- A.4.3.2 X.25 LLC Alert 2
- A.4.3.3 X.25 LLC Alert 3
- A.4.3.4 X.25 LLC Alert 4
- A.4.3.5 X.25 LLC Alert 5
- A.4.3.6 X.25 LLC Alert 6
- A.4.3.7 X.25 LLC Alert 7
- A.4.3.8 X.25 LLC Alert 8
- A.4.3.9 X.25 LLC Alert 9

A.4.3.1 X.25 LLC Alert 1

Alert Condition:

The local station received a Protocol Data Unit (PDU) with an I-field which was too long and sent an FRMR.

Alert ID Number		X'6460D9A9'
Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error
Probable Causes	X'2052'	Logical Link Control
	X'8003'	Communication Configuration
	X'3500'	Communication Equipment
User Causes	(none)	
Install Causes	X'80C4'	Communication Configuration Error
	X'82' SF	(configuration object/record)
	X'82' SF	(configuration parameter)
Actions	X'1503'	Correct Configuration
Failure Causes	X'F023'	Received I-field exceeded maximum length
Actions	X'3302'	If the problem continues to occur repeatedly then do the following

	X'3107'	Contact X.25 Network Information Service
	X'32D0'	Report the following
	X'82' SF	(DTE address called)
	X'82' SF	(DTE address calling)
	X'82' SF	(local DTE address)
	X'F0A0'	For
	X'82' SF	(locally-initiated logical channel)
		or
	X'82' SF	(remotely-initiated logical channel)
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(port name)
		Type=X'3F' (port)
		Second resource below sender:
		Name=(SNA control point name)
		Type=X'F4' (control point)

A.4.3.2 X.25 LLC Alert 2

Alert Condition:

The local station received a PDU with an invalid N(R) value and sent an FRMR.

Alert ID Number		X'CED07C9C'
Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error
Probable Causes	X'2052'	Logical Link Control
	X'3500'	Communication Equipment
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'F022'	Invalid N(R) received
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32D0'	Report the following
	X'82' SF	(DTE address called)
	X'82' SF	(DTE address calling)
	X'82' SF	(local DTE address)
	X'F0A0'	For

	X'82' SF	(locally-initiated logical channel)
		or
	X'82' SF	(remotely-initiated logical channel)
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(port name)
		Type=X'3F' (port)
		Second resource below sender:
		Name=(SNA control point name)
		Type=X'F4' (control point)

Note: This Alert applies only to ELLC.

A.4.3.3 X.25 LLC Alert 3

Alert Condition:

The local station has retried a command PDU the maximum number of times without receiving the appropiate response. The local station enters disconnected state immediately.

Alert ID Number		X'EAC7612A'
Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error
Probable Causes	X'2052'	Logical Link Control
	X'8003'	Communication Configuration
	X'2200'	Remote Node
User Causes	(none)	
Install Causes	X'80C4'	Communication Configuration Error
	X'82' SF	(configuration object/record)
	X'82' SF	(configuration parameter)
Actions	X'1503'	Correct Configuration
Failure Causes	X'2200'	Remote Node
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32D0'	Report the following
	X'82' SF	(DTE address called)

	X'82' SF	(DTE address calling)
	X'82' SF	(local DTE address)
	X'F0A0'	For
	X'82' SF	(locally-initiated logical channel)
		or
	X'82' SF	(remotely-initiated logical channel)
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(port name)
		Type=X'3F' (port)
		Second resource below sender:
		Name=(SNA control point name)
		Type=X'F4' (control point)

A.4.3.4 X.25 LLC Alert 4

Alert Condition:

A FRMR PDU was received during information transfer state. The W-bit was on in the FRMR indicating receipt of a PDU from the local station which had an invalid or unknown control field.

Alert ID Number		X'3DA4F8CD'
Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error
Probable Causes	X'2052'	Logical Link Control
	X'3500'	Communication Equipment
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2006'	X.25 Communications Error
	X'F010'	Frame Reject Received: Invalid/unsupported command or response sent
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32D0'	Report the following
	X'82' SF	(DTE address called)
	X'82' SF	(DTE address calling)
	X'82' SF	(local DTE address)
	X'F0A0'	For

	X'82' SF	(locally-initiated logical channel)
		or
	X'82' SF	(remotely-initiated logical channel)
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(port name)
		Type=X'3F' (port)
		Second resource below sender:
		Name=(SNA control point name)
		Type=X'F4' (control point)

A.4.3.5 X.25 LLC Alert 5

Alert Condition:

A FRMR PDU was received during information transfer state. The X-bit was on in the FRMR indicating receipt of a PDU from the local station which had an invalid I-field.

Alert ID Number		X'C15B15E8'
Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error
Probable Causes	X'2052'	Logical Link Control
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2006'	X.25 Communications Error
	X'F011'	Frame Reject Received: I-field sent when not permitted.
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32D0'	Report the following
	X'82' SF	(DTE address called)
	X'82' SF	(DTE address calling)
	X'82' SF	(local DTE address)
	X'F0A0'	For
	X'82' SF	(locally-initiated logical channel)
		or
	X'82' SF	(remotely-initiated logical channel)
Additional SVs	X'05' SV	Hierarchy/Resource List

	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(port name)
		Type=X'3F' (port)
		Second resource below sender:
		Name=(SNA control point name)
		Type=X'F4' (control point)

A.4.3.6 X.25 LLC Alert 6

Alert Condition:

A FRMR PDU was received during information transfer state. The Y-bit was on in the FRMR indicating receipt of a PDU which had an oversized I-field.

Alert ID Number		X'C8C9E4FF'
Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error
Probable Causes	X'2052'	Logical Link Control
User Causes	(none)	
Install Causes	X'80C4'	Communication Configuration Error
	X'82' SF	(configuration object/record)
	X'82' SF	(configuration parameter)
Actions	X'1503'	Correct Configuration
Failure Causes	X'2006'	X.25 Communications Error
	X'F013'	Frame Reject Received: Maximum I-field length exceeded
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32D0'	Report the following
	X'82' SF	(DTE address called)
	X'82' SF	(DTE address calling)
	X'82' SF	(local DTE address)
	X'F0A0'	For
	X'82' SF	(locally-initiated logical channel)
		or
	X'82' SF	(remotely-initiated logical channel)
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List

		First resource below sender:
		Name=(port name)
		Type=X'3F' (port)
		Second resource below sender:
		Name=(SNA control point name)
		Type=X'F4' (control point)

A.4.3.7 X.25 LLC Alert 7

Alert Condition:

A FRMR PDU was received during information transfer state. The Z-bit was on in the FRMR indicating receipt of a PDU which had an invalid N(R) specified.

Alert ID Number		X'CC183331'
Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error
Probable Causes	X'2052'	Logical Link Control
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2006'	X.25 Communications Error
	X'F012'	Frame Reject Received: Invalid N(R) sent
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32D0'	Report the following
	X'82' SF	(DTE address called)
	X'82' SF	(DTE address calling)
	X'82' SF	(local DTE address)
	X'F0A0'	For
	X'82' SF	(locally-initiated logical channel)
		or
	X'82' SF	(remotely-initiated logical channel)
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(port name)
		Type=X'3F' (port)
		Second resource below sender:
		Name=(SNA control point name)

		Type=X'F4' (control point)

Note: This Alert applies only to ELLC.

A.4.3.8 X.25 LLC Alert 8

Alert Condition:

The local station received a PDU with an invalid or unknown control field and sent an FRMR.

Alert ID Number		X'11A865CF'
Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error
Probable Causes	X'2052'	Logical Link Control
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2006'	X.25 Communications Error
	X'F020'	Invalid/unsupported command or response received
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32D0'	Report the following
	X'82' SF	(DTE address called)
	X'82' SF	(DTE address calling)
	X'82' SF	(local DTE address)
	X'F0A0'	For
	X'82' SF	(locally-initiated logical channel)
		or
	X'82' SF	(remotely-initiated logical channel)
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(port name)
		Type=X'3F' (port)
		Second resource below sender:
		Name=(SNA control point name)
		Type=X'F4' (control point)

A.4.3.9 X.25 LLC Alert 9

Alert Condition:

The local station received a PDU with an invalid I-field and sent an FRMR.

Alert ID Number		X'0283E638'
Alert Type	X'01'	Permanent
Alert Description	X'3320'	X.25 Error
Probable Causes	X'2052'	Logical Link Control
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2006'	X.25 Communications Error
	X'F021'	I-field received when not permitted
	X'1021'	Communication Controller Control Program
Actions	X'3302'	If the problem continues to occur repeatedly then do the following
	X'3107'	Contact X.25 Network Information Service
	X'32D0'	Report the following
	X'82' SF	(DTE address called)
	X'82' SF	(DTE address calling)
	X'82' SF	(local DTE address)
	X'F0A0'	For
	X'82' SF	(locally-initiated logical channel)
		or
	X'82' SF	(remotely-initiated logical channel)
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender:
		Name=(port name)
		Type=X'3F' (port)
		Second resource below sender:
		Name=(SNA control point name)
		Type=X'F4' (control point)

A.4.4 ISDN Alerts

Subtopics:

- A.4.4.1 B- and D-Channel Names in ISDN Alerts

- A.4.4.2 Channel Names in the Hierarchy Name List (X'10') Subfield
- A.4.4.3 Channel Names in the Associated Resources (X'11') Subfield
- A.4.4.4 ISDN Basic and Primary Rate Physical Layer Alerts
- A.4.4.5 ISDN Physical Layer Alert 1
- A.4.4.6 ISDN Physical Layer Alert 2
- A.4.4.7 ISDN Physical Layer Alert 3
- A.4.4.8 ISDN Physical Layer Alert 4
- A.4.4.9 ISDN Physical Layer Alert 5
- A.4.4.10 ISDN Physical Layer Alert 6
- A.4.4.11 ISDN Physical Layer Alert 7
- A.4.4.12 ISDN Physical Layer Alert 8
- A.4.4.13 ISDN Physical Layer Alert 9
- A.4.4.14 ISDN Physical Layer Alert 10
- A.4.4.15 ISDN Physical Layer Alert 11
- A.4.4.16 ISDN Physical Layer Alert 12
- A.4.4.17 ISDN Physical Layer Alert 13
- A.4.4.18 ISDN R-Interface Alerts
- A.4.4.19 ISDN R-interface Alert 1
- A.4.4.20 ISDN Layer 2 B-Channel DLC Alerts
- A.4.4.21 ISDN B-Channel DLC Alert 1
- A.4.4.22 ISDN B-Channel DLC Alert 2
- A.4.4.23 ISDN B-Channel DLC Alert 3
- A.4.4.24 ISDN B-Channel DLC Alert 4
- A.4.4.25 ISDN B-Channel DLC Alert 5
- A.4.4.26 ISDN B-Channel DLC Alert 6
- A.4.4.27 ISDN B-Channel DLC Alert 7
- A.4.4.28 ISDN B-Channel DLC Alert 8
- A.4.4.29 ISDN B-Channel DLC Alert 9
- A.4.4.30 ISDN B-Channel DLC Alert 10
- A.4.4.31 ISDN B-Channel DLC Alert 11
- A.4.4.32 ISDN B-Channel DLC Alert 12
- A.4.4.33 ISDN B-Channel DLC Alert 13
- A.4.4.34 ISDN B-Channel DLC Alert 14
- A.4.4.35 ISDN B-Channel DLC Alert 15
- A.4.4.36 ISDN B-Channel DLC Alert 16
- A.4.4.37 ISDN B-Channel DLC Alert 17
- A.4.4.38 ISDN Layer 2 D-Channel LAPD Alerts
- A.4.4.39 ISDN D-Channel LAPD Alert 1
- A.4.4.40 ISDN D-Channel LAPD Alert 2
- A.4.4.41 ISDN D-Channel LAPD Alert 3
- A.4.4.42 ISDN D-Channel LAPD Alert 4
- A.4.4.43 ISDN D-Channel LAPD Alert 5
- A.4.4.44 ISDN D-Channel LAPD Alert 6
- A.4.4.45 ISDN D-Channel LAPD Alert 7
- A.4.4.46 ISDN D-Channel LAPD Alert 8
- A.4.4.47 ISDN D-Channel LAPD Alert 9
- A.4.4.48 ISDN D-Channel LAPD Alert 10
- A.4.4.49 ISDN D-Channel LAPD Alert 11
- A.4.4.50 ISDN D-Channel LAPD Alert 12
- A.4.4.51 ISDN D-Channel LAPD Alert 13
- A.4.4.52 ISDN D-Channel LAPD Alert 14
- A.4.4.53 ISDN D-Channel LAPD Alert 15
- A.4.4.54 ISDN D-Channel LAPD Alert 16
- A.4.4.55 ISDN D-Channel LAP-D Alert 17
- A.4.4.56 ISDN Layer 3 Q.931 Call Control Alerts
- A.4.4.57 Q.931 Alert 1
- A.4.4.58 Q.931 Alert 2
- A.4.4.59 Q.931 Alert 3
- A.4.4.60 Q.931 Alert 4
- A.4.4.61 Q.931 Alert 5
- A.4.4.62 Q.931 Alert 6
- A.4.4.63 Q.931 Alert 7
- A.4.4.64 Q.931 Alert 8
- A.4.4.65 Q.931 Alert 9
- A.4.4.66 Q.931 Alert 10
- A.4.4.67 Q.931 Alert 11
- A.4.4.68 Q.931 Alert 12
- A.4.4.69 Q.931 Alert 13
- A.4.4.70 Q.931 Alert 14
- A.4.4.71 Q.931 Alert 15
- A.4.4.72 Q.931 Alert 16
- A.4.4.73 Q.931 Alert 17
- A.4.4.74 Q.931 Alert 18
- A.4.4.75 Q.931 Alert 19
- A.4.4.76 Q.931 Alert 20

A.4.4.1 B- and D-Channel Names in ISDN Alerts

```
Names for ISDN B- and D-channels appear in two places in the ISDN Alerts
that follow:

°   The D-channel name appears in the Hierarchy Name List (X'10') subfield
    of the Hierarchy/Resource List (X'05') subvector.

°   Both D-channel and B-channel names appear in the Associated Resources
    (X'11') subfield of the Hierarchy/Resource (X'05') subvector.

Because of the different formats for these two subfields, and the
different roles they play at the Alert receiver, different forms of the
channel names are used in the two of them.
```

A.4.4.2 Channel Names in the Hierarchy Name List (X'10') Subfield

Since names in the Hierarchy Name List subfield are limited to 8 characters each, there is no way to make them globally unique throughout a network. This does not present a problem, though, since a name from this subfield is always interpreted within the context of a hierarchy of names above it.

A sender thus makes the D-channel name in this subfield unique only within that one sender. The format of this name is determined by each Alert sender. One possible format is:

```
A<adapter number>P<port number>C<channel number>
```

e.g., A1P2C01 might identify adapter 1, port 2, channel 1 within the sender.

While it is not required, a sender might find it convenient to use the same convention for this D-channel name that it uses for the last element of the globally-unique channel names used in the Associated Resources subfield. These names are described in the next section.

A.4.4.3 Channel Names in the Associated Resources (X'11') Subfield

In order to allow for proper correlation of ISDN Alerts, names of the D- and B-channels associated with an Alert condition are included in the Associated Resources (X'11') subfield of the Hierarchy/Resource List (X'05') subvector within the Alert. Since these names are compared to each other without regard to their location in a hierarchy, the names themselves must be unique within the network.

To insure its uniqueness in the network, each channel name has the following form:

```
TTTT[MMM]SSSSSSSID
```

where:

- TTTT is the machine type from the first hardware product identifier in the Alert sender's PSID.
- MMM is the model number from the first hardware product identifier in the Alert sender's PSID; this element is omitted if there is no model number present in this product identifier.
- SSSSSSS is the serial number from the first hardware product identifier in the Alert sender's PSID.
- ID is a value that uniquely identifies a channel within an instance of an Alert sender. The format of this element is determined by each Alert sender. One possible format is:

  ```
  A<adapter number>P<port number>C<channel number>
  ```

The overall length of the channel name is limited to 56 characters.

A.4.4.4 ISDN Basic and Primary Rate Physical Layer Alerts

A.4.4.5 ISDN Physical Layer Alert 1

Alert Condition:

ISDN Adapter Lost Frame Alignment - This Alert is generated when the Loss of Frame Alignment (X'0001') counter has reached its threshold. This indicates that there is a synchronization problem between TE and NT, even though the synchronization was recovered, the channel capacity loss due to the problem is not acceptable. This Alert applies to both Basic and Primary Rate ISDN.

Alert ID Number		X'BBF58700'

Alert Type	X'03'	Performance
Alert Description	X'4012'	Threshold has been reached
Probable Causes	X'3530'	ISDN Network Component
	X'3310'	Local ISDN Adapter
User Causes	(none)	
Install Causes	X'3405'	Local communication cable not properly connected
Actions	X'0301'	Check cable and its connections
Failure Causes	X'3536'	ISDN Network Termination Device
	X'3310'	Local ISDN Adapter
	X'40A0'	(sf 82) threshold reached
	X'82' SF	Counter (1)
Actions	X'3301'	If problem persists then do the following
	X'2002'	(review most recent traffic statistics)
	X'3109'	Contact personnel responsible for connection to ISDN network
	X'32D0'	Report the following
	X'82' SF	Adapter number
	X'82' SF	Port number
	X'82' SF	Telephone number (3)
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (2))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (2))
		Type = X'FA'(D-channel)

---

(1) The hexadecimal table index of the counter name.

(2) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

(3) The telephone number associated with the port.

A.4.4.6 ISDN Physical Layer Alert 2

Alert Condition:

ISDN Adapter Lost Synchronization - This Alert is generated when the synchronization is lost between the TE and the NT. A permanent loss is declared when the alignment has not occurred in a specified time. This Alert applies to both Basic and Primary Rate ISDN.

Alert ID Number		X'55DF0F61'
Alert Type	X'01'	Permanent
Alert Description	X'3404'	ISDN Physical Layer error
Probable Causes	X'3530'	ISDN Network Component
	X'3310'	Local ISDN Adapter
User Causes	(none)	
Install Causes	X'3405'	Local communication cable not properly connected
Actions	X'0301'	Check cable and its connections
Failure Causes	X'F06F'	Synchronization lost and not recovered
	X'3530'	ISDN Network Component
	X'3310'	Local ISDN Adapter
Actions	X'3301'	If problem persists, then do the following:
	X'3109'	Contact personnel responsible for connection to ISDN network
	X'32D0'	Report the following
	X'82' SF	Adapter number
	X'82' SF	Port number
	X'82' SF	Telephone number (5)
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (4))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (4))
		Type = X'FA'(D-channel)

---

(4) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

(5) The telephone number associated with the port.

A.4.4.7 ISDN Physical Layer Alert 3

Alert Condition:

Excessive Transmission Error Detected by NT1 for Basic ISDN - This Alert is generated when the Detected Access Transmission System Errors (DTSE-In, X'0002') counter has reached its threshold. This indicates that there are excessive link errors detected by the NT1 across the U interface.

Alert ID Number		X'2F24E652'
Alert Type	X'03'	Performance
Alert Description	X'4012'	Threshold has been reached
Probable Causes	X'3462'	Local DCE loop
	X'3531'	ISDN Network Termination Equipment (NT1)
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'200E'	Local DCE loop
	X'3531'	ISDN Network Termination Equipment (NT1)
	X'40A0'	(sf 82) threshold reached
	X'82' SF	Counter (6)
Actions	X'3301'	If problem persists then do the following
	X'2002'	(review most recent traffic statistics)
	X'3106'	Contact ISDN network information service
	X'32A0'	Report the following
	X'82' SF	Telephone number (8)
	X'3109'	Contact personnel responsible for connection to ISDN network
	X'32C0'	Report the following
	X'82' SF	Adapter number
	X'82' SF	Port number
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (7))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (7))
		Type = X'FA'(D-channel)

(6) The hexadecimal table index of the counter name.

(7) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

(8) The telephone number associated with the port.

A.4.4.8 ISDN Physical Layer Alert 4

Alert Condition:

Excessive Transmission Errors Were Detected by the Network for Basic ISDN - This Alert is generated when the Detected Access Transmission System Errors (DTSE-Out, X'0003') counter has reached its threshold. This indicates that there are excessive link errors detected by the carrier across the U interface.

Alert ID Number		X'94E915B8'
Alert Type	X'03'	Performance
Alert Description	X'4012'	Threshold has been reached
Probable Causes	X'3531'	ISDN Network Termination Equipment (NT1)
	X'3462'	Local DCE loop
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3531'	ISDN Network Termination Equipment (NT1)
	X'200E'	Local DCE loop
	X'40A0'	(sf 82) threshold reached
	X'82' SF	Counter (9)
Actions	X'3301'	If problem persists then do the following
	X'2002'	(review most recent traffic statistics)
	X'3109'	Contact personnel responsible for connection to ISDN network
	X'32C0'	Report the following
	X'82' SF	Adapter number
	X'82' SF	Port number
	X'3106'	Contact ISDN network information service
	X'32A0'	Report the following
	X'82' SF	Telephone number (11)
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (10))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (10))
		Type = X'FA'(D-channel)

(9) The hexadecimal table index of the counter name.

(10) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

(11) The telephone number associated with the port.

A.4.4.9 ISDN Physical Layer Alert 5

Alert Condition:

NT1 power off - This Alert indicates that there has been a power loss in the NT1 - The ISDN Basic Rate Standard specifies that the NT1 notify the attaching DTE when it is losing power. This condition may be detected in some other manner for Primary Rate ISDN where an NT1 is used as an interface device. This Alert applies to both Basic and Primary Rate ISDN.

Alert ID Number		X'B349DD7C'
Alert Type	X'01'	Permanent
Alert Description	X'1411'	Power off detected
Probable Causes	X'3531'	ISDN Network Termination Equipment (NT1)
User Causes	X'2115'	Local NT1 power off
Actions	X'0200'	Check power
Install Causes	(none)	
Failure Causes	X'3531'	ISDN Network Termination Equipment (NT1)
Actions	X'3301'	If problem persists then do the following
	X'3109'	Contact personnel responsible for connection to ISDN network
	X'32C0'	Report the following
	X'82' SF	Adapter number
	X'82' SF	Port number
	X'3106'	Contact ISDN network information service
	X'32A0'	Report the following
	X'82' SF	Telephone number (13)
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (12))
		Type = X'FA'(D-channel)

	X'11' SF	Associated Resources
		Name = (D-channel (12))
		Type = X'FA'(D-channel)

---

(12) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

(13) The telephone number associated with the port.

A.4.4.10 ISDN Physical Layer Alert 6

Alert Condition:

Received Signal Lost - This Alert indicates a problem in the U-interface because no signals are being received at the NT1 from the carrier. This condition may also be detected by DTE's that attach directly to the ISDN network interface. This Alert applies to both Basic and Primary Rate ISDN.

Alert ID Number		X'76190DF4'
Alert Type	X'01'	Permanent
Alert Description	X'3619'	No line signal
Probable Causes	X'3462'	Local DCE loop
	X'200A'	ISDN Network
User Causes	(none)	
Install Causes	X'3405'	Local communication cable not properly connected
Actions	X'0305'	Check cable and its connections
Failure Causes	X'200E'	Local DCE loop
	X'200A'	ISDN Network
Actions	X'3301'	If problem persists then do the following
	X'3106'	Contact ISDN network information service
	X'32C0'	Report the following
	X'82' SF	Adapter number
	X'82' SF	Telephone number (15)
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (14))
		Type = X'FA'(D-channel)

	X'11' SF	Associated Resources
		Name = (D-channel (14))
		Type = X'FA'(D-channel)

---

(14) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

(15) The telephone number associated with the port.

A.4.4.11 ISDN Physical Layer Alert 7

Alert Condition:

Excessive Unintended Code Violation were Detected by the NT1 for Basic ISDN. This Alert is generated when the TE-Detected Code Violations (X'0004') counter has reached its threshold. This indicates that there are excessive link errors between the TE and the NT1.

Alert ID Number		X'1A08523C'
Alert Type	X'03'	Performance
Alert Description	X'4012'	Threshold has been reached
Probable Causes	X'3310'	Local ISDN adapter
	X'3531'	ISDN Network Termination Equipment (NT1)
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3310'	Local ISDN adapter
	X'3531'	ISDN Network Termination Equipment (NT1)
	X'40A0'	(sf 82) threshold reached
	X'82' SF	Counter (16)
Actions	X'3301'	If problem persists then do the following
	X'2002'	(review most recent traffic statistics)
	X'3109'	Contact personnel responsible for connection to ISDN network
	X'32D0'	Report the following
	X'82' SF	Adapter number
	X'82' SF	Port number
	X'82' SF	Telephone number (18)
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender

		Name = (D-channel (17))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (17))
		Type = X'FA'(D-channel)

---

(16) The hexadecimal table index of the counter name.

(17) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

(18) The telephone number associated with the port.

A.4.4.12 ISDN Physical Layer Alert 8

Alert Condition:

Excessive Code Violation as Detected by TE for Basic ISDN. This Alert is generated when the NT-Detected Code Violations (X'0005') counter has reached its threshold. This indicates that there are excessive link errors between the TE and the NT1.

Alert ID Number		X'1FA96E3D'
Alert Type	X'03'	Performance
Alert Description	X'4012'	Threshold has been reached
Probable Causes	X'3531'	ISDN Network Termination Equipment (NT1)
	X'3310'	Local ISDN adapter
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3531'	ISDN Network Termination Equipment (NT1)
	X'3310'	Local ISDN adapter
	X'40A0'	(sf 82) threshold reached
	X'82' SF	Counter (19)
Actions	X'3301'	If problem persists then do the following
	X'2002'	(review most recent traffic statistics)
	X'3109'	Contact personnel responsible for connection to ISDN network
	X'32D0'	Report the following
	X'82' SF	Adapter number
	X'82' SF	Port number
	X'82' SF	Telephone number (21)

Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (20))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (20))
		Type = X'FA'(D-channel)

---

(19) The hexadecimal table index of the counter name.

(20) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

(21) The telephone number associated with the port.

A.4.4.13 ISDN Physical Layer Alert 9

Alert Condition:

Excessive Transmission Errors Were Detected by the TE for Primary ISDN. This condition indicates that there are excessive link errors detected by the TE across the ISDN primary interface.

This Alert is generated for ISDN Primary Rate when the following counters used to detect transmission problems have reached their thresholds: CRC Errors Received (X'0006'), Frame Loss Seconds Near-End (X'0038'), Errored Seconds Near-End (X'003C'), Bursty Errored Seconds Near-End (X'003E'), Severely Errored Seconds Near-End (X'0040'), Slip Seconds Near-End (X'0042').

This Alert may be used by Basic Rate ISDN devices that have a built-in NT1 function to report degradation indicated by the following counters: CRC Errors Received (X'0006'), Errored Seconds Near-End (X'003C'), Severely Errored Seconds Near-End (X'0040').

Alert ID Number		X'1104601D' (22)
Alert Type	X'03'	Performance
Alert Description	X'4012'	Threshold has been reached
Probable Causes	X'200A'	ISDN Network
	X'3310'	Local ISDN adapter
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'200A'	ISDN Network
	X'3310'	Local ISDN adapter
	X'40A0'	(sf 82) threshold reached
	X'82' SF	Counter (23)

Actions	X'3301'	If problem persists then do the following
	X'2002'	(review most recent traffic statistics)
	X'3106'	Contact ISDN network information service
	X'32A0'	Report the following
	X'82' SF	Telephone number (25)
	X'3109'	Contact personnel responsible for connection to ISDN network
	X'32C0'	Report the following
	X'82' SF	Adapter number
	X'82' SF	Port number
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (24))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (24))
		Type = X'FA'(D-channel)

| (22) This Alert was previously documented with the incorrect | Alert ID value of X'A276395A'. Alert receivers should be | able to accept Alerts containing either the current, correct | value or the old, incorrect value.

(23) The hexadecimal table index of the counter name.

(24) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

(25) The telephone number associated with the port.

A.4.4.14 ISDN Physical Layer Alert 10

Alert Condition:

Excessive Transmission Errors Were Detected by the carrier or network provider for Primary ISDN. This condition indicates that there are excessive link errors detected by the network provider and signaled to the TE across the ISDN primary interface.

This Alert is generated when the following counters used to detect transmission have reached their thresholds: CRC Errors Transmitted (X'0007'), Frame Loss Seconds Far-End (X'0039'), Errored Seconds Far-End (X'003D'), Bursty Errored Seconds Far-End (X'003F'), Severely Errored Seconds Far-End (X'0041'), Slip Seconds Far-End (X'0043').

This Alert can be used by Basic Rate ISDN devices that have a built-in NT1 function to report degradation indicated by the following counters: CRC Errors Transmitted (X'0007'), Errored Seconds Far-End (X'003D'), Severely Errored Seconds Far-End (X'0041').

Alert ID Number		X'EEF3487E'
Alert Type	X'03'	Performance
Alert Description	X'4012'	Threshold has been reached

Probable Causes	X'3310'	Local ISDN adapter
	X'200A'	ISDN Network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3310'	Local ISDN adapter
	X'200A'	ISDN Network
	X'40A0'	(sf 82) threshold reached
	X'82' SF	Counter (26)
Actions	X'3301'	If problem persists then do the following
	X'2002'	(review most recent traffic statistics)
	X'3109'	Contact personnel responsible for connection to ISDN network
	X'32C0'	Report the following
	X'82' SF	Adapter number
	X'82' SF	Port number
	X'3106'	Contact ISDN network information service
	X'32A0'	Report the following
	X'82' SF	Telephone number (28)
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (27))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (27))
		Type = X'FA'(D-channel)

---

(26) The hexadecimal table index of the counter name.

(27) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

(28) The telephone number associated with the port.

A.4.4.15 ISDN Physical Layer Alert 11

Alert Condition:

NT1 Activation Failure to Network - This Alert is generated when NT1 connection to the network could not be achieved. This Alert applies to Basic Rate ISDN but may also apply in situations where an NT1 is used for Primary Rate ISDN.

Alert ID Number		X'E5B91C0F'
Alert Type	X'01'	Permanent
Alert Description	X'3405'	ISDN Physical Layer Activation Error
Probable Causes	X'3462'	Local DCE loop
	X'3531'	ISDN Network Termination Equipment (NT1)
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'200E'	Local DCE loop
	X'3531'	ISDN Network Termination Equipment (NT1)
Actions	X'1206'	Wait and retry
	X'3301'	If problem persists then do the following
	X'3106'	Contact ISDN network information service
	X'32A0'	Report the following
	X'82' SF	Telephone number (30)
	X'3109'	Contact personnel responsible for connection to ISDN network
	X'32C0'	Report the following
	X'82' SF	Adapter number
	X'82' SF	Port number
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (29))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (29))
		Type = X'FA'(D-channel)

(29) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

(30) The telephone number associated with the port.

A.4.4.16 ISDN Physical Layer Alert 12

Alert Condition:

TE Activation Failure to NT1 - This Alert is generated when TE connection to the NT1 could not be achieved. This Alert applies to Basic Rate ISDN but may also apply in situations where an NT1 is used for Primary Rate

ISDN.

Alert ID Number		X'64BBFE11'
Alert Type	X'01'	Permanent
Alert Description	X'3405'	ISDN Physical Layer Activation Error
Probable Causes	X'3531'	ISDN Network Termination Equipment (NT1)
	X'3463'	Premises wiring
User Causes	(none)	
Install Causes	X'3400'	Cabling installed incorrectly
Actions	X'0301'	Check cable and its connection
Failure Causes	X'3531'	ISDN Network Termination Equipment (NT1)
	X'3463'	Premises wiring
Actions	X'3109'	Contact personnel responsible for connection to ISDN network
	X'32D0'	Report the following
	X'82' SF	Adapter number
	X'82' SF	Port number
	X'82' SF	Telephone number (32)
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (31))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (31))
		Type = X'FA'(D-channel)

---

(31) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

(32) The telephone number associated with the port.

A.4.4.17 ISDN Physical Layer Alert 13

Alert Condition:

TE Activation Failure to Network - This Alert is generated when TE connection to the network could not be achieved. This Alert applies to Basic Rate and Primary Rate ISDN in situations where the NT1 function is incorporated in the ISDN TE.

Alert ID Number		X'2520F7B1'
Alert Type	X'01'	Permanent
Alert Description	X'3405'	ISDN Physical Layer Activation Error
Probable Causes	X'3462'	Local DCE loop
	X'200A'	ISDN Network
	X'3310'	Local ISDN Adapter
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'200E'	Local DCE loop
	X'200A'	ISDN Network
	X'3310'	Local ISDN Adapter
Actions	X'1206'	Wait and retry
	X'3301'	If problem persists then do the following
	X'3106'	Contact ISDN network information service
	X'32A0'	Report the following
	X'82' SF	Telephone number (34)
	X'3109'	Contact personnel responsible for connection to ISDN network
	X'32C0'	Report the following
	X'82' SF	Adapter number
	X'82' SF	Port number
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (33))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (29))
		Type = X'FA'(D-channel)

(33) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

(34) The telephone number associated with the port.

A.4.4.18 ISDN R-Interface Alerts

A.4.4.19 ISDN R-interface Alert 1

Alert Condition:

```
Excessive R-interface Errors - This Alert is generated when the
R-interface TA Asynchronous Error (X'000E') counter or the R-interface TA
Synchronous Error (X'000F') counter has reached its threshold.  This
indicates that there are excessive link errors between two R-interface
stations.
```

Alert ID Number		X'248A30D8'
Alert Type	X'03'	Performance
Alert Description	X'4012'	Threshold has been reached
Probable Causes	X'3225'	ISDN R-interface
	X'2052'	Logical Link Control
	X'3309'	Line Adapter
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3225'	ISDN R-interface
	X'40A0'	(sf 82) threshold reached
	X'82' SF	Counter (35)
Actions	X'3301'	If problem persists, then do the following:
	X'3109'	Contact personnel responsible for connection to ISDN network
	X'32C0'	Report the following
	X'82' SF	Adapter number
	X'82' SF	Port number
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (36))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (36))
		Type = X'FA'(D-channel)
		Name = (B-channel (36))
		Type = X'FB'(B-channel)

---

(35) The hexadecimal table index of the counter name.

(36) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.20 ISDN Layer 2 B-Channel DLC Alerts

The X'04' Subvector in ISDN Hierarchies

The hierarchies described in this section assume an APPN Alert sender. More precisely, they assume that the Alert sender knows the CP name of the node at the remote end of the B-channel to which the Alert applies, so that it can include this name as the second resource below the Alert sender in the X'10' subfield of the Hierarchy/Resource List. If the Alert sender does not know this name, if it is sending the Alert over an SSCP-PU session, and if the SSCP to which the Alert is being sent also knows the name of the node at the remote end of the B-channel, then the hierarchy is handled as follows:

- The SNA local address of the remote node is sent in the SNA Address List (X'04') subvector; the SSCP will perform address-to-name translation for this address.
- The 8-character locally-unique name of the D-channel associated with the failing B-channel is sent as the only entry in the Hierarchy Name List (X'10') subfield of the Hierarchy/Resource List (X'05') subvector. Special processing is required in the Alert receiver to place this entry above that derived from the SNA Address List, since it would ordinarily appear below it.
- The two entries in the Associated Resources (X'11') subfield of the Hierarchy/Resource List subvector are sent as shown.

The only ISDN Alerts that require the SNA Address List are the B-channel DLC Alerts.

A.4.4.21 ISDN B-Channel DLC Alert 1

Alert Condition:

Excessive B-Channel Receive Errors - This Alert is generated when the CRC Errors Received (X'0006') counter has reached its threshold. This indicates that there are excessive link errors between two DLC stations across the B channel.

Alert ID Number		X'2040E8A6'
Alert Type	X'03'	Performance
Alert Description	X'4012'	Threshold has been reached
Probable Causes	X'200A'	ISDN network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2056'	B-Channel ISDN error
	X'200A'	ISDN communications error
	X'40A0'	(sf 82) threshold reached
	X'82' SF	Counter (37)
Actions	X'3301'	If problem persists then do the following:
	X'2010'	(review link detail data)
	X'2002'	(review most recent traffic statistics)
	X'3106'	Contact ISDN network information service

Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes
	X'09' SF	Remote Telephone Number
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (38))
		Type = X'FA'(D-channel)
		Second resource below sender
		Name = (adjacent node)
		Type = X'F4'(CP)
	X'11' SF	Associated Resources
		Name = (D-channel (38))
		Type = X'FA'(D-channel)
		Name = (B-channel (38))
		Type = X'FB'(B-channel)

---

(37) The hexadecimal table index of the counter name.

(38) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.22 ISDN B-Channel DLC Alert 2

Alert Condition:

Excessive B-Channel Link Station Errors - This Alert is generated when either of the following station error counters have reached their thresholds: the PDUs Retransmitted (X'0017') or the Receive Sequence Errors (X'0018'). This indicates that there are excessive link errors between two DLC stations across the B channel.

Alert ID Number		X'79931598'
Alert Type	X'03'	Performance
Alert Description	X'4012'	Threshold has been reached
Probable Causes	X'210A'	ISDN Communications/remote node
	X'200A'	ISDN network
User Causes	(none)	
Install Causes	(none)	

Failure Causes	X'210A'	ISDN Communications/remote node
	X'1023'	Communications program in remote node
	X'2056'	B-Channel ISDN error
	X'200A'	ISDN communications error
	X'40A0'	(sf 82) threshold reached
	X'82' SF	Counter (39)
Actions	X'3301'	If problem persists then do the following:
	X'2010'	(review link detail data)
	X'2002'	(review most recent traffic statistics)
	X'3106'	Contact ISDN network information service
	X'3122'	Contact called DTE's operator
Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes
	X'09' SF	Remote Telephone Number
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (40))
		Type = X'FA'(D-channel)
		Second resource below sender
		Name = (adjacent node)
		Type = X'F4'(CP)
	X'11' SF	Associated Resources
		Name = (D-channel (40))
		Type = X'FA'(D-channel)
		Name = (B-channel (40))
		Type = X'FB'(B-channel)

(39) The hexadecimal table index of the counter name.

(40) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.23 ISDN B-Channel DLC Alert 3

Alert Condition:

Excessive B-Channel Invalid Frame Errors - This Alert is generated when the count of a type of invalid frame on the B-Channel has reached the assigned threshold. The invalid frame category includes the following: Short Frames Received (X'000B'), Misaddressed Frames Received (X'0012'), Unbounded Frames Received (X'0013'), Non-Integral Frames Received (X'0014'), Aborted Frames Received (X'0010'), Aborted Frames Transmitted (X'0011'), and the summary counter Badly Formed Frames (X'0044'). This indicates that either there are excessive link errors between two DLC stations across the B-Channel or the remote node is failing.

Alert ID Number		X'A77C91CB'
Alert Type	X'03'	Performance
Alert Description	X'4012'	Threshold has been reached
Probable Causes	X'210A'	ISDN Communications/remote node
	X'200A'	ISDN network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3311'	Remote ISDN Adapter
	X'200A'	ISDN communications error
	X'40A0'	(sf 82) threshold reached
	X'82' SF	Counter (41)
Actions	X'3301'	If problem persists then do the following:
	X'2010'	(review link detail data)
	X'2002'	(review most recent traffic statistics)
	X'3106'	Contact ISDN network information service
	X'3122'	Contact called DTE's operator
Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes
	X'09' SF	Remote Telephone Number
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (42))
		Type = X'FA'(D-channel)
		Second resource below sender
		Name = (adjacent node)
		Type = X'F4'(CP)
	X'11' SF	Associated Resources
		Name = (D-channel (42))

		Type = X'FA'(D-channel)
		Name = (B-channel (42))
		Type = X'FB'(B-channel)

(41) The hexadecimal table index of the counter name.

(42) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.24 ISDN B-Channel DLC Alert 4

Alert Condition:

ISDN Adapter Buffer Overrun/Underrun - Either the Buffer Overrun (X'000C') or Buffer Underrun (X'000D') counter has reached its threshold.

Alert ID Number		X'AD564CF0'
Alert Type	X'03'	Performance
Alert Description	X'4012'	Threshold has been reached
Probable Causes	X'3310'	Local ISDN adapter
	X'3222'	ISDN adapter interface
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3310'	Local ISDN adapter
	X'40A0'	(sf 82) threshold reached
	X'82' SF	Counter (43)
Actions	X'3301'	If problem persists, then do the following:
	X'2002'	(review most recent traffic statistics)
	X'3109'	Contact personnel responsible for connection to ISDN network
	X'32D0'	Report the following
	X'82' SF	Adapter number
	X'82' SF	Port number
	X'82' SF	Telephone number (45)
Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes
	X'09' SF	Remote Telephone Number

	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (44))
		Type = X'FA'(D-channel)
		Second resource below sender
		Name = (adjacent node)
		Type = X'F4'(CP)
	X'11' SF	Associated Resources
		Name = (D-channel (44))
		Type = X'FA'(D-channel)
		Name = (B-channel (44))
		Type = X'FB'(B-channel)

---

(43) The hexadecimal table index of the counter name.

(44) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

(45) The telephone number associated with the port.

A.4.4.25 ISDN B-Channel DLC Alert 5

Alert Condition:

An ISDN Logical Link Has Been Lost - The local link station inactivity timer or acknowledgement timer has expired, causing the remote station to be polled. The remote station does not respond to the poll.

Alert ID Number		X'9C8449A0'
Alert Type	X'01'	Permanent
Alert Description	X'3402'	B-Channel ISDN error
Probable Causes	X'210A'	ISDN communications/remote node
	X'1023'	Communications program in remote node
	X'2052'	Logical link control
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'210A'	ISDN communications/remote node
	X'1023'	Communications program in remote node
	X'2055'	Logical link control

	X'F019'	Inactivity timer expired
	X'F017'	Poll count exhausted
Actions	X'1204'	Attempt to reestablish the connection
	X'3301'	If problem persists then do the following
	X'2010'	(review link detail data)
	X'3109'	Contact personnel responsible for connection to ISDN network
Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes
	X'09' SF	Remote Telephone Number
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number
	X'8C' SV	Link station data
	X'01' SF	Current n(s)/n(r) counts
	X'02' SF	Outstanding frame count
	X'03' SF	Last Data Link control field received
	X'04' SF	Last Data Link control field sent
	X'05' SF	Sequence number modulus
	X'06' SF	Link station state
	X'07' SF	Data Link reply timer expiration count
	X'08' SF	Last received n(r) count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (46))
		Type = X'FA'(D-channel)
		Second resource below sender
		Name = (adjacent node)
		Type = X'F4'(CP)
	X'11' SF	Associated Resources
		Name = (D-channel (46))
		Type = X'FA'(D-channel)
		Name = (B-channel (46))
		Type = X'FB'(B-channel)

(46) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.26 ISDN B-Channel DLC Alert 6

Alert Condition:

An ISDN Logical Link Has Been Lost - This Alert indicates that the remote link station sent a Disconnect Mode (DM) response to the local link station.

Alert ID Number		X'3828C45A'
Alert Type	X'01'	Permanent
Alert Description	X'3402'	B-Channel ISDN error
Probable Causes	X'210A'	ISDN communications/remote node
	X'1023'	Communications program in remote node
	X'2052'	Logical link control
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'210A'	ISDN communications/remote node
	X'1023'	Communications program in remote node
	X'2055'	Logical link control
	X'F01A'	DM received
Actions	X'1204'	Attempt to reestablish the connection
	X'3301'	If problem persists then do the following
	X'2010'	(review link detail data)
	X'3109'	Contact personnel responsible for connection to ISDN network
Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes
	X'09' SF	Remote Telephone Number
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number
	X'8C' SV	Link station data
	X'01' SF	Current n(s)/n(r) counts
	X'02' SF	Outstanding frame count
	X'03' SF	Last Data Link control field received
	X'04' SF	Last Data Link control field sent
	X'05' SF	Sequence number modulus
	X'06' SF	Link station state
	X'07' SF	Data Link reply timer expiration count
	X'08' SF	Last received n(r) count
	X'05' SV	Hierarchy/Resource List

	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (47))
		Type = X'FA'(D-channel)
		Second resource below sender
		Name = (adjacent node)
		Type = X'F4'(CP)
	X'11' SF	Associated Resources
		Name = (D-channel (47))
		Type = X'FA'(D-channel)
		Name = (B-channel (47))
		Type = X'FB'(B-channel)

(47) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.27 ISDN B-Channel DLC Alert 7

Alert Condition:

An ISDN Logical Link Has Been Lost - This Alert indicates that the remote station sent a SABME command to the local link station which was already initialized.

Alert ID Number		X'0D245F3E'
Alert Type	X'01'	Permanent
Alert Description	X'3402'	B-Channel ISDN error
Probable Causes	X'210A'	ISDN communications/remote node
	X'1023'	Communications program in remote node
	X'2052'	Logical link control
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'210A'	ISDN communications/remote node
	X'1023'	Communications program in remote node
	X'2055'	Logical link control
	X'F016'	SABME received while in ABME
Actions	X'1204'	Attempt to reestablish the connection
	X'3301'	If problem persists then do the following
	X'2010'	(review link detail data)
	X'3109'	Contact personnel responsible for connection to ISDN network

Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes
	X'09' SF	Remote Telephone Number
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number
	X'8C' SV	Link station data
	X'01' SF	Current n(s)/n(r) counts
	X'02' SF	Outstanding frame count
	X'03' SF	Last Data Link control field received
	X'04' SF	Last Data Link control field sent
	X'05' SF	Sequence number modulus
	X'06' SF	Link station state
	X'07' SF	Data Link reply timer expiration count
	X'08' SF	Last received n(r) count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (48))
		Type = X'FA'(D-channel)
		Second resource below sender
		Name = (adjacent node)
		Type = X'F4'(CP)
	X'11' SF	Associated Resources
		Name = (D-channel (48))
		Type = X'FA'(D-channel)
		Name = (B-channel (48))
		Type = X'FB'(B-channel)

(48) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.28 ISDN B-Channel DLC Alert 8

Alert Condition:

An ISDN Logical Link Has Been Lost - This Alert indicates that the local link station sent an invalid or unsupported command or response to the remote link station. This resulted in the remote link station returning a Frame Reject response.

Alert ID Number		X'1179CD4A'
Alert Type	X'01'	Permanent

Alert Description	X'3402'	B-Channel ISDN error
Probable Causes	X'1022'	Communications program
	X'2052'	Logical link control
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1022'	Communications program
	X'2055'	Logical link control
	X'F010'	Frame reject received: invalid/unsupported command or response sent
Actions	X'1204'	Attempt to reestablish the connection
	X'3301'	If problem persists then do the following
	X'2010'	(review link detail data)
	X'3109'	Contact personnel responsible for connection to ISDN network
Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes
	X'09' SF	Remote Telephone Number
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number
	X'8C' SV	Link station data
	X'01' SF	Current n(s)/n(r) counts
	X'02' SF	Outstanding frame count
	X'03' SF	Last Data Link control field received
	X'04' SF	Last Data Link control field sent
	X'05' SF	Sequence number modulus
	X'06' SF	Link station state
	X'07' SF	Data Link reply timer expiration count
	X'08' SF	Last received n(r) count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (49))
		Type = X'FA'(D-channel)
		Second resource below sender
		Name = (adjacent node)
		Type = X'F4'(CP)
	X'11' SF	Associated Resources

		Name = (D-channel (49))
		Type = X'FA'(D-channel)
		Name = (B-channel (49))
		Type = X'FB'(B-channel)

(49) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.29 ISDN B-Channel DLC Alert 9

Alert Condition:

An ISDN Logical Link Has Been Lost - This Alert indicates that the local link station sent an I-field when not permitted to the remote link station. This resulted in the remote link station returning a Frame Reject response.

Alert ID Number		X'15B8D0FD'
Alert Type	X'01'	Permanent
Alert Description	X'3402'	B-Channel ISDN error
Probable Causes	X'1022'	Communications program
	X'2052'	Logical link control
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1022'	Communications program
	X'2055'	Logical link control
	X'F011'	Frame reject received: I field sent when not permitted
Actions	X'1204'	Attempt to reestablish the connection
	X'3301'	If problem persists then do the following
	X'2010'	(review link detail data)
	X'3109'	Contact personnel responsible for connection to ISDN network
Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes
	X'09' SF	Remote Telephone Number
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number

	X'8C' SV	Link station data
	X'01' SF	Current n(s)/n(r) counts
	X'02' SF	Outstanding frame count
	X'03' SF	Last Data Link control field received
	X'04' SF	Last Data Link control field sent
	X'05' SF	Sequence number modulus
	X'06' SF	Link station state
	X'07' SF	Data Link reply timer expiration count
	X'08' SF	Last received n(r) count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (50))
		Type = X'FA'(D-channel)
		Second resource below sender
		Name = (adjacent node)
		Type = X'F4'(CP)
	X'11' SF	Associated Resources
		Name = (D-channel (50))
		Type = X'FA'(D-channel)
		Name = (B-channel (50))
		Type = X'FB'(B-channel)

(50) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.30 ISDN B-Channel DLC Alert 10

Alert Condition:

An ISDN Logical Link Has Been Lost - This Alert indicates that the local link station sent a frame with an invalid N(r). This resulted in the remote link station returning a Frame Reject response.

Alert ID Number		X'18FBF624'
Alert Type	X'01'	Permanent
Alert Description	X'3402'	B-Channel ISDN error
Probable Causes	X'1022'	Communications program
	X'2052'	Logical link control
User Causes	(none)	
Install Causes	(none)	

Failure Causes	X'1022'	Communications program
	X'2055'	Logical link control
	X'F012'	Frame reject received: invalid N(r) sent
Actions	X'1204'	Attempt to reestablish the connection
	X'3301'	If problem persists then do the following
	X'2010'	(review link detail data)
	X'3109'	Contact personnel responsible for connection to ISDN network
Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes
	X'09' SF	Remote Telephone Number
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number
	X'8C' SV	Link station data
	X'01' SF	Current n(s)/n(r) counts
	X'02' SF	Outstanding frame count
	X'03' SF	Last Data Link control field received
	X'04' SF	Last Data Link control field sent
	X'05' SF	Sequence number modulus
	X'06' SF	Link station state
	X'07' SF	Data Link reply timer expiration count
	X'08' SF	Last received n(r) count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (51))
		Type = X'FA'(D-channel)
		Second resource below sender
		Name = (adjacent node)
		Type = X'F4'(CP)
	X'11' SF	Associated Resources
		Name = (D-channel (51))
		Type = X'FA'(D-channel)
		Name = (B-channel (51))
		Type = X'FB'(B-channel)

(51) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.31 ISDN B-Channel DLC Alert 11

Alert Condition:

An ISDN Logical Link Has Been Lost - This Alert indicates that the local link station sent a frame with an I-field that was too long. This resulted in the remote link station returning a Frame Reject response.

Alert ID Number		X'1C3AEB93'
Alert Type	X'01'	Permanent
Alert Description	X'3402'	B-Channel ISDN error
Probable Causes	X'1022'	Communications program
	X'2052'	Logical link control
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1022'	Communications program
	X'2055'	Logical link control
	X'F013'	Frame reject received: maximum I-field length exceeded
Actions	X'1204'	Attempt to reestablish the connection
	X'3301'	If problem persists then do the following
	X'2010'	(review link detail data)
	X'3109'	Contact personnel responsible for connection to ISDN network
Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes
	X'09' SF	Remote Telephone Number
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adpater Number
	X'0C' SF	Channel Number
	X'8C' SV	Link station data
	X'01' SF	Current n(s)/n(r) counts
	X'02' SF	Outstanding frame count
	X'03' SF	Last Data Link control field received
	X'04' SF	Last Data Link control field sent
	X'05' SF	Sequence number modulus
	X'06' SF	Link station state
	X'07' SF	Data Link reply timer expiration count
	X'08' SF	Last received n(r) count

	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (52))
		Type = X'FA'(D-channel)
		Second resource below sender
		Name = (adjacent node)
		Type = X'F4'(CP)
	X'11' SF	Associated Resources
		Name = (D-channel (52))
		Type = X'FA'(D-channel)
		Name = (B-channel (52))
		Type = X'FB'(B-channel)

---

(52) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.32 ISDN B-Channel DLC Alert 12

Alert Condition:

An ISDN Logical Link Has Been Lost - This Alert indicates that the remote link station sent an invalid or unsupported command or response to the local link station. This resulted in the local link station returning a Frame Reject response.

Alert ID Number		X'C3907F1C'
Alert Type	X'01'	Permanent
Alert Description	X'3402'	B-Channel ISDN error
Probable Causes	X'210A'	ISDN communications/remote node
	X'1023'	Communications program in remote node
	X'2052'	Logical link control
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'210A'	ISDN communications/remote node
	X'1023'	Communications program in remote node
	X'2055'	Logical link control
	X'F020'	Invalid/unsupported command or response received
Actions	X'1204'	Attempt to reestablish the connection
	X'3301'	If problem persists then do the following
	X'2010'	(review link detail data)

	X'3109'	Contact personnel responsible for connection to ISDN network
Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes
	X'09' SF	Remote Telephone Number
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number
	X'8C' SV	Link station data
	X'01' SF	Current n(s)/n(r) counts
	X'02' SF	Outstanding frame count
	X'03' SF	Last Data Link control field received
	X'04' SF	Last Data Link control field sent
	X'05' SF	Sequence number modulus
	X'06' SF	Link station state
	X'07' SF	Data Link reply timer expiration count
	X'08' SF	Last received n(r) count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (53))
		Type = X'FA'(D-channel)
		Second resource below sender
		Name = (adjacent node)
		Type = X'F4'(CP)
	X'11' SF	Associated Resources
		Name = (D-channel (53))
		Type = X'FA'(D-channel)
		Name = (B-channel (53))
		Type = X'FB'(B-channel)

---

(53) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.33 ISDN B-Channel DLC Alert 13

Alert Condition:

An ISDN Logical Link Has Been Lost - This Alert indicates that the remote link station sent an I-field when not permitted to the local link station. This resulted in the local link station returning a Frame Reject response.

Alert ID Number		X'C75162AB'

Alert Type	X'01'	Permanent
Alert Description	X'3402'	B-Channel ISDN error
Probable Causes	X'210A'	ISDN communications/remote node
	X'1023'	Communications program in remote node
	X'2052'	Logical link control
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'210A'	ISDN communications/remote node
	X'1023'	Communications program in remote node
	X'2055'	Logical link control
	X'F021'	I-field received when not permitted
Actions	X'1204'	Attempt to reestablish the connection
	X'3301'	If problem persists then do the following
	X'2010'	(review link detail data)
	X'3109'	Contact personnel responsible for connection to ISDN network
Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes
	X'09' SF	Remote Telephone Number
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number
	X'8C' SV	Link station data
	X'01' SF	Current n(s)/n(r) counts
	X'02' SF	Outstanding frame count
	X'03' SF	Last Data Link control field received
	X'04' SF	Last Data Link control field sent
	X'05' SF	Sequence number modulus
	X'06' SF	Link station state
	X'07' SF	Data Link reply timer expiration count
	X'08' SF	Last received n(r) count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (54))
		Type = X'FA'(D-channel)
		Second resource below sender
		Name = (adjacent node)

		Type = X'F4'(CP)
	X'11' SF	Associated Resources
		Name = (D-channel (54))
		Type = X'FA'(D-channel)
		Name = (B-channel (54))
		Type = X'FB'(B-channel)

---

(54) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.34 ISDN B-Channel DLC Alert 14

Alert Condition:

An ISDN Logical Link Has Been Lost - This Alert indicates that the remote link station sent a frame with an invalid N(r). This resulted in the local link station returning a Frame Reject response.

Alert ID Number		X'CA124472'
Alert Type	X'01'	Permanent
Alert Description	X'3402'	B-Channel ISDN error
Probable Causes	X'210A'	ISDN communications/remote node
	X'1023'	Communications program in remote node
	X'2052'	Logical link control
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'210A'	ISDN communications/remote node
	X'1023'	Communications program in remote node
	X'2055'	Logical link control
	X'F022'	Invalid N(r) received
Actions	X'1204'	Attempt to reestablish the connection
	X'3301'	If problem persists then do the following
	X'2010'	(review link detail data)
	X'3109'	Contact personnel responsible for connection to ISDN network
Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes

	X'09' SF	Remote Telephone Number
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number
	X'8C' SV	Link station data
	X'01' SF	Current n(s)/n(r) counts
	X'02' SF	Outstanding frame count
	X'03' SF	Last Data Link control field received
	X'04' SF	Last Data Link control field sent
	X'05' SF	Sequence number modulus
	X'06' SF	Link station state
	X'07' SF	Data Link reply timer expiration count
	X'08' SF	Last received n(r) count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (55))
		Type = X'FA'(D-channel)
		Second resource below sender
		Name = (adjacent node)
		Type = X'F4'(CP)
	X'11' SF	Associated Resources
		Name = (D-channel (55))
		Type = X'FA'(D-channel)
		Name = (B-channel (55))
		Type = X'FB'(B-channel)

(55) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.35 ISDN B-Channel DLC Alert 15

Alert Condition:

An ISDN Logical Link Has Been Lost - This Alert indicates that the remote link station sent a frame with an I-field that was too long. This resulted in the local link station returning a Frame Reject response.

Alert ID Number		X'CED359C5'
Alert Type	X'01'	Permanent
Alert Description	X'3402'	B-Channel ISDN error
Probable Causes	X'210A'	ISDN communications/remote node
	X'1023'	Communications program in remote node
	X'2052'	Logical link control

User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'210A'	ISDN communications/remote node
	X'1023'	Communications program in remote node
	X'2055'	Logical link control
	X'F023'	Received I-field exceeded maximum length
Actions	X'1204'	Attempt to reestablish the connection
	X'3301'	If problem persists then do the following
	X'2010'	(review link detail data)
	X'3109'	Contact personnel responsible for connection to ISDN network
Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes
	X'09' SF	Remote Telephone Number
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number
	X'8C' SV	Link station data
	X'01' SF	Current n(s)/n(r) counts
	X'02' SF	Outstanding frame count
	X'03' SF	Last Data Link control field received
	X'04' SF	Last Data Link control field sent
	X'05' SF	Sequence number modulus
	X'06' SF	Link station state
	X'07' SF	Data Link reply timer expiration count
	X'08' SF	Last received n(r) count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (56))
		Type = X'FA'(D-channel)
		Second resource below sender
		Name = (adjacent node)
		Type = X'F4'(CP)
	X'11' SF	Associated Resources
		Name = (D-channel (56))
		Type = X'FA'(D-channel)
		Name = (B-channel (56))
		Type = X'FB'(B-channel)

(56) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.36 ISDN B-Channel DLC Alert 16

Alert Condition:

Supporting D-channel Failure - This Alert is generated when a D-channel has failed and as a result the associated B-channels have failed. The logical configuration of the B-channels in the alert sender have taken place but the link connections may or not have been established. This alert applies to both Basic Rate and Primary Rate ISDN.

Alert ID Number		X'0FBD32FA'
Alert Type	X'01'	Permanent
Alert Description	X'3402'	B-Channel ISDN error
Probable Causes	X'352F'	ISDN TE-NT connection
	X'210A'	ISDN communications/remote node
	X'200A'	ISDN network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'352F'	ISDN TE-NT connection
	X'2057'	D-Channel ISDN error
	X'210A'	ISDN communications/remote node
Actions	X'1305'	Reactivate B-channel resource
	X'3301'	If problem persists then do the following
	X'3106'	Contact ISDN network information service
	X'32A0'	Report the following
	X'82' SF	Telephone number (58)
	X'3109'	Contact personnel responsible for connection to ISDN network
	X'32C0'	Report the following
	X'82' SF	Adapter number
	X'82' SF	Port number
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (57))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (57))

		Type = X'FA'(D-channel)
		Name = (B-channel (57))
		Type = X'FB'(B-channel)

(57) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

(58) The telephone number associated with the port.

A.4.4.37 ISDN B-Channel DLC Alert 17

Alert Condition:

An ISDN logical link has been lost - This Alert is generated when the ISDN Network unexpectedly disconnects the ISDN call on the D-channel thereby dropping the B-channel and the established logical link connection(s). This situation occurs anytime the D-channel signaling procedures receive a disconnection message before the B-channel resources have been deactivated.

Alert ID Number		X'1C130D13'
Alert Type	X'01'	Permanent
Alert Description	X'3407'	ISDN call terminated unexpectedly
Probable Causes	X'200A'	ISDN network
	X'210A'	ISDN communications/remote node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'200A'	ISDN network
	X'210A'	ISDN communications/remote node
	X'1023'	ISDN communications program in remote node
Actions	X'1204'	Attempt to reestablish the connection
	X'3301'	If problem persists then do the following
	X'3106'	Contact ISDN network information service
	X'32A0'	Report the following
	X'82' SF	Telephone number (60)
	X'3109'	Contact personnel responsible for connection to ISDN network
	X'32D0'	Report the following
	X'82' SF	Adapter number
	X'82' SF	Port number
	X'82' SF	Call Progress Signal
Additional SVs	X'05' SV	Hierarchy/Resource List

	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (59))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (59))
		Type = X'FA'(D-channel)
		Name = (B-channel (59))
		Type = X'FB'(B-channel)

(59) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

(60) The telephone number associated with the port.

A.4.4.38 ISDN Layer 2 D-Channel LAPD Alerts

The D channel LAPD Alerts are for communication problems between the end-user and the ISDN Network Service provider and not for end-to-end problem reporting.

A.4.4.39 ISDN D-Channel LAPD Alert 1

Alert Condition:

Excessive D-Channel Receive Errors - This Alert is generated when the CRC Errors Received (X'0006') counter has reached its threshold. This indicates that there are excessive link errors between two DLC stations across the D channel.

Alert ID Number		X'C506D595'
Alert Type	X'03'	Performance
Alert Description	X'4012'	Threshold has been reached
Probable Causes	X'3529'	ISDN TE-NT connection
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2057'	D-Channel ISDN error
	X'3529'	ISDN TE-NT connection
	X'200E'	Local DCE loop
	X'40A0'	(sf 82) threshold reached
	X'82' SF	Counter (61)
Actions	X'3301'	If problem persists then do the following:

	X'2002'	(review link detail data)
	X'2010'	(review most recent traffic statistics)
	X'3106'	Contact ISDN network information service
Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (62))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (62))
		Type = X'FA'(D-channel)

---

(61) The hexadecimal table index of the counter name.

(62) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.40 ISDN D-Channel LAPD Alert 2

Alert Condition:

Excessive D-Channel Link Station Errors - This Alert is generated when either of the following station error counters have reached their thresholds: the PDUs Retransmitted (X'0017'), or the Received Sequence Errors (X'0018'). This indicates that there are an excessive link errors between two DLC stations across the D channel.

Alert ID Number		X'1BBEA353'
Alert Type	X'03'	Performance
Alert Description	X'4012'	Threshold has been reached
Probable Causes	X'210A'	ISDN Communications/remote node
	X'3529'	ISDN TE-NT connection
User Causes	(none)	
Install Causes	(none)	

Failure Causes	X'210A'	ISDN Communications/remote node
	X'1023'	Communications program in remote node
	X'2057'	D-Channel ISDN error
	X'3529'	ISDN TE-NT connection
	X'200E'	Local DCE loop
	X'40A0'	(sf 82) threshold reached
	X'82' SF	Counter (63)
Actions	X'3301'	If problem persists then do the following:
	X'2010'	(review link detail data)
	X'2002'	(review most recent traffic statistics)
	X'3106'	Contact ISDN network information service
Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (64))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (64))
		Type = X'FA'(D-channel)

---

(63) The hexadecimal table index of the counter name.

(64) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.41 ISDN D-Channel LAPD Alert 3

Alert Condition:

Excessive D-Channel Invalid Frame Errors - This Alert is generated when the count of a type of invalid frame on the D-Channel has reached the assigned threshold. The invalid frame category includes the following: Short Frames Received (X'000B'), Misaddressed Frames Received (X'0012'), Unbounded Frames Received (X'0013'), Non-Integral Frames Received (X'0014'), Aborted Frames Received (X'0010'), Aborted Frames Transmitted (X'0011'), and. the summary counter Badly Formed Frames (X'0044'). This indicates that either there are excessive link errors between two DLC stations across the D-Channel or the remote node is failing.

Alert ID Number		X'AD6607B6'

Alert Type	X'03'	Performance
Alert Description	X'4012'	Threshold has been reached
Probable Causes	X'210A'	ISDN Communications/remote node
	X'3529'	ISDN TE-NT connection
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3311'	Remote ISDN Adapter
	X'3529'	ISDN TE-NT connection
	X'40A0'	(sf 82) threshold reached
	X'82' SF	Counter (65)
Actions	X'3301'	If problem persists then do the following:
	X'2010'	(review link detail data)
	X'2002'	(review most recent traffic statistics)
	X'3106'	Contact ISDN network information service
Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (66))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (66))
		Type = X'FA'(D-channel)

---

(65) The hexadecimal table index of the counter name.

(66) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.42 ISDN D-Channel LAPD Alert 4

Alert Condition:

ISDN Adapter Buffer Overrun/Underrun - Either the Buffer Overrun (X'000C') or the Buffer Underrun (X'000D') counter has exceeded its threshold.

Alert ID Number		X'AD564CF0'
Alert Type	X'03'	Performance
Alert Description	X'4012'	Threshold has been reached
Probable Causes	X'3310'	Local ISDN adapter
	X'3222'	ISDN adapter interface
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'3310'	Local ISDN adapter
	X'40A0'	(sf 82) threshold reached
	X'82' SF	Counter (67)
Actions	X'3301'	If problem persists, then do the following:
	X'2002'	(review most recent traffic statistics)
	X'3109'	Contact personnel responsible for connection to ISDN network
	X'32D0'	Report the following
	X'82' SF	Adapter number
	X'82' SF	Port number
	X'82' SF	Telephone number (69)
Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (68))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (68))
		Type = X'FA'(D-channel)

(67) The hexadecimal table index of the counter name.

(68) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

(69) The telephone number associated with the port.

A.4.4.43 ISDN D-Channel LAPD Alert 5

Alert Condition:

An ISDN Logical Link Has Been Lost - The local link station inactivity timer or acknowledgement timer has expired, causing the remote station to be polled. The remote station does not respond to the poll.

Alert ID Number		X'965033D0'
Alert Type	X'01'	Permanent
Alert Description	X'3401'	D-Channel ISDN error
Probable Causes	X'210A'	ISDN communications/remote node
	X'1023'	Communications program in remote node
	X'2052'	Logical link control
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'210A'	ISDN communications/remote node
	X'1023'	Communications program in remote node
	X'2055'	Logical link control
	X'F019'	Inactivity timer expired
	X'F017'	Poll count exhausted
Actions	X'1204'	Attempt to reestablish the connection
	X'3301'	If problem persists then do the following
	X'2010'	(review link detail data)
	X'3106'	Contact ISDN network information service
Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number
	X'8C' SV	Link station data
	X'01' SF	Current n(s)/n(r) counts
	X'02' SF	Outstanding frame count
	X'03' SF	Last Data Link control field received

	X'04' SF	Last Data Link control field sent
	X'05' SF	Sequence number modulus
	X'06' SF	Link station state
	X'07' SF	Data Link reply timer expiration count
	X'08' SF	Last received n(r) count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (70))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (70))
		Type = X'FA'(D-channel)

(70) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.44 ISDN D-Channel LAPD Alert 6

Alert Condition:

An ISDN Logical Link Has Been Lost - This Alert indicates that the remote link station sent a Disconnect Mode (DM) response to the local link station.

Alert ID Number		X'72238E39'
Alert Type	X'01'	Permanent
Alert Description	X'3401'	D-Channel ISDN error
Probable Causes	X'210A'	ISDN communications/remote node
	X'1023'	Communications program in remote node
	X'2052'	Logical link control
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'210A'	ISDN communications/remote node
	X'1023'	Communications program in remote node
	X'2055'	Logical link control
	X'F01A'	DM received
Actions	X'1204'	Attempt to reestablish the connection
	X'3301'	If problem persists then do the following
	X'2010'	(review link detail data)

	X'3106'	Contact ISDN network information service
Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number
	X'8C' SV	Link station data
	X'01' SF	Current n(s)/n(r) counts
	X'02' SF	Outstanding frame count
	X'03' SF	Last Data Link control field received
	X'04' SF	Last Data Link control field sent
	X'05' SF	Sequence number modulus
	X'06' SF	Link station state
	X'07' SF	Data Link reply timer expiration count
	X'08' SF	Last received n(r) count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (71))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (71))
		Type = X'FA'(D-channel)

(71) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.45 ISDN D-Channel LAPD Alert 7

Alert Condition:

An ISDN Logical Link Has Been Lost - This Alert indicates that the remote station sent a SABME command to the local link station which was already initialized.

Alert ID Number		X'472F155D'
Alert Type	X'01'	Permanent
Alert Description	X'3401'	D-Channel ISDN error
Probable Causes	X'210A'	ISDN communications/remote node

	X'1023'	Communications program in remote node
	X'2052'	Logical link control
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'210A'	ISDN communications/remote node
	X'1023'	Communications program in remote node
	X'2055'	Logical link control
	X'F016'	SABME received while in ABME
Actions	X'1204'	Attempt to reestablish the connection
	X'3301'	If problem persists then do the following
	X'2010'	(review link detail data)
	X'3106'	Contact ISDN network information service
Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number
	X'8C' SV	Link station data
	X'01' SF	Current n(s)/n(r) counts
	X'02' SF	Outstanding frame count
	X'03' SF	Last Data Link control field received
	X'04' SF	Last Data Link control field sent
	X'05' SF	Sequence number modulus
	X'06' SF	Link station state
	X'07' SF	Data Link reply timer expiration count
	X'08' SF	Last received n(r) count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (72))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (72))
		Type = X'FA'(D-channel)

(72) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.46 ISDN D-Channel LAPD Alert 8

Alert Condition:

An ISDN Logical Link Has Been Lost - This Alert indicates that the local link station sent an invalid or unsupported command or response to the remote link station. This resulted in the remote link station returning a Frame Reject response.

Alert ID Number		X'290C9EAD'
Alert Type	X'01'	Permanent
Alert Description	X'3401'	D-Channel ISDN error
Probable Causes	X'1022'	Communications program
	X'2052'	Logical link control
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1022'	Communications program
	X'2055'	Logical link control
	X'F010'	Frame reject received: invalid/unsupported command or response sent
Actions	X'1204'	Attempt to reestablish the connection
	X'3301'	If problem persists then do the following
	X'2010'	(review link detail data)
	X'3109'	Contact personnel responsible for connection to ISDN network
Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number
	X'8C' SV	Link station data
	X'01' SF	Current n(s)/n(r) counts
	X'02' SF	Outstanding frame count
	X'03' SF	Last Data Link control field received
	X'04' SF	Last Data Link control field sent
	X'05' SF	Sequence number modulus
	X'06' SF	Link station state
	X'07' SF	Data Link reply timer expiration count
	X'08' SF	Last received n(r) count

	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (73))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (73))
		Type = X'FA'(D-channel)

(73) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.47 ISDN D-Channel LAPD Alert 9

Alert Condition:

An ISDN Logical Link Has Been Lost - This Alert indicates that the local link station sent an I-field when not permitted to the remote link station. This resulted in the remote link station returning a Frame Reject response.

Alert ID Number		X'2DCD831A'
Alert Type	X'01'	Permanent
Alert Description	X'3401'	D-Channel ISDN error
Probable Causes	X'1022'	Communications program
	X'2052'	Logical link control
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1022'	Communications program
	X'2055'	Logical link control
	X'F011'	Frame reject received: I field sent when not permitted
Actions	X'1204'	Attempt to reestablish the connection
	X'3301'	If problem persists then do the following
	X'2010'	(review link detail data)
	X'3109'	Contact personnel responsible for connection to ISDN network
Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes

	X'07' SF	LCS Link Attributes
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number
	X'8C' SV	Link station data
	X'01' SF	Current n(s)/n(r) counts
	X'02' SF	Outstanding frame count
	X'03' SF	Last Data Link control field received
	X'04' SF	Last Data Link control field sent
	X'05' SF	Sequence number modulus
	X'06' SF	Link station state
	X'07' SF	Data Link reply timer expiration count
	X'08' SF	Last received n(r) count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (74))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (74))
		Type = X'FA'(D-channel)

(74) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.48 ISDN D-Channel LAPD Alert 10

Alert Condition:

An ISDN Logical Link Has Been Lost - This Alert indicates that the local link station sent a frame with an invalid N(r). This resulted in the remote link station returning a Frame Reject response.

Alert ID Number		X'208EA5C3'
Alert Type	X'01'	Permanent
Alert Description	X'3401'	D-Channel ISDN error
Probable Causes	X'1022'	Communications program
	X'2052'	Logical link control
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1022'	Communications program

	X'2055'	Logical link control
	X'F012'	Frame reject received: invalid N(r) sent
Actions	X'1204'	Attempt to reestablish the connection
	X'3301'	If problem persists then do the following
	X'2010'	(review link detail data)
	X'3109'	Contact personnel responsible for connection to ISDN network
Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number
	X'8C' SV	Link station data
	X'01' SF	Current n(s)/n(r) counts
	X'02' SF	Outstanding frame count
	X'03' SF	Last Data Link control field received
	X'04' SF	Last Data Link control field sent
	X'05' SF	Sequence number modulus
	X'06' SF	Link station state
	X'07' SF	Data Link reply timer expiration count
	X'08' SF	Last received n(r) count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (75))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (75))
		Type = X'FA'(D-channel)

(75) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.49 ISDN D-Channel LAPD Alert 11

Alert Condition:

An ISDN Logical Link Has Been Lost - This Alert indicates that the local link station sent a frame with an I-field that was too long. This resulted in the remote link station returning a Frame Reject response.

Alert ID Number		X'244FB874'

Alert Type	X'01'	Permanent
Alert Description	X'3401'	D-Channel ISDN error
Probable Causes	X'1022'	Communications program
	X'2052'	Logical link control
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1022'	Communications program
	X'2055'	Logical link control
	X'F013'	Frame reject received: maximum I-field length exceeded
Actions	X'1204'	Attempt to reestablish the connection
	X'3301'	If problem persists then do the following
	X'2010'	(review link detail data)
	X'3109'	Contact personnel responsible for connection to ISDN network
Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number
	X'8C' SV	Link station data
	X'01' SF	Current n(s)/n(r) counts
	X'02' SF	Outstanding frame count
	X'03' SF	Last Data Link control field received
	X'04' SF	Last Data Link control field sent
	X'05' SF	Sequence number modulus
	X'06' SF	Link station state
	X'07' SF	Data Link reply timer expiration count
	X'08' SF	Last received n(r) count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (76))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (76))
		Type = X'FA'(D-channel)

(76) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.50 ISDN D-Channel LAPD Alert 12

Alert Condition:

An ISDN Logical Link Has Been Lost - This Alert indicates that the remote link station sent an invalid or unsupported command or response to the local link station. This resulted in the local link station returning a Frame Reject response.

Alert ID Number		X'899B357F'
Alert Type	X'01'	Permanent
Alert Description	X'3401'	D-Channel ISDN error
Probable Causes	X'210A'	ISDN communications/remote node
	X'1023'	Communications program in remote node
	X'2052'	Logical link control
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'210A'	ISDN communications/remote node
	X'1023'	Communications program in remote node
	X'2055'	Logical link control
	X'F020'	Invalid/unsupported command or response received
Actions	X'1204'	Attempt to reestablish the connection
	X'3301'	If problem persists then do the following
	X'2010'	(review link detail data)
	X'3106'	Contact ISDN network information service
Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number
	X'8C' SV	Link station data
	X'01' SF	Current n(s)/n(r) counts
	X'02' SF	Outstanding frame count

	X'03' SF	Last Data Link control field received
	X'04' SF	Last Data Link control field sent
	X'05' SF	Sequence number modulus
	X'06' SF	Link station state
	X'07' SF	Data Link reply timer expiration count
	X'08' SF	Last received n(r) count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (77))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (77))
		Type = X'FA'(D-channel)

(77) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.51 ISDN D-Channel LAPD Alert 13

Alert Condition:

An ISDN Logical Link Has Been Lost - This Alert indicates that the remote link station sent an I-field when not permitted to the local link station. This resulted in the local link station returning a Frame Reject response.

Alert ID Number		X'8D5A28C8'
Alert Type	X'01'	Permanent
Alert Description	X'3401'	D-Channel ISDN error
Probable Causes	X'210A'	ISDN communications/remote node
	X'1023'	Communications program in remote node
	X'2052'	Logical link control
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'210A'	ISDN communications/remote node
	X'1023'	Communications program in remote node
	X'2055'	Logical link control
	X'F021'	I-field received when not permitted
Actions	X'1204'	Attempt to reestablish the connection
	X'3301'	If problem persists then do the following

	X'2010'	(review link detail data)
	X'3106'	Contact ISDN network information service
Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number
	X'8C' SV	Link station data
	X'01' SF	Current n(s)/n(r) counts
	X'02' SF	Outstanding frame count
	X'03' SF	Last Data Link control field received
	X'04' SF	Last Data Link control field sent
	X'05' SF	Sequence number modulus
	X'06' SF	Link station state
	X'07' SF	Data Link reply timer expiration count
	X'08' SF	Last received n(r) count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (78))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (78))
		Type = X'FA'(D-channel)

---

(78) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.52 ISDN D-Channel LAPD Alert 14

Alert Condition:

An ISDN Logical Link Has Been Lost - This Alert indicates that the remote link station sent a frame with an invalid N(r). This resulted in the local link station returning a Frame Reject response.

Alert ID Number		X'80190E11'
Alert Type	X'01'	Permanent
Alert Description	X'3401'	D-Channel ISDN error

Probable Causes	X'210A'	ISDN communications/remote node
	X'1023'	Communications program in remote node
	X'2052'	Logical link control
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'210A'	ISDN communications/remote node
	X'1023'	Communications program in remote node
	X'2055'	Logical link control
	X'F022'	Invalid N(r) received
Actions	X'1204'	Attempt to reestablish the connection
	X'3301'	If problem persists then do the following
	X'2010'	(review link detail data)
	X'3106'	Contact ISDN network information service
Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number
	X'8C' SV	Link station data
	X'01' SF	Current n(s)/n(r) counts
	X'02' SF	Outstanding frame count
	X'03' SF	Last Data Link control field received
	X'04' SF	Last Data Link control field sent
	X'05' SF	Sequence number modulus
	X'06' SF	Link station state
	X'07' SF	Data Link reply timer expiration count
	X'08' SF	Last received n(r) count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (79))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (79))
		Type = X'FA'(D-channel)

(79) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.53 ISDN D-Channel LAPD Alert 15

Alert Condition:

An ISDN Logical Link Has Been Lost - This Alert indicates that the remote link station sent a frame with an I-field that was too long. This resulted in the local link station returning a Frame Reject response.

Alert ID Number		X'84D813A6'
Alert Type	X'01'	Permanent
Alert Description	X'3401'	D-Channel ISDN error
Probable Causes	X'210A'	ISDN communications/remote node
	X'1023'	Communications program in remote node
	X'2052'	Logical link control
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'210A'	ISDN communications/remote node
	X'1023'	Communications program in remote node
	X'2055'	Logical link control
	X'F023'	Received I-field exceeded maximum length
Actions	X'1204'	Attempt to reestablish the connection
	X'3301'	If problem persists then do the following
	X'2010'	(review link detail data)
	X'3106'	Contact ISDN network information service
Additional SVs	X'52' SV	LCS Configuration Data
	X'01' SF	Port Address
	X'04' SF	Local Device Address
	X'06' SF	LCS Link Station Attributes
	X'07' SF	LCS Link Attributes
	X'0A' SF	Local Telephone Number
	X'0B' SF	Adapter Number
	X'0C' SF	Channel Number
	X'8C' SV	Link station data
	X'01' SF	Current n(s)/n(r) counts
	X'02' SF	Outstanding frame count
	X'03' SF	Last Data Link control field received
	X'04' SF	Last Data Link control field sent
	X'05' SF	Sequence number modulus
	X'06' SF	Link station state

	X'07' SF	Data Link reply timer expiration count
	X'08' SF	Last received n(r) count
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (80))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (80))
		Type = X'FA'(D-channel)

---

(80) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.54 ISDN D-Channel LAPD Alert 16

Alert Condition:

TEI Assignment Failure - This Alert is generated when the attempt to request a Terminal Equipment Identifier from the network has failed.

Alert ID Number		X'BE1F4058'
Alert Type	X'01'	Permanent
Alert Description	X'3401'	D-Channel ISDN error
Probable Causes	X'230B'	Link setup failure
	X'3222'	ISDN adapter interface
	X'200A'	ISDN network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'352F'	ISDN TE-NT connection
	X'200A'	ISDN communications error
	X'2055'	Logical link control
	X'230B'	Terminal Equipment Identifier (TEI) assignment
Actions	X'1204'	Attempt to reestablish the connection
	X'3301'	If problem persists, then do the following:
	X'3109'	Contact personnel responsible for connection to ISDN network
	X'32D0'	Report the following
	X'82' SF	Adapter number
	X'82' SF	Port number

	X'82' SF	Telephone number (82)
	X'3106'	Contact ISDN network information service
	X'32A0'	Report the following
	X'82' SF	Telephone number (82)
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (81))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (81))
		Type = X'FA'(D-channel)

---

(81) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

(82) The telephone number associated with the port.

A.4.4.55 ISDN D-Channel LAP-D Alert 17

Alert Condition:

Primary D-channel Failure, Backup D-channel Activated - This Alert is generated when a D-channel has failed but a Backup D-channel has been activated to carry the signaling load. The Backup D-channel number and location is determined by agreement with ISDN network provider at service subscription time. No B-channels should be affected by this procedure. Returning the primary D-channel to operation requires coordination with the network provider. This alert applies to Primary Rate ISDN.

Alert ID Number		X'8CC035A5' (83)
Alert Type	X'01'	Permanent
Alert Description	X'3402'	D-Channel ISDN error
Probable Causes	X'352F'	ISDN TE-NT connection
	X'200A'	ISDN network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'352F'	ISDN TE-NT connection
	X'2057'	D-Channel ISDN error
	X'200A'	ISDN network
Actions	X'0163'	Verify backup D-channel operation
	X'3106'	Contact ISDN network information service

	X'32D0'	Report the following
	X'82' SF	Telephone number (86)
	X'82' SF	D-channel number (84)
	X'82' SF	Backup D-channel number (84)
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (85))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (85))
		Type = X'FA'(D-channel)

| (83) This Alert was previously documented with the incorrect | Alert ID value of X'B9F1A620'. Alert receivers should be | able to accept Alerts containing either the current, correct | value or the old, incorrect value.

(84) The D-channel or Backup D-channel number assigned at subscription time.

(85) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

(86) The telephone number associated with the port.

A.4.4.56 ISDN Layer 3 Q.931 Call Control Alerts

A.4.4.57 Q.931 Alert 1

Alert Condition:

This Alert is generated as a result of one of the following conditions:

- A call has been rejected due to cause code number 1, unassigned number. This cause indicates that the destination requested by the calling user cannot be reached because the number is not presently assigned.

- A call has been rejected due to cause code number 28, invalid number format(address incomplete). This cause indicates that the called user can not be reached because the called party number is not a valid format or is not complete.

Alert ID Number		X'50350202'
Alert Type	X'01'	Permanent
Alert Description	X'3406'	ISDN Call Control error
Probable Causes	X'2304'	Incorrect number called
User Causes	X'2304'	Incorrect telephone number specified

Actions	X'3122'	Contact called DTE's operator
	X'0103'	Verify telephone number
	X'1204'	Attempt to reestablish the connection
Install Causes	(none)	
Failure Causes	X'200A'	ISDN communications error
	X'2307'	Called number outside numbering plan or unknown by network
Actions	X'3302'	If problem continues to occur repeatedly then do the following
	X'3106'	Contact ISDN network information service
	X'32D0'	Report the following
	X'82' SF	Calling telephone number
	X'82' SF	Telephone number called
	X'82' SF	Call progress signal
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (87))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (87))
		Type = X'FA'(D-channel)

---

(87) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.58 Q.931 Alert 2

Alert Condition:

This Alert is generated as a result of one of the following conditions:

- A call has been rejected due to cause code number 2, No route to specific transit network. This cause indicates that the equipment sending the cause has received a request to route the call through a particular transit network which it does not recognize, either the transit network does not exist or while it does, it does not serve the equipment sending this cause.

- A call has been rejected due to cause code number 3, No route to destination. This cause indicates that the called user can not be reached because the network through which the call has been routed does not serve the destination desired.

- A call has been rejected due to cause code number 91, invalid transit network selection. This cause indicates that a transit network identification was received which is of an incorrect format.

Alert ID Number		X'B06E9FB3'

Alert Type	X'01'	Permanent
Alert Description	X'3406'	ISDN Call Control error
Probable Causes	X'2307'	Invalid request
	X'2311'	Service unavailable or not supported
User Causes	X'2307'	Incorrect parameter specified
	X'F001'	Invalid transit network routing selected
Actions	X'3122'	Contact called DTE's operator
	X'1512'	Change call setup parameters
	X'1204'	Attempt to reestablish the connection
Install Causes	(none)	
Failure Causes	X'2311'	Service not available or not supported
Actions	X'3302'	If problem continues to occur repeatedly then do the following
	X'3106'	Contact ISDN network information service
	X'32D0'	Report the following
	X'82' SF	Calling telephone number
	X'82' SF	Telephone number called
	X'82' SF	Call progress signal
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (88))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (88))
		Type = X'FA'(D-channel)

(88) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.59 Q.931 Alert 3

Alert Condition:

This Alert is generated as a result of one of the following conditions:

- A call has been rejected due to cause code number 6, channel unacceptable. This cause indicates that the channel most recently identified is not acceptable to the sender for use in this call.

- A call has been rejected due to resource unavailable, the cause code

received is number 44, requested circuit/channel not available. This cause indicates that the circuit or channel requested by the user can not be provided by the other side of the interface.

° A call has been rejected due to cause code number 66, channel type not implemented. This cause indicates that the equipment generating this cause does not support the channel type requested.

° A call has been rejected due to cause code received is number 82, identified channel does not exist. This cause indicates that the equipment sending this cause has received a request to use a channel not activated on the interface for a call.

Alert ID Number		X'61972780'
Alert Type	X'01'	Permanent
Alert Description	X'3406'	ISDN Call Control error
Probable Causes	X'2307'	Invalid Request
	X'2311'	Service unavailable or not supported
User Causes	X'2307'	Incorrect parameter specified
	X'F002'	Channel request invalid
Actions	X'1512'	Change call setup parameters
	X'1204'	Attempt to reestablish the connection
Install Causes	(none)	
Failure Causes	X'2311'	Service unavailable or not supported
Actions	X'3302'	If problem continues to occur repeatedly then do the following
	X'3106'	Contact ISDN network information service
	X'32D0'	Report the following
	X'82' SF	Calling telephone number
	X'82' SF	Telephone number called
	X'82' SF	Call progress signal
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (89))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (89))
		Type = X'FA'(D-channel)

(89) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.60 Q.931 Alert 4

Alert Condition:

This Alert is generated as a result of a call that has been rejected due to cause code number 17, user busy. This cause indicates that the called user does not have the ability to accept another call.

Alert ID Number		X'E4D34AA2'
Alert Type	X'01'	Permanent
Alert Description	X'3406'	ISDN Call Control error
Probable Causes	X'2301'	Called number busy
User Causes	X'2304'	Incorrect telephone number specified
Actions	X'3122'	Contact called DTE's operator
	X'0103'	Verify telephone number
	X'1204'	Attempt to reestablish the connection
Install Causes	(none)	
Failure Causes	X'2201'	Called DTE
	X'200A'	ISDN communications error
Actions	X'1206'	Wait and retry
	X'3302'	If problem continues to occur repeatedly then do the following
	X'3106'	Contact ISDN network information service
	X'32D0'	Report the following
	X'82' SF	Calling telephone number
	X'82' SF	Telephone number called
	X'82' SF	Call progress signal
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (90))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (90))
		Type = X'FA'(D-channel)

(90) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.61 Q.931 Alert 5

Alert Condition:

This Alert is generated as a result of one of the following conditions:

° A call has been rejected due to cause code number 18, no user responding. This cause indicates that the called user does not respond to a call establishment message with either an alerting or connection indication within the prescribed period of time.

° A call has been rejected due to cause code number 19, user alerting, no answer. This cause indicates that the called user has provided an alerting indication but has not provided a connect indication within the prescribed period of time.

Alert ID Number		X'CC48E2E7'
Alert Type	X'01'	Permanent
Alert Description	X'3406'	ISDN Call Control error
Probable Causes	X'2300'	Connection not established
User Causes	X'2304'	Incorrect telephone number specified
Actions	X'3122'	Contact called DTE's operator
	X'0103'	Verify telephone number
	X'1204'	Attempt to reestablish the connection
Install Causes	(none)	
Failure Causes	X'2310'	Called DTE not responding
	X'200A'	ISDN communications error
Actions	X'1206'	Wait and retry
	X'3302'	If problem continues to occur repeatedly then do the following
	X'3106'	Contact ISDN network information service
	X'32D0'	Report the following
	X'82' SF	Calling telephone number
	X'82' SF	Telephone number called
	X'82' SF	Call progress signal
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (91))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (91))
		Type = X'FA'(D-channel)

(91) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.62 Q.931 Alert 6

Alert Condition:

This Alert is generated as a result of a call that has been rejected due to cause code number 21, call rejected. This cause indicates that the equipment sending this cause does not wish to accept the call, although it could have accepted the call because the equipment sending this cause is neither busy nor incompatible.

Alert ID Number		X'D865FCB3'
Alert Type	X'01'	Permanent
Alert Description	X'3406'	ISDN Call Control error
Probable Causes	X'2300'	Connection not established
User Causes	X'2304'	Incorrect telephone number specified
Actions	X'3132'	Contact called DTE's operator
	X'0103'	Verify telephone number
	X'1204'	Attempt to reestablish the connection
Install Causes	(none)	
Failure Causes	X'230C'	Call rejected by the called DTE
	X'200A'	ISDN communications error
Actions	X'1206'	Wait and retry
	X'3302'	If problem continues to occur repeatedly then do the following
	X'3106'	Contact ISDN network information service
	X'32D0'	Report the following
	X'82' SF	Calling telephone number
	X'82' SF	Telephone number called
	X'82' SF	Call progress signal
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (92))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (92))
		Type = X'FA'(D-channel)

(92) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.63 Q.931 Alert 7

Alert Condition:

This Alert is generated as a result of a call that has been rejected due to cause code number 22, number changed. This cause indicates that the called number is no longer assigned. The new number is included in the diagnostic field.

Alert ID Number		X'8ED0D26D'
Alert Type	X'01'	Permanent
Alert Description	X'3406'	ISDN Call Control error
Probable Causes	X'2306'	Changed number
User Causes	X'2306'	New telephone number assigned to called DTE
Actions	X'13A3'	Repeat call with (sf 82)
	X'82' SF	new number
Install Causes	(none)	
Failure Causes	(none)	
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (93))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (93))
		Type = X'FA'(D-channel)

(93) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.64 Q.931 Alert 8

Alert Condition:

This Alert is generated as a result of a call that has been rejected due to cause code number 26, non-selected user clearing. This cause indicates that the user has not been awarded the incoming call.

Alert ID Number		X'2534BDF0'
Alert Type	X'01'	Permanent
Alert Description	X'3406'	ISDN Call Control error
Probable Causes	X'200A'	ISDN network
	X'2201'	Called DTE
User Causes	X'2304'	Incorrect telephone number specified
Actions	X'3122'	Contact called DTE's operator
	X'0103'	Verify telephone number
	X'1204'	Attempt to reestablish the connection
Install Causes	(none)	
Failure Causes	X'200A'	ISDN communications error
	X'2201'	Called DTE
Actions	X'1206'	Wait and retry
	X'3302'	If problem continues to occur repeatedly then do the following
	X'3106'	Contact ISDN network information service
	X'32D0'	Report the following
	X'82' SF	Calling telephone number
	X'82' SF	Telephone number called
	X'82' SF	Call progress signal
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (94))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (94))
		Type = X'FA'(D-channel)

(94) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.65 Q.931 Alert 9

Alert Condition:

This Alert is generated as a result of a call that has been rejected due to cause code number 27, destination out of order. This cause indicates that the destination can not be reached because the interface to the destination is not functioning correctly such that the signalling message was unable to be delivered to the destination. This could be a physical

layer or data link layer failure in the destination.

Alert ID Number		X'85699B16'
Alert Type	X'01'	Permanent
Alert Description	X'3406'	ISDN Call Control error
Probable Causes	X'2201'	Called DTE
	X'2303'	Called number out of order
User Causes	X'2304'	Incorrect telephone number specified
Actions	X'3122'	Contact called DTE's operator
	X'0103'	Verify telephone number
	X'1204'	Attempt to reestablish the connection
Install Causes	X'3406'	Remote telecommunication cable not properly connected
Actions	X'0302'	Check cables and their connections
Failure Causes	X'2201'	Called DTE
	X'3302'	Communications adapter
Actions	X'3109'	Contact personnel responsible for connection to ISDN network
	X'32C0'	Report the following
	X'82' SF	Telephone number called
	X'82' SF	Call progress signal
	X'3106'	Contact ISDN network information service
	X'32D0'	Report the following
	X'82' SF	Calling telephone number
	X'82' SF	Telephone number called
	X'82' SF	Call progress signal
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (95))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (95))
		Type = X'FA'(D-channel)

(95) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.66 Q.931 Alert 10

Alert Condition:

This Alert is generated as a result of one of the following conditions:

- A call has been rejected due to resource unavailable, the cause code received is number 34, no circuit/channel available. This cause indicates that there is no appropriate circuit/channel presently available to handle the call.

- A call has been rejected/terminated due to resource unavailable, the cause code received is number 41, temporary failure. This cause indicates that the network is not functioning correctly and that the condition is not likely to last a long period of time. User may wish to try another call attempt almost immediately.

- A call has been rejected due to resource unavailable, the cause code received is number 47, resource unavailable, unspecified. This cause indicates a resource unavailable event when no other cause in the resource unavailable class applies.

Alert ID Number		X'21DFF757'
Alert Type	X'01'	Permanent
Alert Description	X'3406'	ISDN Call Control error
Probable Causes	X'200A'	ISDN network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'200A'	ISDN communications error
	X'3534'	Temporary lack of resources in the ISDN network
Actions	X'1206'	Wait and retry
	X'3302'	If problem continues to occur repeatedly then do the following
	X'3106'	Contact ISDN network information service
	X'32D0'	Report the following
	X'82' SF	Calling telephone number
	X'82' SF	Telephone number called
	X'82' SF	Call progress signal
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (96))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (96))
		Type = X'FA'(D-channel)

(96) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.67 Q.931 Alert 11

Alert Condition:

This Alert is generated as a result of a call that has been rejected due to cause code number 38, network out of order. This cause indicates that the network is not functioning correctly and that the condition is likely to last a relatively long period of time.

Alert ID Number		X'251EEAE0'
Alert Type	X'01'	Permanent
Alert Description	X'3406'	ISDN Call Control error
Probable Causes	X'200A'	ISDN network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'200A'	ISDN communications error
	X'3535'	Long-term lack of resources in the ISDN network
Actions	X'3106'	Contact ISDN network information service
	X'32D0'	Report the following
	X'82' SF	Calling telephone number
	X'82' SF	Telephone number called
	X'82' SF	Call progress signal
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (97))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (97))
		Type = X'FA'(D-channel)

(97) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.68 Q.931 Alert 12

Alert Condition:

This Alert is generated as a result of a call that has been rejected due to cause code number 42, switching equipment congestion. This cause indicates that the switching equipment generating this cause is experiencing a period of high traffic.

Alert ID Number		X'C5EC0B8E'
Alert Type	X'01'	Permanent
Alert Description	X'5001'	Network congestion
Probable Causes	X'200A'	ISDN network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'200A'	ISDN communications error
	X'3534'	Temporary lack of resources in the ISDN network
Actions	X'1206'	Wait and retry
	X'3302'	If problem continues to occur repeatedly then do the following
	X'3106'	Contact ISDN network information service
	X'32D0'	Report the following
	X'82' SF	Calling telephone number
	X'82' SF	Telephone number called
	X'82' SF	Call progress signal
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (98))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (98))
		Type = X'FA'(D-channel)

---

(98) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.69 Q.931 Alert 13

Alert Condition:

This Alert is generated as a result of one of the following conditions:

- A call has been rejected due to cause code number 29, facility

rejected. This cause indicates that a facility requested by the user cannot be provided by the network.

° A call has been rejected due to cause code number 50, invalid facility. This cause indicates that a facility requested by the user is invalid.

Alert ID Number		X'6CD51756'
Alert Type	X'01'	Permanent
Alert Description	X'3406'	ISDN Call Control error
Probable Causes	X'200A'	ISDN network
	X'2307'	Invalid request
User Causes	X'2315'	Facility parameter invalid or not supported
Actions	X'1512'	Change call setup parameters
	X'1204'	Attempt to reestablish the connection
Install Causes	(none)	
Failure Causes	X'200A'	ISDN communications error
	X'2311'	Service not available or not supported
Actions	X'3106'	Contact ISDN network information service
	X'32D0'	Report the following
	X'82' SF	Calling telephone number
	X'82' SF	Telephone number called
	X'82' SF	Call progress signal
	X'3109'	Contact personnel responsible for connection to ISDN network
	X'32C0'	Report the following
	X'82' SF	Telephone number called
	X'82' SF	Call progress signal
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (99))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (99))
		Type = X'FA'(D-channel)

(99) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.70 Q.931 Alert 14

Alert Condition:

This Alert is generated as a result of one of the following conditions:

- A call has been rejected due to cause code number 57, bearer capability not authorized. This cause indicates that the user has requested a bearer capability which is implemented by the equipment that generated this cause, but the user is not authorized to use it.

- A call has been rejected due to service/option not available, the cause code received is number 58, bearer capability not presently available. This cause indicates that the user has requested a bearer capability which is implemented by the equipment generated this cause but the user is not available at this time.

- A call has been rejected due to service/option not available, the cause code received is number 63, service/option not available, unspecified. This cause indicates that a service or option is not available but no other cause in the service/option not available class applies.

- A call has been rejected due to service/option not implemented, the cause code received is number 65, bearer capability not implemented. This cause indicates that the user has requested a bearer capability which is not implemented by the equipment generated this cause.

- A call has been rejected due to service/option not implemented, the cause code received is number 70, only restricted digital information bearer capability is available. This cause indicates that the user has requested an unrestricted bearer service but that the equipment sending this cause does only supports the restricted version of the requested bearer capability.

- A call has been rejected due to service/option not implemented, the cause code received is number 79, service/option not implemented. This cause indicates that a service/option not implemented event only when no other cause in the service/option not implemented class applies.

Alert ID Number		X'3385EA5E'
Alert Type	X'01'	Permanent
Alert Description	X'3406'	ISDN Call Control error
Probable Causes	X'2307'	Invalid request
User Causes	X'230D'	Service or Subscription parameter error
	X'2308'	Calling DTE does not subscribe to this facility
Actions	X'1512'	Change call setup parameters
	X'1204'	Attempt to reestablish the connection
Install Causes	(none)	
Failure Causes	X'2311'	Service not available or not supported
Actions	X'3106'	Contact ISDN network information service
	X'32D0'	Report the following

	X'82' SF	Calling telephone number
	X'82' SF	Telephone number called
	X'82' SF	Call progress signal
	X'3109'	Contact personnel responsible for connection to ISDN network
	X'32C0'	Report the following
	X'82' SF	Telephone number called
	X'82' SF	Call progress signal
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (100))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (100))
		Type = X'FA'(D-channel)

(100) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.71 Q.931 Alert 15

Alert Condition:

This Alert is generated as a result of one of the following conditions:

- A call has been rejected due to cause code number 81, invalid call reference value. This cause indicates that the equipment sending this cause has received a message with a call reference which is not currently in use on the user network interface.

- A call has been rejected due to cause code number 83, a suspended call exist, but this call identity does not. This cause indicates that a call resume has been attempted with a call identity which differs from that is in use for any presently suspended call(s).

- A call has been rejected due to cause code number 84, call identity in use. This cause indicates that a call suspend request contained a call identity which is already in use for a suspended call for another suspended call.

- A call has been rejected due to cause code number 85, no call suspended. This cause indicates that a call resume request has been received and it contained a call identity which does not identify any suspended call .

- A call has been rejected due to cause code number 86, call having the requested call identity that has been cleared. This cause indicates that a call resume request contained a call identity which once indicated a suspended call; however, that suspended call was cleared while suspended.

Alert ID Number		X'61A14221'
Alert Type	X'01'	Permanent
Alert Description	X'3406'	ISDN Call Control error

Probable Causes	X'2307'	Invalid request
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2312'	Call reference number error
Actions	X'1204'	Attempt to reestablish the connection
	X'3302'	If problem continues to occur repeatedly then do the following
	X'3106'	Contact ISDN network information service
	X'32D0'	Report the following
	X'82' SF	Calling telephone number
	X'82' SF	Telephone number called
	X'82' SF	Call progress signal
	X'3109'	Contact personnel responsible for connection to ISDN network
	X'32C0'	Report the following
	X'82' SF	Telephone number called
	X'82' SF	Call progress signal
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (101))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (101))
		Type = X'FA'(D-channel)

(101) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.72 Q.931 Alert 16

Alert Condition:

This Alert is generated as a result of one of the following conditions:

- A call has been rejected due to cause code number 95, invalid message, unspecified. This cause indicates that an invalid message event when no other cause in the invalid message class applies.

- A call that has been rejected due to cause code number 96, mandatory information element is missing. This cause indicates that the equipment sending this cause has received a message which is missing an information element which must be present before that message can be processed.

- A call has been rejected due to cause code number 97, message type non-existent or not implemented. This cause indicates that the equipment sending this cause has received a message with a message type it does not recognize either it is not defined or not

implemented.

- A call has been rejected/terminated due to protocol error class, the cause code received is number 98, message type not compatible with call state or message type non-existent or not implemented. This cause indicates that the equipment sending this cause has received a message such that the procedures do not indicate that this is a permissible message to receive while in the call state.

- A call has been rejected/terminated due to protocol error class, the cause code received is number 99, information element non-existent or not implemented. This cause indicates that the equipment sending this cause has received a message which includes information element not recognized because the information element identifier is not defined or it is defined but not implemented by the equipment sending this cause.

- A call has been rejected/terminated due to protocol error class, the cause code received is number 100, invalid information element contents. This cause indicates that the equipment sending this cause has received an information element which it has implemented; however, one or more of the fields in the information element are coded in such a way which has not been implemented by the equipment sending this cause.

- A call has been rejected/terminated due to protocol error class, the cause code received is number 101, message not compatible with call state. This cause indicates that the message has been received which is incompatible with the call state.

- A call has been rejected due to cause code number 111, protocol error, unspecified. This cause indicates a protocol error event only when no other cause in the protocol error class applies.

Alert ID Number		X'6B440A05'
Alert Type	X'01'	Permanent
Alert Description	X'3406'	ISDN Call Control error
Probable Causes	X'200A'	ISDN Network
User Causes	X'2313'	Incomplete or malformed call setup message
Actions	X'3109'	Contact personnel responsible for connection to ISDN network
	X'32C0'	Report the following
	X'82' SF	Telephone number called
	X'82' SF	Call progress signal
Install Causes	(none)	
Failure Causes	X'2311'	Service not available or not supported
Actions	X'3109'	Contact personnel responsible for connection to ISDN network
	X'32C0'	Report the following
	X'82' SF	Telephone number called
	X'82' SF	Call progress signal
	X'3106'	Contact ISDN network information service
	X'32D0'	Report the following
	X'82' SF	Calling telephone number

	X'82' SF	Telephone number called
	X'82' SF	Call progress signal
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (102))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (102))
		Type = X'FA'(D-channel)

---

(102) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.73 Q.931 Alert 17

Alert Condition:

This Alert is generated as a result of a call that has been rejected due to cause code number 88, incompatible destination. This cause indicates that the equipment sending this cause has received a request to establish a call which has low layer compatibility, high layer compatibility, or other compatibility attributes that can not be accommodated.

Alert ID Number		X'6E8813BD'
Alert Type	X'01'	Permanent
Alert Description	X'3406'	ISDN Call Control error
Probable Causes	X'2201'	Called DTE
	X'1022'	Communications program
User Causes	X'2304'	Incorrect telephone number specified
	X'230A'	User classes of service incompatible
Actions	X'3122'	Contact called DTE's operator
	X'0103'	Verify telephone number
	X'1512'	Change call setup parameters
	X'1204'	Attempt to reestablish the connection
Install Causes	(none)	
Failure Causes	X'2201'	Called DTE
Actions	X'3109'	Contact personnel responsible for connection to ISDN network
	X'32D0'	Report the following
	X'82' SF	Calling telephone number

	X'82' SF	Telephone number called
	X'82' SF	Call progress signal
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (103))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (103))
		Type = X'FA'(D-channel)

---

(103) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.74 Q.931 Alert 18

Alert Condition:

This Alert is generated as a result of one of the following conditions:

- This Alert is generated as a result of a call that has been rejected due to cause code number 102, recovery on timer expiration. This cause indicates that a procedure has been initiated by the expiration of a timer in association with call set-up error handling procedure.

- T308 timer was activated when Release was sent. It expired without receivng the Receive Complete or a Release from the network. The ISDN port will be put into the maintenance state.

Alert ID Number		X'946BDCC4'
Alert Type	X'01'	Permanent
Alert Description	X'3406'	ISDN Call Control error
Probable Causes	X'200A'	ISDN network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'200A'	ISDN communications error
	X'20A1'	No response from ISDN network -- (sf 82) expired
	X'82' SF	Timer
Actions	X'3106'	Contact ISDN network information service
	X'32D0'	Report the following
	X'82' SF	Calling telephone number
	X'82' SF	Telephone number called

	X'82' SF	Call progress signal
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (104))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (104))
		Type = X'FA'(D-channel)

---

(104) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.75 Q.931 Alert 19

Alert Condition:

A call that has been rejected due to cause code number 127, interworking, unspecified. This cause indicates that there has been interworking problem with a network which does not provide causes for action it takes; thus the precise cause for a message that is being sent cannot be ascertained.

Alert ID Number		X'4C7B6502'
Alert Type	X'01'	Permanent
Alert Description	X'3406'	ISDN Call Control error
Probable Causes	X'200A'	ISDN network
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'200A'	ISDN communications error
	X'2314'	ISDN to non-ISDN operation error
Actions	X'3106'	Contact ISDN network information service
	X'32D0'	Report the following
	X'82' SF	Calling telephone number
	X'82' SF	Telephone number called
	X'82' SF	Call progress signal
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender
		Name = (D-channel (105))

		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (105))
		Type = X'FA'(D-channel)

---

(105) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.4.4.76 Q.931 Alert 20

Alert Condition:

A call has been rejected due to cause code number 43, access information discarded. This cause indicates that the network could not deliver access information to the remote user as requested; i.e., user-user information, low layer compatibility, high layer compatibility, or subaddress as indicated in the diagnostic.

Alert ID Number		X'36390AEE'
Alert Type	X'01'	Permanent
Alert Description	X'3406'	ISDN Call Control error
Probable Causes	X'200A'	ISDN network
	X'230C'	Service not available or not supported
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'200A'	ISDN communications error
	X'2311'	Service not available or not supported
Actions	X'3106'	Contact ISDN network information service
	X'32D0'	Report the following
	X'82' SF	Calling telephone number
	X'82' SF	Telephone number called
	X'82' SF	Call progress signal
	X'3109'	Contact personnel responsible for connection to ISDN network
	X'32C0'	Report the following
	X'82' SF	Telephone number called
	X'82' SF	Call progress signal
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		First resource below sender

		Name = (D-channel (106))
		Type = X'FA'(D-channel)
	X'11' SF	Associated Resources
		Name = (D-channel (106))
		Type = X'FA'(D-channel)

(106) See "B- and D-Channel Names in ISDN Alerts" in topic A.4.4.1 for the form of this channel name.

A.5 Alerts for APPN Nodes

This section defines Alerts sent by APPN node components.

Subtopics:

- A.5.1 Address Space Manager
- A.5.2 Configuration Services
- A.5.3 Topology Database Manager
- A.5.4 Directory Services
- A.5.5 Session Connector Manager
- A.5.6 Management Services
- A.5.7 Path Control
- A.5.8 Session Services

A.5.1 Address Space Manager

Subtopics:

- A.5.1.1 Address Space Manager Alert CPAM001

A.5.1.1 Address Space Manager Alert CPAM001

Alert Condition:

Session setup failed with one of the following SNA sense data:

```
X'1001 0003'    X'2011 0001'    X'8007 0000'    X'800F 0000'
X'1002 0000'    X'2011 0002'    X'8007 0001'    X'800F 0001'
X'2011 0000'                    X'8007 0002'
```

Alert ID Number		X'B558D310' (note 1)
Alert Type	X'01'	Permanent
Alert Description	X'3112'	SNA session setup failure
Probable Causes	X'1023'	Communications program in remote node
User Causes	(none)	
Install Causes	(none)	

Failure Causes	X'1023'	Communications program in remote node
	X'F0BE'	BIND processing failed with (82)
	X'82' SF	(SNA sense data)
Actions	X'3000'	Contact appropriate service representative
	X'32A0'	Report the following:
	X'82' SF	(Product Alert Reference Code) (note 2)
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(Partner CP name)
		Type=X'F4' (CP)

Notes:

1. This Alert was previously documented with the incorrect Alert ID value of X'ABB3D962'. Alert receivers should be able to accept Alerts containing either the current, correct value or the old, incorrect value.

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

   An alternative to this latter use of the Product Alert Reference Code could be to include the detecting module name.

A.5.2 Configuration Services

Subtopics:

- A.5.2.1 Configuration Services Alert CPCS001
- A.5.2.2 Configuration Services Alert CPCS002
- A.5.2.3 Configuration Services Alert CPCS003
- A.5.2.4 Configuration Services Alert CPCS004
- A.5.2.5 Configuration Services Alert CPCS005

A.5.2.1 Configuration Services Alert CPCS001

Alert Condition:

XID negotiation was terminated by this node because the received XID was invalid in format, or contained unacceptable values. This Alert covers the following SNA sense data:

```
X'0806 002C'  X'0809 0054'  X'0809 0055'  X'086F 0000'
X'088C nnyy'  X'088C 1000'  X'0891 0004'  X'0891 0005'
X'0895 xxyy'  X'1015 0001'  X'1015 0002'  X'1016 0001'
X'1016 0002'  X'1016 0003'  X'1016 0004'  X'1016 0005'
X'1016 0006'  X'1016 0007'  X'1016 0008'  X'1016 0009'
X'1016 000A'  X'1016 000B'  X'1016 000C'  X'1016 000D'
X'1016 000E'  X'1016 000F'  X'1016 0010'  X'1016 0011'
X'1016 0012'  X'1016 0013'  X'1016 0014'  X'1016 0015'
X'1016 0016'  X'1016 0017'  X'1016 0018'  X'1016 001A'
X'1016 001B'  X'1016 001C'  X'1016 001D'  X'1016 001E'
X'1016 0020'
```

where

nn,yy missing control vector key, Network Name Type

xx,yy key of control vector in error, byte offset of error

In addition to being signaled in the alert the sense data is sent in the XID frame.

Alert ID Number		X'EBEE390E' (note 1)
Alert Type	X'01'	Permanent
Alert Description	X'3111'	Invalid XID received
Probable Causes	X'1023'	Communications program in remote node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1023'	Communications program in remote node
	X'F0A4'	XID negotiation failed with
	X'82' SF	(SNA sense data)
Actions	X'3110'	Contact communications systems programmer
	X'32A0'	Report the following:
	X'82' SF	(Product Alert Reference Code) (note 2)
Additional SVs	X'47' SV	MSU Correlation (note 3)
	X'01' SF	(Link Instance Identifier)
	X'48' SV	Supporting Data Correlation (note 4)
	X'82' SF	(File Name)
	X'82' SF	(Log Record Number)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(Adapter number)
		Type=X'21' (ADAPTER)
		Second resource below sender:
		Name=(CP name taken from X'0E' CV) or
		'UNKNOWN' (note 5)
	X'10' SV	Type=X'F4' (CP)
		Indicated Resource Product Set ID
		(Received X'10' CV) (note 6)

Notes:

1. This Alert was previously documented with the incorrect Alert ID value of X'CF9E7A2F'. Alert receivers should be able to accept Alerts containing either the current, correct value or the old, incorrect value.

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

   An alternative to this latter use of the Product Alert Reference Code could be to include the detecting module could be to include the detecting module name.

3. This SV is included only when XID negotiation has progressed far enough that the correlator can be constructed.

4. The Supporting Data Correlation (X'48') SV is present to report that diagnostic data for this Alert was stored by the Alert sender. For most XID-related errors, the appropriate supporting data would be copies of the XIDs in question (both the XID sent and the one received).

5. The CP name, which is the second resource below the Alert sender, is taken from the Network Name (X'0E') CV (type X'F4') which was present in the received XID. For back-level LEN end nodes, this is the node identifier field from the received XID-3. However, for those cases in which the CP name is not available, this entry in the Hierarchy Name List (X'10') SF is 'UNKNOWN'.

6. The Product Set ID (X'10') SV, is taken from the Product Set ID (X'10') CV which was present in the received XID. However, for those cases in which the PSID is not available, this SV is omitted.

A.5.2.2 Configuration Services Alert CPCS002

Alert Condition:

XID negotiation was terminated by this node because the remote node violated protocols for XID exchange. This Alert covers the following SNA sense data:

X'0809 003A'	X'0809 003F'	X'0809 0047'
X'0809 003B'	X'0809 0041'	X'0809 004E'
X'0809 003C'	X'0809 0042'	
X'0809 003D'	X'0809 0045'	
X'0809 003E'	X'0809 0046'	

In addition to being signaled in the alert the sense data is sent in the XID frame.

Alert ID Number		X'034A6F0B' (note 1)
Alert Type	X'01'	Permanent
Alert Description	X'3110'	XID protocol error
Probable Causes	X'1023'	Communications program in remote node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1023'	Communications program in remote node
	X'F0A4'	XID negotiation failed with
	X'82' SF	(SNA sense data)

Actions	X'3110'	Contact communications systems programmer
	X'32A0'	Report the following:
	X'82' SF	(Product Alert Reference Code) (note 2)
Additional SVs	X'47' SV	MSU Correlation (note 3)
	X'01' SF	(Link Instance Identifier)
	X'48' SV	Supporting Data Correlation (note 4)
	X'82' SF	(File Name)
	X'82' SF	(Log Record Number)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(Adapter number)
		Type=X'21' (ADAPTER)
		Second resource below sender:
		Name=(CP name taken from X'0E' CV) or
		'UNKNOWN' (note 5)
		Type=X'F4' (CP)
	X'10' SV	Indicated Resource Product Set ID
		(Taken from X'10' CV) (note 6)

Notes:

1. This Alert was previously documented with the incorrect Alert ID value of X'3632FDC1'. Alert receivers should be able to accept Alerts containing either the current, correct value or the old, incorrect value.

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

   An alternative to this latter use of the Product Alert Reference Code could be to include the detecting module name.

3. This SV is included only when XID negotiation has progressed far enough that the correlator can be constructed.

4. The Supporting Data Correlation (X'48') SV is present to report that diagnostic data for this Alert was stored by the Alert sender. For most XID-related errors, the appropriate supporting data would be copies of the XIDs in question (both the XID sent and the one received).

5. The CP name, which is the second resource below the Alert sender, is taken from the Network Name (X'0E') CV (type X'F4') which was present in the received XID. However, for those cases in which the CP name is not available, this entry in the Hierarchy Name List (X'10') SF is 'UNKNOWN'.

6. The Product Set ID (X'10') SV, is taken from the Product Set ID (X'10') CV which was present in the received XID. However, for those cases in which the PSID is not available, this SV is omitted.

A.5.2.3 Configuration Services Alert CPCS003

Alert Condition:

Invalid SET MODE was received. This Alert covers the following SNA sense data:

X'0809 0040'

The sense data is sent only in the alert.

Alert ID Number		X'0DF28A14' (note 1)
Alert Type	X'01'	Permanent
Alert Description	X'1605'	Wrong link mode setting command received
Probable Causes	X'1023'	Communications program in remote node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1023'	Communications program in remote node
Actions	X'3110'	Contact communications system programmer
	X'32C0'	Report the following:
	X'82' SF	(SNA sense data)
	X'82' SF	(Product Alert Reference Code) (note 2)
Additional SVs	X'47' SV	MSU Correlation (note 3)
	X'01' SF	(Link Instance Identifier)
	X'48' SV	Supporting Data Correlation (note 4)
	X'82' SF	(File Name)
	X'82' SF	(Log Record Number)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(Adapter number)
		Type=X'21' (ADAPTER)
		Second resource below sender:
		Name=(CP name taken from X'0E' CV) or
		'UNKNOWN' (note 5)
		Type=X'F4' (CP)
	X'10' SV	Indicated Resource Product Set ID
		(Received X'10' CV) (note 6)

Notes:

1. This Alert was previously documented with the incorrect Alert ID value of X'5BEE4A45'. Alert receivers should be able to accept Alerts containing either the current, correct value or the old, incorrect value.

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

   An alternative to this latter use of the Product Alert Reference Code could be to include the detecting module name.

3. This SV is included only when XID negotiation has progressed far enough that the correlator can be constructed.

4. The Supporting Data Correlation (X'48') SV is present to report that diagnostic data for this Alert was stored by the Alert sender. For this Alert, the supporting data is the SET MODE received.

5. The CP name, which is the second resource below the Alert sender, is taken from the Network Name (X'0E') CV (type X'F4') which was present in the received XID. For migration nodes, this is the node identifier field from the received XID-3. However, for those cases in which the CP name is not available, this entry in the Hierarchy Name List (X'10') SF is 'UNKNOWN'.

6. The Product Set ID (X'10') SV, is taken from the Product Set ID (X'10') CV which was present in the received XID. However, for those cases in which the PSID is not available, this SV is omitted.

A.5.2.4 Configuration Services Alert CPCS004

Alert Condition:

XID negotiation was terminated by the remote node. CV22 was received. This Alert covers the following SNA sense data:

X'0809 0049'

Alert ID Number		X'6D27D125' (note 1)
Alert Type	X'01'	Permanent
Alert Description	X'1604'	XID negotiation terminated
Probable Causes	X'1022'	Communications program
	X'1023'	Communications program in remote node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1022'	Communications program
	X'F0A4'	XID negotiation failed with
	X'82' SF	(SNA sense data)
Actions	X'3110'	Contact communications systems programmer
	X'32A0'	Report the following:
	X'82' SF	(Product Alert Reference Code) (note 2)

Additional SVs	X'47' SV	MSU Correlation (note 3)
	X'01' SF	(Link Instance Identifier)
	X'48' SV	Supporting Data Correlation (note 4)
	X'82' SF	(File Name)
	X'82' SF	(Log Record Number)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(Adapter number)
		Type=X'21' (ADAPTER)
		Second resource below sender:
		Name=(CP name taken from X'0E' CV) or
		'UNKNOWN' (note 5)
		Type=X'F4' (CP)
	X'10' SV	Indicated Resource Product Set ID
		(Taken from X'10' CV) (note 6)

Notes:

1. This Alert was previously documented with the incorrect Alert ID value of X'1E4395E7'. Alert receivers should be able to accept Alerts containing either the current, correct value or the old, incorrect value.

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

   An alternative to this latter use of the Product Alert Reference Code could be to include the detecting module name.

3. This SV is included only when XID negotiation has progressed far enough that the correlator can be constructed.

4. The Supporting Data Correlation (X'48') SV is present to report that diagnostic data for this Alert was stored by the Alert sender. For most XID-related errors, the appropriate supporting data would be copies of the XIDs in question (both the XID sent and the one received).

5. The CP name, which is the second resource below the Alert sender, is taken from the Network Name (X'0E') CV (type X'F4') which was present in the received XID. However, for those cases in which the CP name is not available, this entry in the Hierarchy Name List (X'10') SF is 'UNKNOWN'.

6. The Product Set ID (X'10') SV, is taken from the Product Set ID (X'10') CV which was present in the received XID. However, for those cases in which the PSID is not available, this SV is omitted.

A.5.2.5 Configuration Services Alert CPCS005

Alert Condition:

XID negotiation was terminated by the Alert sender because no response was received and the XID retry limit was reached. This Alert covers the following SNA sense data:

```
X'083E 0001'
X'083E 0002'
```

Alert ID Number		X'F52A0C01'
Alert Type	X'01'	Permanent
Alert Description	X'3305'	Unable to communiate with remote node
Probable Causes	X'1023'	Communications program in remote node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1023'	Communications program in remote node
	X'F018'	XID poll count exhausted
Actions	X'010D'	Check configuration of the remote node
	X'1200'	Retry
Additional SVs	X'98' SV	Detailed Data
	X'82' SF	(Product Alert Reference Code) (note 1)
	X'82' SF	(SNA sense data)
	X'47' SV	MSU Correlation (note 4)
	X'01' SF	(Link Instance Identifier)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(Adapter number)
		Type=X'21' (ADAPTER)
		Second resource below sender:
		Name=(CP name taken from X'0E' CV) or
		'UNKNOWN' (note 2)
		Type=X'F4' (CP)
	X'10' SV	Indicated Resource Product Set ID
		(Taken from X'10' CV) (note 3)

Notes:

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

   An alternative to this latter use of the Product Alert Reference Code could be to include the detecting module name.

2. The CP name, which is the second resource below the Alert sender, is taken from the Network Name (X'0E') CV (type X'F4') which was present in the received XID. For back-level LEN end nodes, this is the node identifier field from the received XID-3. However, for those cases in which the CP name is not available, this entry in the Hierarchy Name List (X'10') SF is 'UNKNOWN'.

3. The Product Set ID (X'10') SV, is taken from the Product Set ID (X'10') CV which was present in the received XID. However, for those cases in which the PSID is not available, this SV is omitted.

4. This SV is included only when XID negotiation has progressed far enough that the correlator can be constructed.

| A.5.3 Topology Database Manager

Subtopics:

- A.5.3.1 Topology and Routing Service Alert CPDB001
- A.5.3.2 Topology and Routing Services Alert CPDB002
- A.5.3.3 Topology and Routing Service Alert CPDB003

| A.5.3.1 Topology and Routing Service Alert CPDB001

| *Alert Condition:*

| Format error detected in a topology database update (TDU) GDS variable.
| This Alert covers the following SNA sense data:

```
|   X'086C 4400'   X'0895 44yy'   X'08A0 0004'   X'1002 0000'
|   X'086C 8000'   X'0895 45yy'   X'08A0 0005'   X'1003 000D'
|   X'086D 4580'   X'0895 46yy'   X'08A0 0006'   X'101A nnmm'
|   X'086D 4680'   X'0895 47yy'
```

| The topology and routing services component instructs session services to
| deactivate the CP-CP session over which the TDU was received. The
| associated sense data is sent in the UNBIND. The TDU causing the Alert is
| ignored.

Alert ID Number		X'B8E072C2'
Alert Type	X'01'	Permanent
Alert Description	X'3113'	CP-CP Session Failure
Probable Causes	X'1023'	Communications program in remote node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1023'	Communications program in remote node
	X'2205'	Topology Protocol Error
Actions	X'3000'	Contact appropriate service representative
	X'32C0'	Report the following:
	X'82' SF	(SNA sense data)
	X'82' SF	(Product Alert Reference Code) (note 1)

Additional SVs	X'98' SV	Detailed Data
	X'82' SF	(NetID) (note 2)
	X'48' SV	Supporting Data Correlation (note 3)
	X'82' SF	(File Name)
	X'82' SF	(Log Record Number)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(Partner CP name)
		Type=X'F4' (CP)

Notes:

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

   An alternative to this latter use of the Product Alert Reference Code could be to include the detecting module name.

2. The network ID may be needed if an end node not having the same net ID as its network node server is sending the alert.

3. The Supporting Data Correlation (X'48') SV is present to report that diagnostic data for this Alert was stored by the Alert sender. For this Alert, supporting data is the TDU (topology database update) that caused the error.

A.5.3.2 Topology and Routing Services Alert CPDB002

Alert Condition:

The node table is full. This condition may be the result of a system definition error (i.e., inadequate storage allocated for the table). If the node table is dynamically extendable, this may be a failure to obtain additional storage.

No SNA sense data is used in this Alert.

The TDU causing the Alert is ignored.

Alert ID Number		X'14CDF23C'
Alert Type	X'01'	Permanent
Alert Description	X'1606'	Topology capacity exceeded
Probable Causes	X'1022'	Communications program
User Causes	(none)	
Install Causes	X'8001'	Topology storage exceeded
Actions	X'3110'	Contact communications system programmer

	X'32C0'	Report the following:
	X'82' SF	(Maximum number of node table entries)
	X'82' SF	(Product Alert Reference Code) (note 1)
Failure Causes	(none)	
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)

Notes:

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

   An alternative to this latter use of the Product Alert Reference Code could be to include the detecting module name.

A.5.3.3 Topology and Routing Service Alert CPDB003

Alert Condition:

The architecturally allowed maximum for the resource sequence number (RSN) has been reached. The resource (a node or transmission group) is no longer available for use.

Note: This is not an error condition, but rather an architecturally allowed condition that can occur after some long period of time. At the rate of one TDU (topology database update) per second for any given resource, it will take 68 years for this condition to occur. However, certain error conditions can cause the RSN space to be used up faster than that.

No SNA sense data is used in this Alert.

The topology and routing services component continues to run but the TDU causing the Alert is discarded.

Alert ID Number		X'0CAB2FAE'
Alert Type	X'01'	Permanent
Alert Description	X'1607'	Topology Protocol error
Probable Causes	X'1022'	Communications program
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1022'	Communications program
Actions	X'3110'	Contact communications system

		programmer
	X'32C0'	Report the following:
	X'82' SF	(Name of the resource causing the error)
	X'82' SF	(Product Alert Reference Code) (note 1)
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)

Notes:

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

   An alternative to this latter use of the Product Alert Reference Code could be to include the detecting module name.

A.5.4 Directory Services

Subtopics:

- A.5.4.1 Directory Services Alert CPDN001
- A.5.4.2 Directory Services Alert CPDN002
- A.5.4.3 Directory Services Alert CPDN003

A.5.4.1 Directory Services Alert CPDN001

Alert Condition:

Protocol violation on a Locate or BIND request. This Alert covers the following SNA sense data:

Incorrect Locate request received (Find request not included). The Locate message must include a Find GDS variable. The Find was either left off by the originator of the Locate or deleted by a propagator of the Locate:

X'1010 A002'

Locate Procedure Status indicator bit was set to 1 (procedure is active, procedure correlation information is kept) in a reply to a broadcast, though reservation is not permitted on a broadcast:

X'1010 4004'

A broadcast search resulted in two or more conflicting positive replies that differ on the CP owning the target resource. Multiple replies are acceptable, as long as all indicate the same owning CP:

X'1010 0001'

Additional sense data:

X'0812 0016'	X'0890 0080'	X'0890 0048'
X'0891 0004'	X'0895 0000'	X'0897 0000'
X'1010 B080'	X'1014 003C'	X'1014 003D'
X'1014 0080'	X'1014 023C'	X'1014 A082'
X'1014 B280'	X'FFFF 0002'	

Alert ID Number		X'170F7710' (note 1)
Alert Type	X'01'	Permanent
Alert Description	X'3001'	Directory Services Protocol Error
Probable Causes	X'1023'	Communications program in remote node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2206'	Directory program remote node
Actions	X'3000'	Contact appropriate service representative
	X'32C0'	Report the following:
	X'82' SF	(SNA sense data)
	X'82' SF	(Product Alert Reference Code) (note 2)
Additional SVs	X'48' SV	Supporting Data Correlation (note 3)
	X'60' SF	(Fully Qualified Session PCID)
	X'82' SF	(File Name)
	X'82' SF	(Log Record Number)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
	X'11' SF	Associated Resources
		Name=(adjacent node)
		Type=X'F4' (CP)

Notes:

1. This Alert was previously documented with the incorrect Alert ID value of X'598E4438'. Alert receivers should be able to accept Alerts containing either the current, correct value or the old, incorrect value.

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

   An alternative to this latter use of the Product Alert Reference Code could be to include the detecting module name.

3. The Supporting Data Correlation (X'48') SV is present to report that diagnostic data for this Alert was stored by the Alert sender. For directory request errors, the whole request is included as supporting data.

A.5.4.2 Directory Services Alert CPDN002

Alert Condition:

Insufficient resources available for directory services. Deadlock detected between DS components in two nodes. When this condition is detected, DS will instruct SS to UNBIND the CP-CP session to the other node. If this problem reoccurs, it may indicate a node system definition error or a configuration problem. This Alert covers the following SNA sense data:

X'0812 000A'
X'0890 0060'

Alert ID Number		X'769022F0' (note 1)
Alert Type	X'01'	Permanent
Alert Description	X'3113'	CP-CP Session Failure
Probable Causes	X'1022'	Communications program
User Causes	(none)	
Install Causes	X'8002'	Insufficient storage for directory services
	X'8003'	Communications subsystem definition
Actions	X'3302'	If problem continues to occur repeatedly then do
		the following
	X'3110'	Contact communications System Programmer
	X'32C0'	Report the following:
	X'82' SF	(SNA sense data)
	X'82' SF	(Product Alert Reference Code) (note 2)
Failure Causes	(none)	
Additional SVs	X'48' SV	Supporting Data Correlation (note 3)
	X'60' SF	(Fully Qualified Session PCID)
	X'82' SF	(File Name)
	X'82' SF	(Log Record Number)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
	X'11' SF	Associated Resources
		Name=(adjacent node)
		Type=X'F4' (CP)

Notes:

1. This Alert was previously documented with the incorrect Alert ID value of X'7B5DEF8F'. Alert receivers should be able to accept Alerts containing either the current, correct value or the old, incorrect

value.

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

   An alternative to this latter use of the Product Alert Reference Code could be to include the detecting module name.

3. The Supporting Data Correlation (X'48') SV is present to report that diagnostic data for this Alert was stored by the Alert sender. For directory request errors, the whole request is included as supporting data.

A.5.4.3 Directory Services Alert CPDN003

Alert Condition:

An end node CP without authorization has attempted to delete a resource.

Probable cause is attempted network modification by an unauthorized user.
This alert covers the following sense data:

X'080E 0009'

Alert ID Number		X'E1497660'
Alert Type	X'12'	Unknown
Alert Description	X'C003'	Unauthorized network change attempted
Probable Causes	X'1023'	Communications program in remote node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2207'	Unauthorized network change attempted
Actions	X'3302'	If problem continues to occur repeatedly then do
		the following
	X'3110'	Contact communications systems programmer
	X'32C0'	Report the following:
	X'82' SF	(SNA sense data)
	X'82' SF	(Product Alert Reference Code) (note 1)
Additional SVs	X'48' SV	Supporting Data Correlation (note 2)
	X'82' SF	(File Name)
	X'82' SF	(Log Record Number)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		Sending node:
		Name=(CP name)

			Type=X'F4' (CP)
			First resource below sender:
			Name=(Partner CP name)
			Type=X'F4' (CP)

Notes:

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

   An alternative to this latter use of the Product Alert Reference Code could be to include the detecting module name.

2. The Supporting Data Correlation (X'48') SV is present to report that diagnostic data for this Alert was stored by the Alert sender. For directory request errors, the whole request is included as supporting data.

A.5.5 Session Connector Manager

Subtopics:

- A.5.5.1 Session Connector Manager Alert CPIM001
- A.5.5.2 Session Connector Manager Alert CPIM002
- A.5.5.3 Session Connector Manager Alert CPIM003

A.5.5.1 Session Connector Manager Alert CPIM001

Alert Condition:

ABEND of a session connector. This occurs when the session connector is unable to allocate storage for processing. Session connector manager deactivates the session. This Alert covers the following SNA sense data:

X'0812 000D'
X'085C 0003'

Alert ID Number		X'2110D168'
Alert Type	X'01'	Permanent
Alert Description	X'5004'	Out of resources
Probable Causes	X'1022'	Communications program
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2054'	Insufficient storage for intermediate session processing
Actions	X'3302'	If problem continues to occur repeatedly then do

		the following
	X'3110'	Contact communications systems programmer
	X'32C0'	Report the following:
	X'82' SF	(SNA sense data)
	X'82' SF	(Product Alert Reference Code) (note 1)
Additional SVs	X'48' SV	Supporting Data Correlation
	X'60' SF	(Fully Qualified Session PCID)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)

Notes:

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

   An alternative to this latter use of the Product Alert Reference Code could be to include the detecting module name.

A.5.5.2 Session Connector Manager Alert CPIM002

Alert Condition:

Session connector received an invalid RU. Session connector signals session connector manager to deactivate the session. This Alert covers the following SNA sense data:

X'8007 0000'	X'1002 0000'	X'2011 0002'
X'083B 0002'	X'1022 0001'	X'2011 0003'
X'1001 0003'	X'2001 0000'	X'086F 0000'
X'8007 0003'		

Alert ID Number		X'8FC7EF8D'
Alert Type	X'01'	Permanent
Alert Description	X'3100'	SNA protocol error
Probable Causes	X'2131'	Communications program in adjacent node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'2208'	Communications program in adjacent node
	X'F06D'	Isolated Pacing Message Failure
Actions	X'3110'	Contact communications systems

		programmer
	X'32C0'	Report the following:
	X'82' SF	(SNA sense data)
	X'82' SF	(Product Alert Reference Code) (note 1)
	X'F0F1'	Resource associated with the alert condition
	X'84' SF	(Associated resource)
Additional SVs	X'48' SV	Supporting Data Correlation (note 2)
	X'60' SF	(Fully Qualified Session PCID)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
	X'11' SF	Associated Resources
		Name=(name of intended destination of
		the BIND)
		Type=X'F3' (LU)

Notes:

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

   An alternative to this latter use of the Product Alert Reference Code could be to include the detecting module name.

2. The Supporting Data Correlation (X'48') SV is present to report that diagnostic data for this Alert was stored by the Alert sender. For this Alert the supporting data is the RU received.

A.5.5.3 Session Connector Manager Alert CPIM003

Alert Condition:

ABEND of a session connector for some reason other than "out of resources." Session connector manager deactivates the session. This Alert covers the following SNA sense data:

X'085C 0000'

Alert ID Number		X'AB9BD79D'
Alert Type	X'01'	Permanent
Alert Description	X'2100'	Software program error
Probable Causes	X'1022'	Communications program
User Causes	(none)	

Install Causes	(none)	
Failure Causes	X'2053'	Communications program abnormally terminated
Actions	X'3302'	If problem continues to occur repeatedly then do
		the following
	X'3110'	Contact communications systems programmer
	X'32D0'	Report the following:
	X'82' SF	(SNA sense data)
	X'82' SF	(Product Alert Reference Code) (note 1)
	X'82' SF	(ABEND code) (note 2)
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)

Notes:

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

   An alternative to this latter use of the Product Alert Reference Code could be to include the detecting module name.

2. ABEND code - ABEND code returned by implementation run-time component.

A.5.6 Management Services

Subtopics:

- A.5.6.1 Management Services Alert CPMS002

A.5.6.1 Management Services Alert CPMS002

Alert Condition:

Management Services protocol error. The received Multiple-domain support message unit (MDS-MU) cannot be processed. This Alert covers the following SNA sense data:

MDS Format Exceptions:

```
X'08AA 1310'   X'086C 9000'    X'086D 8202'
X'08AA 1311'   X'086D 8101'    X'086D 8203'
X'08AA 1549'   X'086D 8102'    X'086F 0002'
X'086C 8100'   X'086D 8103'    X'086F 8103'
X'086C 8200'   X'086D 8201'    X'086F 8203'
X'0870 8106'   X'0870 8206'    X'0870 9002'
X'0870 9003'
```

MS_Capabilities Format Exceptions:

```
X'080C 0005'   X'080C 0163'    X'086D 2111'
```

```
X'080C 0006'   X'080C 0164'    X'086D 2121'
X'080C 0012'   X'086C 2100'    X'086D 6240'
X'080C 01nn'   X'086D 2101'    X'086D 6241'
X'080C 0161'   X'086D 2102'    X'086F mm05'
X'080C 0162'   X'086D 2110'    X'086F 2105'
```

(nn = subvector id, mm = subfield id)

Alert ID Number		X'2313A399' (note 1)
Alert Type	X'01'	Permanent
Alert Description	X'3114'	Management Services protocol error
Probable Causes	X'1023'	Communication program in remote node
	X'1022'	Communication program
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1023'	Communication program in remote node
	X'1022'	Communication program
Actions	X'3000'	Contact appropriate service representative
	X'32C0'	Report the following:
	X'82' SF	(SNA sense data)
	X'82' SF	(Product Alert Reference Code) (note 2)
Additional SVs	X'48' SV	Supporting Data Correlation (note 3)
	X'82' SF	(File Name)
	X'82' SF	(Log Record Number)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)

Notes:

1. This Alert was previously documented with the incorrect Alert ID value of X'39E8BB74'. Alert receivers should be able to accept Alerts containing either the current, correct value or the old, incorrect value.

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

   An alternative to this latter use of the Product Alert Reference Code could be to include the detecting module name.

3. The Supporting Data Correlation (X'48') SV is present to report that diagnostic data for this Alert was stored by the Alert sender. For this Alert the supporting data is the received MDS-MU that caused the error.

A.5.7 Path Control

Subtopics:

- A.5.7.1 Path Control Alert CPPL001

A.5.7.1 Path Control Alert CPPL001

Alert Condition:

SNA protocol violation. The received path information unit (PIU) is discarded. This Alert covers the following SNA sense data:

```
X'8006 0000'
X'4005 0000'
X'1002 0000'
X'800B 0000'
X'8005 0000'
```

Alert ID Number		X'C781E91E' (note 1)
Alert Type	X'01'	Permanent
Alert Description	X'3100'	SNA protocol error
Probable Causes	X'1023'	Communications program in remote node
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1023'	Communications program in remote node
	X'F06E'	Invalid PIU (path information unit) received
Actions	X'3110'	Contact communications systems programmer
	X'32D0'	Report the following:
	X'82' SF	(SNA sense data)
	X'82' SF	(Product Alert Reference Code) (note 2)
	X'82' SF	(User-specified link name) (note 3)
Additional SVs	X'48' SV	Supporting Data Correlation (note 4)
	X'82' SF	(File Name)
	X'82' SF	(Log Record Number)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(TG over which the PIU

		was
		received)
		Type=X'2C' (transmission group)
		Second resource below sender:
		Name=(adjacent node)
		Type=X'F4' (CP)

Notes:

1. This Alert was previously documented with the incorrect Alert ID value of X'E6B578D3'. Alert receivers should be able to accept Alerts containing either the current, correct value or the old, incorrect value.

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

   An alternative to this latter use of the Product Alert Reference Code could be to include the detecting module name.

3. If an implementation provides for a user-specified link name in addition to the TG number and CP name, include that name here. If not available, the name is omitted.

4. The Supporting Data Correlation (X'48') SV is present to report that diagnostic data for this Alert was stored by the Alert sender. For this Alert the supporting data is the path information unit (PIU) rejected.

A.5.8 Session Services

Subtopics:

- A.5.8.1 Session Services Alert CPSS001
- A.5.8.2 Session Services Alert CPSS002
- A.5.8.3 Session Services Alert CPSS003

A.5.8.1 Session Services Alert CPSS001

Alert Condition:

Required transaction program cannot be started (initiated by setup CP-CP session).

Alert ID Number		X'EDD0D10F' (note 1)
Alert Type	X'01'	Permanent
Alert Description	X'3113'	CP-CP Session Failure
Probable Causes	X'8003'	Communication configuration
User Causes	(none)	
Install Causes	X'1301'	Communications program
Actions	X'3110'	Contact communications systems

		programmer
	X'32C0'	Report the following
	X'82' SF	(Product Alert Reference Code) (note 2)
	X'82' SF	(Transaction Program)
Failure Causes	(none)	
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)

Notes:

1. This Alert was previously documented with the incorrect Alert ID value of X'F60FBD84'. Alert receivers should be able to accept Alerts containing either the current, correct value or the old, incorrect value.

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

   An alternative to this latter use of the Product Alert Reference Code could be to include the detecting module name.

A.5.8.2 Session Services Alert CPSS002

Alert Condition:

BIND received from an end node that this network node does not serve. If the end node includes the APPN Route Selection control vector (RSCV) in the BIND, the network node does not make this check.

Alert ID Number		X'26F47FA9'
Alert Type	X'01'	Permanent
Alert Description	X'C004'	Unauthorized access attempted
Probable Causes	X'1023'	Communications program in remote node
	X'1022'	Communications program
User Causes	none	
Install Causes	X'8004'	Remote Node Definition Error
	X'8005'	System Definition Error
Actions	X'3110'	Contact communications systems programmer
	X'32A0'	Report the following
	X'82' SF	(Product Alert Reference Code) (note 1)
	X'F0F1'	Resource associated with the alert

		condition
	X'84' SF	(Associated resource)
Failure Causes	(none)	
Additional SVs	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
		First resource below sender:
		Name=(name of requesting CP)
		Type=X'F4' (CP)
		Second resource below sender:
		Name=(name of requesting LU)
		Type=X'F3' (LU)
	X'11' SF	Associated Resources
		Name=(name of intended destination of
		the BIND)
		Type=X'F3' (LU)

Notes:

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

   An alternative to this latter use of the Product Alert Reference Code could be to include the detecting module name.

A.5.8.3 Session Services Alert CPSS003

Alert Condition:

Protocol violation on a Locate or BIND request.

This Alert covers the following SNA sense data:

```
X'086C 2B00'    X'086C 2C00'    X'086D 4680'
X'1010 5002'    X'1014 5046'
```

Alert ID Number		X'21745F28'
Alert Type	X'01'	Permanent
Alert Description	X'3002'	Session Services Protocol Error
Probable Causes	X'1023'	Communications program in remote node
User Causes	(none)	
Install Causes	(none)	

Failure Causes	X'2209'	Session Services program in remote node
Actions	X'3000'	Contact appropriate service representative
	X'32C0'	Report the following:
	X'82' SF	(SNA sense data)
	X'82' SF	(Product Alert Reference Code) (note 1)
Additional SVs	X'48' SV	Supporting Data Correlation (note 2)
	X'60' SF	(Fully Qualified Session PCID)
	X'82' SF	(File Name)
	X'82' SF	(Log Record Number)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
	X'11' SF	Associated Resources
		Name=(adjacent node)
		Type=X'F4' (CP)

Notes:

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

   An alternative to this latter use of the Product Alert Reference Code could be to include the detecting module name.

2. The Supporting Data Correlation (X'48') SV is present to report that diagnostic data for this Alert was stored by the Alert sender.

A.6 LU6.2 Alerts

Subtopics:

- A.6.1 LU6.2 Alert LU62001
- A.6.2 LU6.2 Alert LU62002
- A.6.3 LU6.2 Alert LU62003
- A.6.4 LU6.2 Alert LU62004
- A.6.5 LU6.2 Alert LU62005

A.6.1 LU6.2 Alert LU62001

Alert Condition:

A partner LU that supports only nonextended BINDs detected an error causing it to send a -RSP(BIND). The LU receiving the -RSP(BIND) generates this Alert. The SNA sense data denotes the exact error condition detected.

Alert ID Number		X'FE1C42EB'

Alert Type	X'01'	Permanent
Alert Description	X'3115'	LU6.2 Received negative BIND response
Probable Causes	X'1023'	Communications program in remote node
	X'1022'	Communications program
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1023'	Communications program in remote node
	X'1022'	Communications program
Actions	X'3110'	Contact communications systems programmer
	X'32C0'	Report the following:
	X'82' SF	(Mode name) (Note 1)
	X'82' SF	(SNA sense data)
	X'2012'	(Review Associated Resources)
Additional SVs	X'47' SV	MSU Correlation
	X'60' SF	(Fully Qualified session PCID) (Note 2)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
	X'11' SF	Associated Resources
		Name=(First resource below sender, NQ Local LU name)
		Type=X'F3' (LU)
		Name=(Network-qualified partner LU name) (Note 3)
		Type=X'F3' (LU)

Notes:

1. Subfield contains the mode name associated with the deactivated session. If no mode name is available when the Alert is sent, the mode name is encoded in the subfield as eight X'40' bytes.

2. Subfield contains the NQPCID, if available (contains the session ID otherwise). The subfield contains a nonnull value when the Alert reports an error related to a session. If neither ID is available, the subfield is not sent.

3. Network-qualified Partner LU Name and Type fields are not sent if unavailable.

A.6.2 LU6.2 Alert LU62002

Alert Condition:

The LU has detected an error, causing it to send a -RSP(BIND) response to its partner LU. The SNA sense data denotes the exact error condition detected.

Alert ID Number		X'9DCD7CCA'
Alert Type	X'01'	Permanent
Alert Description	X'3116'	LU6.2 Sent negative BIND response
Probable Causes	X'1023'	Communications program in remote node
	X'1022'	Communications program
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1023'	Communications program in remote node
	X'1022'	Communications program
Actions	X'3110'	Contact communications systems programmer
	X'32C0'	Report the following:
	X'82' SF	(Mode name) (Note 1)
	X'82' SF	(SNA sense data)
	X'2012'	(Review Associated Resources)
Additional SVs	X'47' SV	MSU Correlation
	X'60' SF	(Fully Qualified session PCID) (Note 2)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
	X'11' SF	Associated Resources
		Name=(First resource below sender, NQ Local LU name)
		Type=X'F3' (LU)
		Name=(Network-qualified partner LU name) (Note 3)
		Type=X'F3' (LU)

Notes:

1. Subfield contains the mode name associated with the deactivated session. If no mode name is available when the Alert is sent, the mode name is encoded in the subfield as eight X'40' bytes.

2. Subfield contains the NQPCID, if available (contains the session ID otherwise). The subfield contains a nonnull value when the Alert reports an error related to a session. If neither ID is available, the subfield is not sent.

3. Network-qualified Partner LU Name and Type fields are not sent if unavailable.

A.6.3 LU6.2 Alert LU62003

Alert Condition:

The CP rejects the LU's request for assistance in the initiation of a session. SM informs RM that session activation has failed. The SNA sense data denotes the exact error condition being reported.

Alert ID Number		X'47302521'
Alert Type	X'01'	Permanent
Alert Description	X'3117'	LU6.2 Session activation rejected
Probable Causes	X'1022'	Communications program
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1022'	Communications program
Actions	X'3110'	Contact communications systems programmer
	X'32C0'	Report the following:
	X'82' SF	(Mode name) (Note 1)
	X'82' SF	(SNA sense data)
	X'2012'	(Review Associated Resources)
Additional SVs	X'47' SV	MSU Correlation
	X'60' SF	(Fully Qualified session PCID) (Note 2)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
	X'11' SF	Associated Resources
		Name=(First resource below sender, NQ Local LU name)
		Type=X'F3' (LU)
		Name=(Network-qualified partner LU name) (Note 3)
		Type=X'F3' (LU)

Notes:

1. Subfield contains the mode name associated with the deactivated session. If no mode name is available when the Alert is sent, the mode name is encoded in the subfield as eight X'40' bytes.

2. Subfield contains the NQPCID, if available (contains the session ID otherwise). The subfield contains a nonnull value when the Alert reports an error related to a session. If neither ID is available, the subfield is not sent.

3. Network-qualified Partner LU Name and Type fields are not sent if unavailable.

A.6.4 LU6.2 Alert LU62004

Alert Condition:

The LU has detected an error causing it to send an UNBIND to its partner LU. The SNA sense data denotes the exact error condition being reported.

Alert ID Number		X'A89646AA'
Alert Type	X'01'	Permanent
Alert Description	X'3118'	LU6.2 UNBIND request sent
Probable Causes	X'1023'	Communications program in remote node
	X'1022'	Communications program
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1023'	Communications program in remote node
	X'1022'	Communications program
Actions	X'3110'	Contact communications systems programmer
	X'32D0'	Report the following:
	X'82' SF	(Mode name) (Note 1)
	X'82' SF	(UNBIND type)
	X'82' SF	(SNA sense data)
	X'2012'	(Review Associated Resources)
Additional SVs	X'47' SV	MSU Correlation
	X'60' SF	(Fully Qualified session PCID) (Note 2)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
	X'11' SF	Associated Resources
		Name=(First resource below sender, NQ Local LU name)
		Type=X'F3' (LU)
		Name=(Network-qualified partner LU name) (Note 3)
		Type=X'F3' (LU)

Notes:

1. Subfield contains the mode name associated with the deactivated session. If no mode name is available when the Alert is sent, the mode name is encoded in the subfield as eight X'40' bytes.

2. Subfield contains the NQPCID, if available (contains the session ID otherwise). The subfield contains a nonnull value when the Alert

reports an error related to a session. If neither ID is available, the subfield is not sent.

3. Network-qualified Partner LU Name and Type fields are not sent if unavailable.

A.6.5 LU6.2 Alert LU62005

Alert Condition:

The partner LU (which supports extended BINDs) has rejected the BIND with an UNBIND. The SNA sense data denotes the exact error condition being reported.

Alert ID Number		X'7599A7D8'
Alert Type	X'01'	Permanent
Alert Description	X'3119'	LU6.2 UNBIND request received
Probable Causes	X'1023'	Communications program in remote node
	X'1022'	Communications program
User Causes	(none)	
Install Causes	(none)	
Failure Causes	X'1023'	Communications program in remote node
	X'1022'	Communications program
Actions	X'3110'	Contact communications systems programmer
	X'32D0'	Report the following:
	X'82' SF	(Mode name) (Note 1)
	X'82' SF	(UNBIND type)
	X'82' SF	(SNA sense data)
	X'2012'	(Review Associated Resources)
Additional SVs	X'47' SV	MSU Correlation
	X'60' SF	(Fully Qualified session PCID) (Note 2)
	X'05' SV	Hierarchy/Resource List
	X'10' SF	Hierarchy Name List
		Sending node:
		Name=(CP name)
		Type=X'F4' (CP)
	X'11' SF	Associated Resources
		Name=(First resource below sender, NQ Local LU name)
		Type=X'F3' (LU)
		Name=(Network-qualified partner LU name) (Note 3)
		Type=X'F3' (LU)

Notes:

1. Subfield contains the mode name associated with the deactivated session. If no mode name is available when the Alert is sent, the mode name is encoded in the subfield as eight X'40' bytes.

2. Subfield contains the NQPCID, if available (contains the session ID otherwise). The subfield contains a nonnull value when the Alert reports an error related to a session. If neither ID is available, the subfield is not sent.

3. Network-qualified Partner LU Name and Type fields are not sent if unavailable.

B.0 Appendix B. Management Services Protocol Boundary Verbs

Subtopics:

- B.1 Introduction
- B.2 Verb Description Table
- B.3 Parameter Description
- B.4 Protocol Boundary Verbs for File Services
- B.5 Protocol Boundary Verbs for Change Management
- B.6 Subtables
- B.7 Parameter descriptions

B.1 Introduction

The term *protocol boundary* is used generally to refer to the semantic definition of the data and control exchanges between two components in an SNA node. This appendix focuses on the protocol boundaries between MS and the network planners that interact with SNA/MS to request services.

These protocol boundaries are described generically here in the form of precisely defined verbs.

No one-to-one correspondence exists between a protocol boundary verb and a related SNA format description. The verbs in this appendix should not be misinterpreted as being a representation of the actual formats for SNA/MS requests and replies. *SNA Formats* contains the architecture of record for all SNA encodings.

The MS Protocol Boundary Verbs are formally described in verb description tables and parameter descriptions. A Verb Description Table contains the primary syntax for each parameter associated with a particular verb. A Parameter Description contains a prose description of the parameter, enumeration values, and any conditions of presence associated with a particular parameter.

B.2 Verb Description Table

Subtopics:

- B.2.1 Column Descriptions

B.2.1 Column Descriptions

Subtopics:

- B.2.1.1 Supplied Parameter Name
- B.2.1.2 Returned Parameter Name
- B.2.1.3 Parameter Reference Page (Parm Ref Page)
- B.2.1.4 Length
- B.2.1.5 Occurrences
- B.2.1.6 Children

B.2.1.1 Supplied Parameter Name

The first column of the verb description table identifies the parameters supplied on the invocation of a particular verb, and illustrates their hierarchical relationship by indentation of the column entries.

B.2.1.2 Returned Parameter Name

The Returned Parameter Name identifies the parameters returned as a result of the invocation of a particular verb, and illustrates their hierarchical relationship by indentation of the column entries.

B.2.1.3 Parameter Reference Page (Parm Ref Page)

As primary syntax is described in a particular verb desciption table, the semantics, enumeration values, and other characteristics are described formally in the parameter description. This column contains a reference page number to where this parameter information is found.

B.2.1.4 Length

The range of length values specifies the minimum and maximum lengths of parameters which an implementation is required to accept across the protocol boundary. Sometimes the length is described as an enumeration, which may be implemented as an integer, character string, pointer, or any implementation choice.

B.2.1.5 Occurrences

Multiple occurrences of parameters may or may not be permitted. A value of "1 - <some number>" in this column indicates the allowed range of occurrences of the corresponding parameter. A value of "°1" indicates that there is no architecturally-defined maximum. A value of "1" in this column indicates that only a single instance of the corresponding parameter is appropriate. A value of "0 - 1" indicates that an instance of the corresponding parameter is optional.

Note: An asterisk denotes special presence rules for a particular parameter. These presence rules are detailed in the corresponding parameter description.

B.2.1.6 Children

Number (Num): Each parent parameter contains a certain number of different children. This column specifies the minimum and maximum number of different children for a particular parent parameter. This column also specifies mutual exclusion among a set of optional children. If all children are optional ("0-1") and the parent parameter contains "1" for children number, one of the set of children occurs within that particular parent. This column does not account for multiple occurrences of a particular child within the parent parameter. Multiple occurrences of a particular parameter are indicated in the "Occurrences" column.

Subtable (Subtab): Sometimes the need to divide large tables into subtables becomes apparent, particularly when common parameters appear frequently within different parameter description tables. This column contains a reference page number to the page on which these common parameters are described.

B.3 Parameter Description

This description is referenced by a page number appearing in the "Parameter Reference Page" column corresponding to each parameter in the

verb description table. The parameter description contains information pertaining to a particular parameter. Prose descriptions, presence rules, enumeration values and semantics associated with the corresponding entry in the verb description table may appear in the parameter description.

B.4 Protocol Boundary Verbs for File Services

Subtopics:

- B.4.1 VERB: Retrieve
- B.4.2 VERB: Send
- B.4.3 VERB: Delete
- B.4.4 VERB: Reply_To_Retrieve
- B.4.5 VERB: Reply_To_Send
- B.4.6 VERB: Reply_To_Delete
- B.4.7 VERB: Notification_Of_Arrival

B.4.1 VERB: Retrieve

Retrieve is issued by the network planner to request the focal point (or by a local operator to request the entry point) to, in turn, request another focal point or an entry point to transfer a file to the requester.

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Supplied Parameter Name					
Correlator	B.7	4	1	__	__
Object_Class	B.7	4	0-1	__	__
To_Be_Fetched_Name	B.7	1-61	1	__	__
Fetching_Match_Flags	B.7	ENUM	0-1	__	__
Source_Location	__	__	1	__	B.6.11
DS_Priority	B.7	ENUM	0-1	__	__
DS_Security	__	__	0-1	__	B.6.6
Returned Parameter Name					
Return_Code	B.7	ENUM	1	__	__

B.4.2 VERB: Send

Send is issued by the network planner to request the focal point (or by a local operator to request the entry point) to send a file to one or more focal points or entry points.

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Supplied Parameter Name					
Correlator	B.7	4	1	__	__
Object_Class	B.7	4	0-1	__	__
To_Be_Fetched_Name	B.7	1-61	1	__	__
Fetching_Match_Flags	B.7	ENUM	0-1	__	__
To_Be_Deleted_Name	B.7	1-61	0-1	__	__
Deleting_Match_Flags	B.7	ENUM	0-1	__	__
Destruction	B.7	ENUM	1	__	__
Target_List	__	__	1	__	B.6.12

DS_Priority	B.7	ENUM	0-1	__	__
DS_Security	__	__	0-1	__	B.6.6
Returned Parameter Name					
Return_Code	B.7	ENUM	1	__	__

B.4.3 VERB: Delete

Delete is issued by the network planner to request the focal point (or by a local operator to request the entry point) to delete a file at one or more focal points or entry points.

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Supplied Parameter Name					
Correlator	B.7	4	1	__	__
Object_Class	B.7	4	0-1	__	__
To_Be_Deleted_Name	B.7	1-61	1	__	__
Deleting_Match_Flags	B.7	ENUM	0-1	__	__
Target_List	__	__	1	__	B.6.12
DS_Priority	B.7	ENUM	0-1	__	__
DS_Security	__	__	0-1	__	B.6.6
Returned Parameter Name					
Return_Code	B.7	ENUM	1	__	__

B.4.4 VERB: Reply_To_Retrieve

Reply_To_Retrieve is issued by the focal point to report to the network planner (or by the entry point to report to the local operator) the results of a Retrieve request.

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Supplied Parameter Name					
Correlator	B.7	4	1	__	__
DS_Security	__	__	0-1	__	B.6.6
Object_Class	B.7	4	0-1	__	__
Fetched_Name	B.7	1-61	1	__	__
Source_Location	__	__	1	__	B.6.11
Time_Stamp	__	__	1	__	B.6.14
FS_Action_Summary	B.7	__	1	__	__
SNA_Condition_Report	B.7	__	°0	__	__
Returned Parameter Name					
Return_Code	B.7	ENUM	1	__	__

B.4.5 VERB: Reply_To_Send

Reply_To_Send is issued by the focal point to report to the network

planner (or by the entry point to report to the local operator) the results of a Send request.

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Supplied Parameter Name					
Correlator	B.7	4	1	__	__
DS_Security	__	__	0-1	__	B.6.6
Object_Class	B.7	4	0-1	__	__
Stored_Name	B.7	1-61	1	__	__
Deleted_Name	B.7	1-61	0-1	__	__
Target_Location	__	__	1	__	B.6.13
Time_Stamp	__	__	1	__	B.6.14
FS_Action_Summary	B.7	__	1	__	__
SNA_Condition_Report	B.7	__	°0	__	__
Returned Parameter Name					
Return_Code	B.7	ENUM	1	__	__

B.4.6 VERB: Reply_To_Delete

Reply_To_Delete is issued by the focal point to report to the network planner (or by the entry point to report to the local operator) the results of a Delete request.

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Supplied Parameter Name					
Correlator	B.7	4	1	__	__
DS_Security	__	__	0-1	__	B.6.6
Object_Class	B.7	4	0-1	__	__
Deleted_Name	B.7	1-61	1	__	__
Target_Location	__	__	1	__	B.6.13
Time_Stamp	__	__	1	__	B.6.14
FS_Action_Summary	B.7	__	1	__	__
SNA_Condition_Report	B.7	__	°0	__	__
Returned Parameter Name					
Return_Code	B.7	ENUM	1	__	__

B.4.7 VERB: Notification_Of_Arrival

Notification_Of_Arrival is issued by the focal point to report to the network planner (or by the entry point to report to the local operator) that a file has arrived in an unsolicited manner from another entry point or focal point.

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Supplied Parameter Name					
Correlator	B.7	4	1	__	__

DS_Security	__	__	0-1	__	B.6.6
Object_Class	B.7	4	0-1	__	__
Stored_Name	B.7	1-61	1	__	__
Source_Location	__	__	1	__	B.6.11
Time_Stamp	__	__	1	__	B.6.14
FS_Action_Summary	B.7	__	1	__	__
SNA_Condition_Report	B.7	__	°0	__	__
Returned Parameter Name					
Return_Code	B.7	ENUM	1	__	__

B.5 Protocol Boundary Verbs for Change Management

Subtopics:

- B.5.1 VERB: Send_and_Install
- B.5.2 VERB: Send_and_Initiate
- B.5.3 VERB: Install
- B.5.4 VERB: Initiate
- B.5.5 VERB: Remove
- B.5.6 VERB: Accept
- B.5.7 VERB: Activate
- B.5.8 VERB: Reporting_Installation
- B.5.9 VERB: Reporting_Initiation
- B.5.10 VERB: Reporting_Removal
- B.5.11 VERB: Reporting_Acceptance
- B.5.12 VERB: Activation_Acceptance
- B.5.13 VERB: Reporting_Activation

B.5.1 VERB: Send_and_Install

Send_and_Install is issued by the network planner to request the focal point to send a change file to one or more entry points and to install it there.

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Supplied Parameter Name					
Correlator	B.7	4	1	__	__
To_Be_Fetched_Name	B.7	1-61	1	__	__
Fetching_Match_Flags	B.7	ENUM	0-1	__	__
To_Be_Deleted_Name	B.7	1-61	0-1	__	__
Deleting_Match_Flags	B.7	ENUM	0-1	__	__
Destruction	B.7	ENUM	1	__	__
Target_List	__	__	1	__	B.6.12
DS_Priority	B.7	ENUM	0-1	__	__
DS_Security	__	__	0-1	__	B.6.6
Execution_Window_Tim__g		__	0-1	__	B.6.7
Corequisite_Change_N__e_List		__	0-1	__	B.6.4
Removability	B.7	ENUM	1	__	__
Automatic_Removal	B.7	ENUM	0-1	__	__
Pre-Test	B.7	ENUM	1	__	__
Post-Test	B.7	ENUM	1	__	__
Automatic_Acceptance__		__	0-1	__	B.6.1
Activation_Use	B.7	ENUM	1	__	__

Alter_Active_Compo	eB.7	ENUM	0-1	__	__
Object_Disposition	B.7	ENUM	0-1	__	__
Object_Class	B.7	4	0-1	__	__
Returned Parameter Name					
Return_Code	B.7	ENUM	1	__	__

B.5.2 VERB: Send_and_Initiate

Send_and_Initiate is issued by the network planner to request the focal point to send a command procedure to one or more entry points and to initiate it there.

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Supplied Parameter Name					
Correlator	B.7	4	1	__	__
To_Be_Fetched_Name	B.7	1-61	1	__	__
Fetching_Match_Fla	sB.7	ENUM	0-1	__	__
To_Be_Deleted_Name	B.7	1-61	0-1	__	__
Deleting_Match_Fla	sB.7	ENUM	0-1	__	__
Destruction	B.7	ENUM	1	__	__
Target_List	__	__	1	__	B.6.12
DS_Priority	B.7	ENUM	0-1	__	__
DS_Security	__	__	0-1	__	B.6.6
Execution_Window_T	m__g	__	0-1	__	B.6.7
Submission_Type	B.7	ENUM	1	__	__
Object_Number	B.7	2	1	__	__
Object_Disposition	B.7	ENUM	0-1	__	__
Object_Class	B.7	4	0-1	__	__
Returned Parameter Name					
Return_Code	B.7	ENUM	1	__	__

B.5.3 VERB: Install

Install is issued by the network planner to request the focal point to install a change file at one or more entry points. It may also be issued by a local operator to install a change file at the local node.

Since the entry point remembers the list of corequisite change file names (as part of maintaining removability) the network planner need specify only one of the names in the Stored_Name parameter when installing into production corequisite changes that had previously been installed on trial.

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Supplied Parameter Name					
Correlator	B.7	4	1	__	__
Stored_Name	B.7	1-61	1	__	__
Target_List	__	__	1	__	B.6.12

DS_Priority	B.7	ENUM	0-1	__	__
DS_Security	__	__	0-1	__	B.6.6
Execution_Window_T	m__g	__	0-1	__	B.6.7
Corequisite_Change	N__e_List	__	0-1	__	B.6.4
Removability	B.7	ENUM	1	__	__
Automatic_Removal	B.7	ENUM	0-1	__	__
Pre-Test	B.7	ENUM	1	__	__
Post-Test	B.7	ENUM	1	__	__
Automatic_Acceptan	e__	__	0-1	__	B.6.1
Activation_Use	B.7	ENUM	1	__	__
Alter_Active_Compo	eB.7	ENUM	0-1	__	__
Object_Disposition	B.7	ENUM	0-1	__	__
Object_Class	B.7	4	0-1	__	__
Returned Parameter Name					
Return_Code	B.7	ENUM	1	__	__

B.5.4 VERB: Initiate

Initiate is issued by the network planner to request the focal point to initiate a command procedure at one or more entry points. It may also be issued by a local operator to initiate a command procedure at the local node.

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Supplied Parameter Name					
Correlator	B.7	4	1	__	__
Stored_Name	B.7	1-61	1	__	__
Target_List	__	__	1	__	B.6.12
DS_Priority	B.7	ENUM	0-1	__	__
DS_Security	__	__	0-1	__	B.6.6
Execution_Window_T	m__g	__	0-1	__	B.6.7
Submission_Type	B.7	ENUM	1	__	__
Object_Number	B.7	2	1	__	__
Object_Disposition	B.7	ENUM	0-1	__	__
Object_Class	B.7	4	0-1	__	__
Returned Parameter Name					
Return_Code	B.7	ENUM	1	__	__

B.5.5 VERB: Remove

Remove is issued by the network planner to request the focal point to remove a change described in a change file from one or more entry points, and if successful, to release the change file. It may also be issued by a local operator to remove a change described in a change file from the local node, and if successful, to release the change file.

Removing a change means returning all components previously altered in connection with a change to their condition prior to the installation of the change.

Since the entry point remembers the list of corequisite change file names

(as part of maintaining removability) the network planner need specify only one of the names in the Stored_Name parameter.

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Supplied Parameter Name					
Correlator	B.7	4	1	__	__
Stored_Name	B.7	1-61	1	__	__
Target_List	__	__	1	__	B.6.12
DS_Priority	B.7	ENUM	0-1	__	__
DS_Security	__	__	0-1	__	B.6.6
Execution_Window_T	m__g	__	0-1	__	B.6.7
Post-Test	B.7	ENUM	1	__	__
Alter_Active_Compo	eB.7	ENUM	0-1	__	__
Object_Class	B.7	4	0-1	__	__
Returned Parameter Name					
Return_Code	B.7	ENUM	1	__	__

B.5.6 VERB: Accept

Accept is issued by the network planner to request the focal point to cause one or more entry points to release resources required to maintain the removability of a change described in a change file. It may also be issued by a local operator to cause the local node to release resources required to maintain removability of a change described in a change file. The resources released are, typically, unaltered versions of components affected by the change.

Since the entry point remembers the list of corequisite change file names (as part of maintaining removability) the network planner or local operator need specify only one of the names in the Stored_Name parameter.

The changes must be installed in production removably; a request to accept changes installed on trial will be rejected.

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Supplied Parameter Name					
Correlator	B.7	4	1	__	__
Stored_Name	B.7	1-61	1	__	__
Target_List	__	__	1	__	B.6.12
DS_Priority	B.7	ENUM	0-1	__	__
DS_Security	__	__	0-1	__	B.6.6
Execution_Window_T	m__g	__	0-1	__	B.6.7
Object_Class	B.7	4	0-1	__	__
Returned Parameter Name					
Return_Code	B.7	ENUM	1	__	__

B.5.7 VERB: Activate

Activate is issued by the network planner to request the focal point to send a command to one or more entry points to cause reactivation of the entire entry point. It may also be issued by a local operator to cause reactivation of the local node.

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Supplied Parameter Name					
Correlator	B.7	4	1	__	__
Target_List	__	__	1	__	B.6.12
DS_Security	__	__	0-1	__	B.6.6
Execution_Window_T	m__g	__	0-1	__	B.6.7
Force	__	__	1	__	B.6.8
Change_Management_	cB.7ation	UENUM	0-1	__	__
Returned Parameter Name					
Return_Code	B.7	ENUM	1	__	__

B.5.8 VERB: Reporting_Installation

Reporting_Installation is issued by the focal point to report to the network planner that one or more change files were (or were not) successfully installed at an entry point. This report is unsolicited in the case where a successful installation was the result of a command issued by a local operator at the entry point.

The storing of the change file may also be reported.

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Supplied Parameter Name					
Correlator	B.7	4	1	__	__
DS_Security	__	__	0-1	__	B.6.6
Stored_Name	B.7	1-61	0-1	__	__
Deleted_Name	B.7	1-61	0-1	__	__
Target_Location	__	__	1	__	B.6.13
Time_Stamp	__	__	1	__	B.6.14
FS_Action_Summary	B.7	__	0-1	__	__
SNA_Condition_Repo	tB.7	__	°0	__	__
Detailed_Data	B.7	__	°0	__	__
Installation_Resul	sB.7	__	1	3	__
Installation_Sta	uB.7	ENUM	1	__	__
When_Effective	B.7	ENUM	1	__	__
Reported_Change_	a___List	__	1	__	B.6.9
Pre-Test_Status	B.7	ENUM	1	__	__
Post-Test_Status	B.7	ENUM	1	__	__
Automatic_Removal_	eB.7ts	__	1	2	__
Automatic_Remova	B.7tus	ENUM	1	__	__
When_Effective	B.7	ENUM	1	__	__
Automatic_Acceptan	eB.7atus	ENUM	1	__	__
Removability_Statu	B.7	ENUM	1	__	__
Activation_Use_Sta	uB.7	ENUM	1	__	__
Back-Level_Change_	a___List	__	0-1	__	B.6.2
Deleted_Change_Nam	___st	__	0-1	__	B.6.5
Object_Class	B.7	4	0-1	__	__

Returned Parameter Name					
Return_Code	B.7	ENUM	1	__	__

B.5.9 VERB: Reporting_Initiation

Reporting_Initiation is issued by the focal point to report to the network planner whether or not a command procedure was successfully initiated at an entry point.

The storing of the command procedure may also be reported.

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Supplied Parameter Name					
Correlator	B.7	4	1	__	__
DS_Security	__	__	0-1	__	B.6.6
Stored_Name	B.7	1-61	0-1	__	__
Deleted_Name	B.7	1-61	0-1	__	__
Target_Location	__	__	1	__	B.6.13
Time_Stamp	__	__	1	__	B.6.14
FS_Action_Summary	B.7	__	0-1	__	__
SNA_Condition_Report	B.7	__	°0	__	__
Detailed_Data	B.7	__	°0	__	__
Initiation_Results	B.7	__	1	2	__
Job_Element_Stat	B.7	ENUM	1	__	__
Completion_Code	B.7	2	1	__	__
Object_Class	B.7	4	0-1	__	__
Returned Parameter Name					
Return_Code	B.7	ENUM	1	__	__

B.5.10 VERB: Reporting_Removal

Reporting_Removal is issued by the focal point to report to the network planner that one or more change files were (or were not) successfully removed at an entry point. This report is unsolicited in the case where a successful removal was the result of a command issued by a local operator at the entry point.

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Supplied Parameter Name					
Correlator	B.7	4	1	__	__
DS_Security	__	__	0-1	__	B.6.6
Target_Location	__	__	1	__	B.6.13
Time_Stamp	__	__	1	__	B.6.14
SNA_Condition_Report	B.7	__	°0	__	__
Detailed_Data	B.7	__	°0	__	__
Removal_Results	B.7	__	1	3	__
Removal_Status	B.7	ENUM	1	__	__
When_Effective	B.7	ENUM	1	__	__

Reported_Change_	a___List	__	1	__	B.6.9
Post-Test_Status	B.7	ENUM	1	__	__
Secondary_Installa	iB.7Resul	s__	0-1	4	__
Secondary_Instal	aB.7n_Sta	uENUM	1	__	__
When_Secondary_E	fB.7ive	ENUM	1	__	__
Secondary_Activa	iB.7Use_S	aENUM	1	__	__
Secondary_Instal	a__on_Cha	g__Name_Li	t 1	__	B.6.10
Object_Class	B.7	4	0-1	__	__
Returned Parameter Name					
Return_Code	B.7	ENUM	1	__	__

B.5.11 VERB: Reporting_Acceptance

Reporting_Acceptance is issued by the focal point to report to the network planner that one or more change files were (or were not) successfully accepted at an entry point. This report is unsolicited in the case where a successful acceptance was the result of a command issued by a local operator at the entry point.

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Supplied Parameter Name					
Correlator	B.7	4	1	__	__
DS_Security	__	__	0-1	__	B.6.6
Target_Location	__	__	1	__	B.6.13
Time_Stamp	__	__	1	__	B.6.14
SNA_Condition_Repo	tB.7	__	°0	__	__
Detailed_Data	B.7	__	°0	__	__
Acceptance_Results	B.7	__	1	2	__
Acceptance_Statu	B.7	ENUM	1	__	__
Reported_Change_	a___List	__	1	__	B.6.9
Deleted_Change_Nam	___st	__	0-1	__	B.6.5
Object_Class	B.7	4	0-1	__	__
Returned Parameter Name					
Return_Code	B.7	ENUM	1	__	__

B.5.12 VERB: Activation_Acceptance

Activation_Acceptance is issued by the focal point to report to the network planner that an Activate request will (or will not) be attempted by the entry point.

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Supplied Parameter Name					
Correlator	B.7	4	1	__	__
DS_Security	__	__	0-1	__	B.6.6
Target_Location	__	__	1	__	B.6.13
Time_Stamp	__	__	1	__	B.6.14

SNA_Condition_Report	B.7	__	°0	__	__
Detailed_Data	B.7	__	°0	__	__
Activation_Acceptance_Status	B.7	ENUM	1	__	__
Additional_Reports	B.7	ENUM	0-1	__	__
Returned Parameter Name					
Return_Code	B.7	ENUM	1	__	__

B.5.13 VERB: Reporting_Activation

Reporting_Activation is issued by the focal point to report to the network planner that the entry point was (or was not) successfully activated. This report is unsolicited in the case where a successful activation was the result of a command issued by a local operator at the entry point.

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Supplied Parameter Name					
Correlator	B.7	4	1	__	__
DS_Security	__	__	0-1	__	B.6.6
Target_Location	__	__	1	__	B.6.13
Time_Stamp	__	__	1	__	B.6.14
SNA_Condition_Report	B.7	__	°0	__	__
Detailed_Data	B.7	__	°0	__	__
Activation_Status	B.7	ENUM	1	__	__
Activation_Stage	B.7	ENUM	0-1	__	__
Change_Management_Activation_Status	B.7	ENUM	0-1	__	B.6.3
Returned Parameter Name					
Return_Code	B.7	ENUM	1	__	__

B.6 Subtables

Subtopics:

- B.6.1 SUBTABLE: Automatic_Acceptance
- B.6.2 SUBTABLE: Back-Level_Change_Name_List
- B.6.3 SUBTABLE: Change_Management_Activation_Use_Status
- B.6.4 SUBTABLE: Corequisite_Change_Name_List
- B.6.5 SUBTABLE: Deleted_Change_Name_List
- B.6.6 SUBTABLE: DS_Security
- B.6.7 SUBTABLE: Execution_Window_Timing
- B.6.8 SUBTABLE: Force
- B.6.9 SUBTABLE: Reported_Change_Name_List
- B.6.10 SUBTABLE: Secondary_Installation_Change_Name_List
- B.6.11 SUBTABLE: Source_Location
- B.6.12 SUBTABLE: Target_List
- B.6.13 SUBTABLE: Target_Location
- B.6.14 SUBTABLE: Time_Stamp

B.6.1 SUBTABLE: Automatic_Acceptance

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab

Automatic_Acceptan	eB.7	__	__	1-2	__
Automatic_Accept	nB.7Reque	tENUM	1	__	__
Automatic_Accept	nB.7Delay	1	0-1	__	__

B.6.2 SUBTABLE: Back-Level_Change_Name_List

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Back-Level_Change_	aB.7List	__	__	1	__
Change_File_Name	B.7	1-61	°1	__	__

B.6.3 SUBTABLE: Change_Management_Activation_Use_Status

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Change_Management_	cB.7ation	U___Status	__	1-2	__
C_M_Activation_U	eB.7atus_	eENUM	1	__	__
C_M_Activation_U	eB.7atus_	oENUMion	0-1	__	__

B.6.4 SUBTABLE: Corequisite_Change_Name_List

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Corequisite_Change	NB.7_List	__	__	1	__
Change_File_Name	B.7	1-61	1-6	__	__

B.6.5 SUBTABLE: Deleted_Change_Name_List

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Deleted_Change_Nam	_B.7t	__	__	1	__
Change_File_Name	B.7	1-61	°1	__	__

B.6.6 SUBTABLE: DS_Security

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
DS_Security	B.7	__	__	2	__
DS_Security_Requ	sB.7	ENUM	1	__	__
DS_Security_Comp	OB.7	ENUM	1	__	__

B.6.7 SUBTABLE: Execution_Window_Timing

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Execution_Window_T	mB.7	__	__	1	__
Execute_Not_Befo	eB.7me	__	1	6	__
Execute_Not_Be	oB.7Year	2	1	__	__
Execute_Not_Be	oB.7Month	1	1	__	__
Execute_Not_Be	oB.7Day	1	1	__	__
Execute_Not_Be	oB.7Hour	1	1	__	__
Execute_Not_Be	oB.7Minut	1	1	__	__
Execute_Not_Be	oB.7Local	OENUMT	1	__	__

B.6.8 SUBTABLE: Force

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Force	B.7	__	__	1	__
Force_Request	B.7	ENUM	1	__	__

B.6.9 SUBTABLE: Reported_Change_Name_List

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Reported_Change_Na	eB.7st	__	__	1	__
Change_File_Name	B.7	1-61	°1	__	__

B.6.10 SUBTABLE: Secondary_Installation_Change_Name_List

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Secondary_Installa	iB.7Chang	___me_List	__	1	__
Change_File_Name	B.7	1-61	°1	__	__

B.6.11 SUBTABLE: Source_Location

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Source_Location	B.7	__	__	2	__
NETID	B.7	1-8	1	__	__
NAU_Name	B.7	1-8	1	__	__

B.6.12 SUBTABLE: Target_List

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Target_List	B.7	__	__	°1	__
Target_Location	__	__	°1	__	B.6.13

B.6.13 SUBTABLE: Target_Location

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Target_Location	B.7	__	__	2	__
NETID	B.7	1-8	1	__	__
NAU_Name	B.7	1-8	1	__	__

B.6.14 SUBTABLE: Time_Stamp

Parameter Name	Parm Ref Page	Length	Occurrences	Children	
				Num	Subtab
Time_Stamp	B.7	__	__	6-7	__
Time_Stamp_Year	B.7	1	1	__	__
Time_Stamp_Month	B.7	1	1	__	__
Time_Stamp_Day	B.7	1	1	__	__
Time_Stamp_Hour	B.7	1	1	__	__
Time_Stamp_Minut	B.7	1	1	__	__
Time_Stamp_Secon	B.7	1	1	__	__
Time_Stamp_Secon	B.7dredt	s1	0-1	__	__

B.7 Parameter descriptions

```
___ Correlator ____________________________________________________
|                                                                 |
|                                                                 |
| Description:  A 4-byte binary sequence number assigned by the   |
|               originator of the unit of work. A receiver of a request |
|               echoes it in the reply.                           |
|                                                                 |
| Verbs Supplied on: Retrieve, Send, Send_and_Install,            |
|               Send_and_Initiate, Install, Initiate, Remove, Accept, |
|               Activate, Reply_to_Retrieve, Reply_to_Send,       |
|               Notification_of_Arrival, Reporting_Installation,  |
|               Reporting_Initiation, Reporting_Removal,          |
|               Reporting_Acceptance, Activation_Acceptance,      |
|               Reporting_Activation                              |
|                                                                 |
| Verbs Returned on: None                                         |
|                                                                 |
| Subtables Found in: None                                        |
|                                                                 |
| Format:       Undefined byte string                             |
|                                                                 |
|_________________________________________________________________|
```

```
___ To_Be_Fetched_Name ____________________________________________
|                                                                 |
|                                                                 |
| Description:  The FS file name used by the FS server to identify the |
```

```
|               file to be fetched.                                      |
|                                                                        |
| Verbs Supplied on: Retrieve, Send, Send_and_Install, Send_and_Initiate |
|                                                                        |
| Verbs Returned on: None                                                |
|                                                                        |
| Subtables Found in: None                                               |
|                                                                        |
| Format:       Character string                                         |
|                                                                        |
|________________________________________________________________________|
```

CGCSGID: 01134-00500

String Conventions: Leading, imbedded, or trailing space characters (X'40') are not allowed.

```
 ___ Stored_Name ____________________________________________________
|                                                                    |
|                                                                    |
| Description:  In the case of a request, the FS file name used by the |
|               entry point to identify the file that pertains to the |
|               command in the DS agent object.  In the case of a     |
|               report, the file name that was stored by the entry    |
|               point's FS server.                                    |
|                                                                    |
|               This parameter is present on Reporting_Installation and |
|               Reporting_Initiation if the FS action to store the   |
|               object is also being reported, and the entry point has |
|               not deleted the object.                               |
|                                                                    |
| Verbs Supplied on: Install, Initiate, Remove, Accept, Reply_to_Send, |
|               Reporting_Installation, Reporting_Initiation          |
|                                                                    |
| Verbs Returned on: None                                            |
|                                                                    |
| Subtables Found in: None                                           |
|                                                                    |
| Format:       Character string                                     |
|                                                                    |
|____________________________________________________________________|
```

CGCSGID: 01134-00500

String Conventions: Leading, imbedded, or trailing space characters (X'40') are not allowed.

```
 ___ Fetching_Match_Flags ___________________________________________
|                                                                    |
|                                                                    |
| Description:  Matching flags to be provided to the FS server on the DS |
|               protocol boundary. See the FS parameter of the same name |
|               for specification details.                            |
|                                                                    |
| Verbs Supplied on: Retrieve, Send, Send_and_Install, Send_and_Initiate |
|                                                                    |
| Verbs Returned on: None                                            |
|                                                                    |
| Subtables Found in: None                                           |
|                                                                    |
| Format:       Enumeration                                          |
|                                                                    |
|____________________________________________________________________|

 ___ Destruction ____________________________________________________
|                                                                    |
|                                                                    |
| Description:  Specifies whether or not one or more files can be    |
|               destroyed (overwritten) as part of the FS action     |
|               requested.                                            |
|                                                                    |
| Verbs Supplied on: Send, Send_and_Install, Send_and_Initiate       |
|                                                                    |
| Verbs Returned on: None                                            |
|                                                                    |
| Subtables Found in: None                                           |
|                                                                    |
| Format:       Enumeration                                          |
|                                                                    |
|____________________________________________________________________|
```

Possible Values

ALLOWED
NO

___ Deleting_Match_Flags ___

```
Description:  Deleting flags to be provided to the FS server. See the
              FS parameter of the same name for specification details.

              On Send, Send_and_Install and Send_and_Initiate, this
              parameter cannot be specified if Destruction (NO) is
              specified. If Destruction (ALLOWED) is specified and
              this parameter is omitted, then complete matching is
              implied.

Verbs Supplied on:  Send, Send_and_Install, Send_and_Initiate

Verbs Returned on: None

Subtables Found in: None

Format:       Enumeration
```

___ To_Be_Deleted_Name ___

```
Description:  The FS file name to be used by the FS server to identify
              the file to be deleted.

              On Send, Send_and_Install and Send_and_Initiate, this
              parameter cannot be specified if Destruction (NO) is
              specified. If Destruction (ALLOWED) is specified and
              this parameter is omitted, then the To_Be_Fetched_Name
              is used for matching.

Verbs Supplied on:  Send, Send_and_Install, Send_and_Initiate

Verbs Returned on: None

Subtables Found in: None

Format:       Character string
```

```
CGCSGID:       01134-00500

String Conventions: Leading, imbedded, or trailing space characters
               (X'40') are not allowed.
```

___ Fetched_Name ___

```
Description:  The FS file name of the file that was fetched.

Verbs Supplied on: Reply_To_Retrieve

Verbs Returned on: None

Subtables Found in: None

Format:       Character string
```

```
CGCSGID:       01134-00500

String Conventions: Leading, imbedded, or trailing space characters
               (X'40') are not allowed.
```

___ Deleted_Name ___

```
Description:  The FS file name of the file that was deleted.

              This parameter is present on Reply_to_Send,
              Reporting_Installation and Reporting_Initiation if a
              file was deleted as part of the FS action.

Verbs Supplied on: Reply_To_Send, Reporting_Installation,
              Reporting_Initiation

Verbs Returned on: None

Subtables Found in: None

Format:       Character string
```

```
CGCSGID:       01134-00500


String Conventions: Leading, imbedded, or trailing space characters
               (X'40') are not allowed.


 ___ Source_Location ____________________________________________________
|                                                                        |
|                                                                        |
| Description:  The source location of a received file, consisting of    |
|               the (NETID, NAU_Name) pair.                              |
|                                                                        |
| Verbs Supplied on: Retrieve, Reply_To_Retrieve,                        |
|               Notification_Of_Arrival                                  |
|                                                                        |
| Verbs Returned on: None                                                |
|                                                                        |
| Subtables Found in: None                                               |
|                                                                        |
|________________________________________________________________________|




 ___ NETID ______________________________________________________________
|                                                                        |
|                                                                        |
| Description:  The network ID portion of the location name.             |
|                                                                        |
| Verbs Supplied on: Retrieve, Send, Send_and_Install,                   |
|               Send_and_Initiate, Install, Initiate, Remove, Accept,    |
|               Activate, Reply_to_Retrieve, Reply_to_Send,              |
|               Notification_of_Arrival, Reporting_Installation,         |
|               Reporting_Initiation, Reporting_Removal,                 |
|               Reporting_Acceptance, Activation_Acceptance,             |
|               Reporting_Activation                                     |
|                                                                        |
| Verbs Returned on: None                                                |
|                                                                        |
| Subtables Found in: Source_Location, Target_Location                   |
|                                                                        |
| Format:       Character string                                         |
|                                                                        |
|________________________________________________________________________|


CGCSGID:       01134-00500


String Conventions: Leading, imbedded, or trailing space characters
               (X'40') are not allowed.
```

Note: In existing networks where network IDs are defined using SNA character set A (includes character set AR, plus the special characters @, #, and $), the network IDs may contain any of the three special characters; however, these characters may not be available on keyboards in every country and should not be used in new LU names.

```
 ___ NAU_Name ___________________________________________________________
|                                                                        |
|                                                                        |
| Description:  The network-addressable unit name (e.g. logical unit     |
|               name) portion of the location name.                      |
|                                                                        |
| Verbs Supplied on: Retrieve, Send, Send_and_Install,                   |
|               Send_and_Initiate, Install, Initiate, Remove, Accept,    |
|               Activate, Reply_to_Retrieve, Reply_to_Send,              |
|               Notification_of_Arrival, Reporting_Installation,         |
|               Reporting_Initiation, Reporting_Removal,                 |
|               Reporting_Acceptance, Activation_Acceptance,             |
|               Reporting_Activation                                     |
|                                                                        |
| Verbs Returned on: None                                                |
|                                                                        |
| Subtables Found in: Source_Location, Target_Location                   |
|                                                                        |
| Format:       Character string                                         |
|                                                                        |
|________________________________________________________________________|


CGCSGID:       01134-00500


String Conventions: Leading, imbedded, or trailing space characters
               (X'40') are not allowed.
```

Note: In existing networks where LU names are defined using SNA character set A (includes character set AR, plus the special characters @, #, and

```
$), the LU names may contain any of the three special characters; however,
these characters may not be available on keyboards in every country and
should not be used in new LU names.
```

___ Target_List ___

```
|
|
| Description:  The list of target locations of the requested file,
|               consisting of (NETID, NAU_Name) pairs.
|
| Verbs Supplied on: Send, Send_and_Install, Send_and_Initiate, Install,
|               Initiate, Remove, Accept, Activate,
|
| Verbs Returned on: None
|
| Subtables Found in: None
|
|
```

___ Target_Location ___

```
|
|
| Description:  One of the target locations of a file, consisting of a
|               (NETID, NAU_NAME) pair.
|
| Verbs Supplied on: Send, Reply_To_Send,  Send_and_Install,
|               Send_and_Initiate, Install, Initiate, Remove, Accept,
|               Activate, Reporting_Installation, Reporting_Initiation,
|               Reporting_Removal, Reporting_Acceptance,
|               Activation_Acceptance, Reporting_Activation
|
| Verbs Returned on: None
|
| Subtables Found in: Target_List
|
|
```

___ DS_Priority ___

```
|
|
| Description:  The requested DS priority.
|
|               If this parameter is not specified, the default value of
|               DATA8 is implied.
|
| Verbs Supplied on: Retrieve, Send, Send_and_Install,
|               Send_and_Initiate, Install, Initiate, Remove, Accept
|
| Verbs Returned on: None
|
| Subtables Found in: None
|
| Format:       Enumeration
|
|
```

Possible Values

```
DATA_12
DATA_11
DATA_10
DATA_9
DATA_8
DATA_7
DATA_6
DATA_5
DATA_4
DATA_3
DATA_2
DATA_1
```

___ DS_Security ___

```
|
|
| Description:  The DS security service parameter indicates the security
|               requirements for the distribution.  The combination of
|               this parameter and the DS security_comparison_operator
|               yields the permitted levels of security for the
|               distribution.
|
| Verbs Supplied on: Retrieve, Send, Send_and_Install,
|               Send_and_Initiate, Install, Initiate, Remove, Accept,
|               Activate, Reply_to_Retrieve, Reply_to_Send,
|               Notification_of_Arrival, Reporting_Installation,
|               Reporting_Initiation, Reporting_Removal,
```

```
|               Reporting_Acceptance, Activation_Acceptance,              |
|               Reporting_Activation                                      |
|                                                                         |
| Verbs Returned on: None                                                 |
|                                                                         |
| Subtables Found in: None                                                |
|                                                                         |
| Format:       Enumeration                                               |
|                                                                         |
|_________________________________________________________________________|
```

___ DS_Security_Request _____

```
|                                                                         |
|                                                                         |
| Description:  Specifies the level of DS security requested.             |
|                                                                         |
| Verbs Supplied on: Retrieve, Send, Send_and_Install,                    |
|               Send_and_Initiate, Install, Initiate, Remove, Accept,     |
|               Activate, Reply_to_Retrieve, Reply_to_Send,               |
|               Notification_of_Arrival, Reporting_Installation,          |
|               Reporting_Initiation, Reporting_Removal.                  |
|               Reporting_Acceptance, Activation_Acceptance,              |
|               Reporting_Activation                                      |
|                                                                         |
| Verbs Returned on: None                                                 |
|                                                                         |
| Subtables Found in: DS_Security                                         |
|                                                                         |
| Format:       Enumeration                                               |
|                                                                         |
|_________________________________________________________________________|
```

Possible Values Meaning or Usage

level1 Security is not required.

level2 Security is required.

___ DS_Security_Comp_Op _____

```
|                                                                         |
|                                                                         |
| Description:  The DS security_comparison_operator parameter is used to |
|               allow a range of security service levels for a            |
|               distribution.  For the values it can take, refer to the  |
|               DS documentation.                                         |
|                                                                         |
| Verbs Supplied on: Retrieve, Send, Send_and_Install,                    |
|               Send_and_Initiate, Install, Initiate, Remove, Accept,     |
|               Activate, Reply_to_Retrieve, Reply_to_Send,               |
|               Notification_of_Arrival, Reporting_Installation,          |
|               Reporting_Initiation, Reporting_Removal,                  |
|               Reporting_Acceptance, Activation_Acceptance,              |
|               Reporting_Activation                                      |
|                                                                         |
| Verbs Returned on: None                                                 |
|                                                                         |
| Subtables Found in: DS_Security                                         |
|                                                                         |
| Format:       Enumeration                                               |
|                                                                         |
|_________________________________________________________________________|
```

___ Execution_Window_Timing _____

```
|                                                                         |
|                                                                         |
| Description:  The date and time specifications relative to when the    |
|               entry point is expected to execute the command, based on |
|               its own time-of-day clock.                                |
|                                                                         |
|               If this parameter is not specified, then immediate        |
|               execution is implied.                                     |
|                                                                         |
| Verbs Supplied on: Send_and_Install, Send_and_Initiate, Install,        |
|               Initiate, Remove, Accept, Activate                        |
|                                                                         |
| Verbs Returned on: None                                                 |
|                                                                         |
| Subtables Found in: None                                                |
|                                                                         |
|_________________________________________________________________________|
```

Execute_Not_Before_Time

```
Description:  The date and time before which the entry point may not
              begin execution of the command.

Verbs Supplied on: Send_and_Install, Send_and_Initiate, Install,
              Initiate, Remove, Accept, Activate

Verbs Returned on: None

Subtables Found in: Execution_Window_Timing
```

Execute_Not_Before_Year

```
Description:  The year before which the command may not be executed.

Verbs Supplied on: Send_and_Install, Send_and_Initiate, Install,
              Initiate, Remove, Accept, Activate

Verbs Returned on: None

Subtables Found in: Execution_Window_Timing

Format:       Signed binary integer (1 origin)
```

Execute_Not_Before_Month

```
Description:  The month before which the command may not be executed.

Verbs Supplied on: Send_and_Install, Send_and_Initiate, Install,
              Initiate, Remove, Accept, Activate

Verbs Returned on: None

Subtables Found in: Execution_Window_Timing

Format:       Signed binary integer (1 origin)
```

Execute_Not_Before_Day

```
Description:  The day before which the command may not be executed.

Verbs Supplied on: Send_and_Install, Send_and_Initiate, Install,
              Initiate, Remove, Accept, Activate

Verbs Returned on: None

Subtables Found in: Execution_Window_Timing

Format:       Signed binary integer (1 origin)
```

Execute_Not_Before_Hour

```
Description:  The hour before which the command may not be executed
              (24-hour clock).

Verbs Supplied on: Send_and_Install, Send_and_Initiate, Install,
              Initiate, Remove, Accept, Activate

Verbs Returned on: None

Subtables Found in: Execution_Window_Timing

Format:       Signed binary integer (1 origin)
```

___ Execute_Not_Before_Minute ___

```
Description:  The minute within the hour before which the command may
              not be executed.

Verbs Supplied on: Send_and_Install, Send_and_Initiate, Install,
              Initiate, Remove, Accept, Activate

Verbs Returned on: None

Subtables Found in: Execution_Window_Timing

Format:       Signed binary integer (1 origin)
```

___ Execute_Not_Before_Local_Or_GMT ___

```
Description:  Specifies whether the entry point is to interpret the
              specified time as local time or Greenwich Mean Time
              (GMT).

Verbs Supplied on: Send_and_Install, Send_and_Initiate, Install,
              Initiate, Remove, Accept, Activate

Verbs Returned on: None

Subtables Found in: Execution_Window_Timing

Format:       Enumeration
```

Possible Values

LOCAL
GMT

___ Corequisite_Change_Name_List ___

```
Description:  A list of names of change files that are to be installed
              by the entry point as part of the installation of the
              change file to be fetched and stored (in the case of the
              Send_and_Install verb) or already stored (in the case of
              the Install verb).

Verbs Supplied on: Send_and_Install, Install

Verbs Returned on: None

Subtables Found in: None
```

___ Reported_Change_Name_List ___

```
Description:  A list of names of change files that were installed,
              removed, or accepted by the entry point as requested, or
              as the result of a corequisite relationship with the
              change file named in the request.

Verbs Supplied on: Reporting_Installation, Reporting_Removal,
              Reporting_Acceptance

Verbs Returned on: None

Subtables Found in: None
```

___ Secondary_Installation_Change_Name_List ___

```
Description:  A list of names of change files that were installed
```

```
|               again as the result of a remove request.                |
|                                                                       |
| Verbs Supplied on: Reporting_Removal                                  |
|                                                                       |
| Verbs Returned on: None                                               |
|                                                                       |
| Subtables Found in: None                                              |
|                                                                       |
|_______________________________________________________________________|
```

___ Back-Level_Change_Name_List ___

```
|                                                                       |
| Description:  A list of names of change files that were kept as       |
|               back-level copies as the result of the installation, but |
|               that are now no longer installed.                      |
|                                                                       |
| Verbs Supplied on: Reporting_Installation                             |
|                                                                       |
| Verbs Returned on: None                                               |
|                                                                       |
| Subtables Found in: None                                              |
|                                                                       |
|_______________________________________________________________________|
```

___ Deleted_Change_Name_List ___

```
|                                                                       |
| Description:  A list of names of change files that were deleted as the |
|               result of the installation or acceptance being reported. |
|                                                                       |
| Verbs Supplied on: Reporting_Installation, Reporting_Acceptance       |
|                                                                       |
| Verbs Returned on: None                                               |
|                                                                       |
| Subtables Found in: None                                              |
|                                                                       |
|_______________________________________________________________________|
```

___ Change_File_Name ___

```
|                                                                       |
| Description:  A change file name.                                     |
|                                                                       |
| Verbs Supplied on: Send_and_Install, Install, Reporting_Installation, |
|               Reporting_Removal, Reporting_Acceptance                 |
|                                                                       |
| Verbs Returned on: None                                               |
|                                                                       |
| Subtables Found in: Corequisite_Change_Name_List,                     |
|               Reported_Change_Name_List, Back-Level_Change_Name_List, |
|               Secondary_Installation_Change_Name_List,                |
|               Deleted_Change_Name_List                                |
|                                                                       |
| Format:       Character string                                        |
|                                                                       |
|_______________________________________________________________________|
```

```
CGCSGID:       01134-00500

String Conventions: Leading, imbedded, or trailing space characters
               (X'40') are not allowed.
```

___ Removability ___

```
|                                                                       |
| Description:  Specifies whether or not change files are to be         |
|               installed in a removable manner (so that a subsequent   |
|               Remove command can be issued against them).             |
|                                                                       |
|               If Activation_Use (TRIAL) is specified, then            |
|               Removability (YES) must be specified.                   |
|                                                                       |
| Verbs Supplied on: Send_and_Install, Install                          |
|                                                                       |
| Verbs Returned on: None                                               |
|                                                                       |
| Subtables Found in: None                                              |
|                                                                       |
| Format:       Enumeration                                             |
|                                                                       |
```

Possible Values

YES
DESIRED
NO

Automatic_Removal

Description: Specifies whether the entry point is to remove the change files automatically if a post-test fails.

Unlike a separate Remove request, the entry point does not release the change after a successful automatic removal. This is to avoid resending changes if a simple parameter change by the network operator is sufficient to solve the problem.

This parameter is specified unless Removability (NO) is specified.

Verbs Supplied on: Send_and_Install, Install

Verbs Returned on: None

Subtables Found in: None

Possible Values Meaning or Usage

YES — May not be specified if Removability (DESIRED) is specified.

DESIRED

NO

Pre-Test

Description: Specifies whether or not the entry point is to perform a test on the change files prior to installing them.

Verbs Supplied on: Send_and_Install, Install

Verbs Returned on: None

Subtables Found in: None

Possible Values

YES
DESIRED
NO

Post-Test

Description: Specifies whether or not the entry point is to perform a test on the change files after installing or removing them.

Verbs Supplied on: Send_and_Install, Install, Remove

Verbs Returned on: None

Subtables Found in: None

Possible Values

YES
DESIRED

NO

___ Automatic_Acceptance ___

```
Description:  Specifies whether the entry point is to accept change
              files automatically if installation and any tests
              performed are successful, in order to release resources
              required to maintain removability as soon as possible.

              This parameter is specified unless Removability (NO) is
              specified.

Verbs Supplied on: Send_and_Install, Install

Verbs Returned on: None

Subtables Found in: None
```

___ Automatic_Acceptance_Request ___

```
Description:  Specifies the type of request.

Verbs Supplied on: Send_and_Install, Install

Verbs Returned on: None

Subtables Found in: Automatic_Acceptance

Format:       Enumeration
```

Possible Values Meaning or Usage

YES May not be specified if Removability (DESIRED) is specified.

DESIRED

NO

___ Automatic_Acceptance_Delay ___

```
Description:  The number of days the entry point is expected to wait
              before accepting the change files automatically, if
              installation and any tests performed are successful.

              This parameter is not specified, if
              Automatic_Acceptance_Request (NO) is specified.  If
              Automatic_Acceptance_Request (YES) or (DESIRED) is
              specified, and this parameter is omitted, then immediate
              automatic acceptance is implied.

Verbs Supplied on: Send_and_Install, Install

Verbs Returned on: None

Subtables Found in: Automatic_Acceptance

Format:       Unsigned decimal integer
```

___ Activation_Use ___

```
Description:  Specifies whether the components to be altered by the
              installation process will be trial versions or
              production versions.

Verbs Supplied on: Send_and_Install, Install

Verbs Returned on: None

Subtables Found in: None
```

```
| Format:       Enumeration                                             |
|                                                                       |
|_______________________________________________________________________|
```

Possible Values Meaning or Usage

```
TRIAL          The altered components are used only for an Activate
               request specifying Change_Management_Activation_Use
               (TRIAL_AND_PRODUCTION), and supersede production components
               in that case.

PRODUCTION     The altered components may be used for any activation.
```

___ Alter_Active_Components _____

```
|                                                                       |
|                                                                       |
| Description:  Specifies whether the entry point is allowed to modify  |
|               components in the active system or if a subsequent      |
|               activation is required to effect the modifications in   |
|               the active system.                                       |
|                                                                       |
|               This parameter is optionally specified, unless the entry |
|               point is known not to support it. When omitted, the     |
|               entry point is allowed to modify components in the      |
|               active system, as long as Activation_Use (TRIAL) has not |
|               been specified.                                          |
|                                                                       |
| Verbs Supplied on: Send_and_Install, Install, Remove                  |
|                                                                       |
| Verbs Returned on: None                                               |
|                                                                       |
| Subtables Found in: None                                              |
|                                                                       |
| Format:       Enumeration                                             |
|                                                                       |
|_______________________________________________________________________|
```

Possible Values Meaning or Usage

```
ALLOWED        The entry point is responsible for deciding whether or not
               to modify components in the active system. The results of
               this decision are returned to the focal point in the
               When_Effective parameter as described on page B.7. This
               value may not be specified when the Activation_Use
               parameter specifies (TRIAL).

NO             The entry point may not modify components in the active
               system and as a result will never return to the focal point
               the IN_USE setting of the When_Effective parameter as
               described on page B.7.
```

___ Submission_Type _____

```
|                                                                       |
|                                                                       |
| Description:  Specifies the type of command procedure initiation to be |
|               performed by the entry point.                           |
|                                                                       |
| Verbs Supplied on: Send_and_Initiate, Initiate                        |
|                                                                       |
| Verbs Returned on: None                                               |
|                                                                       |
| Subtables Found in: None                                              |
|                                                                       |
| Format:       Enumeration                                             |
|                                                                       |
|_______________________________________________________________________|
```

Possible Values Meaning or Usage

```
RUN_CL         Run command language
```

___ Object_Number _____

```
|                                                                       |
|                                                                       |
| Description:  Specifies the relative sequence number, in a series of  |
|               command procedures to be initiated, of this command     |
|               procedure. Its value is always one.                     |
|                                                                       |
| Verbs Supplied on: Send_and_Initiate, Initiate                        |
```

```
|                                                                      |
| Verbs Returned on: None                                              |
|                                                                      |
| Subtables Found in: None                                             |
|                                                                      |
| Format:       Unsigned binary integer (1 origin)                     |
|                                                                      |
|______________________________________________________________________|
```

___ Object_Disposition ___

```
|                                                                      |
|                                                                      |
| Description:  Specifies whether the entry point should keep or delete |
|               the newly installed or initiated object after having   |
|               successfully executed the install or initiate command.  |
|                                                                      |
|               This parameter is optionally specified, unless the entry |
|               point is known not to support it. When omitted, the    |
|               decision concerning the disposition of the object is   |
|               made by the entry point.                               |
|                                                                      |
| Verbs Supplied on: Send_and_Install, Send_and_Initiate, Install,     |
|               Initiate                                               |
|                                                                      |
| Verbs Returned on: None                                              |
|                                                                      |
| Subtables Found in: None                                             |
|                                                                      |
| Format:       Enumeration                                            |
|                                                                      |
|______________________________________________________________________|
```

Possible Values

KEEP

DELETE

___ Force ___

```
|                                                                      |
|                                                                      |
| Description:  Specifies whether or not the receiving entry point     |
|               should proceed with the activation based on its quiesced |
|               state.  (In either case, the receiver may reply with   |
|               Activation Acceptance before activation is attempted.) |
|                                                                      |
| Verbs Supplied on: Activate                                          |
|                                                                      |
| Verbs Returned on: None                                              |
|                                                                      |
| Subtables Found in: None                                             |
|                                                                      |
|______________________________________________________________________|
```

___ Force_Request ___

```
|                                                                      |
|                                                                      |
| Description:  Specifies the type of request.                         |
|                                                                      |
| Verbs Supplied on: Activate                                          |
|                                                                      |
| Verbs Returned on: None                                              |
|                                                                      |
| Subtables Found in: Force                                            |
|                                                                      |
| Format:       Enumeration                                            |
|                                                                      |
|______________________________________________________________________|
```

Possible Values Meaning or Usage

NO — Reject if the entry point is not quiesced, otherwise activate

YES — Activate regardless.

___ Change_Management_Activation_Use ___

```
Description:  Specifies which components altered by changes will be
              used during the activation.

              It is specified unless the entry point is known not to
              support Change_Management_Activation_Use.

Verbs Supplied on: Activate

Verbs Returned on: None

Subtables Found in: None

Format:       Enumeration
```

Possible Values Meaning or Usage

```
TRIAL_AND_PRODUCTION Components altered by changes installed on trial will
               be used and supersede both those altered by changes
               installed in production as well as unchanged components.

PRODUCTION_ONLY Components altered by changes installed in production will
               be used and supersede unchanged components, but those with
               changes installed on trial will not be used.
```

___ Time_Stamp ___

```
Description:  Specifies the date and time that the request was
              executed by the entry point. In the case of FS reports,
              the date and time in the DS header (representing when
              Send_Distribution was issued by the entry point) is
              used.  In the case of MS reports, the date and time of
              command execution is specified by the entry point in the
              reply CP-MSU.

Verbs Supplied on: Reply_to_Retrieve, Reply_to_Send,
              Notification_of_Arrival, Reporting_Installation,
              Reporting_Initiation, Reporting_Removal,
              Reporting_Acceptance, Activation_Acceptance,
              Reporting_Activation

Verbs Returned on: None

Subtables Found in: None
```

___ Time_Stamp_Year ___

```
Description:  The year.

Verbs Supplied on: Reply_to_Retrieve, Reply_to_Send,
              Notification_of_Arrival, Reporting_Installation,
              Reporting_Initiation, Reporting_Removal,
              Reporting_Acceptance, Activation_Acceptance,
              Reporting_Activation

Verbs Returned on: None

Subtables Found in: Time_Stamp

Format:       Unsigned decimal integer
```

___ Time_Stamp_Month ___

```
Description:  The month.

Verbs Supplied on: Reply_to_Retrieve, Reply_to_Send,
              Notification_of_Arrival, Reporting_Installation,
              Reporting_Initiation, Reporting_Removal,
              Reporting_Acceptance, Activation_Acceptance,
              Reporting_Activation

Verbs Returned on: None
```

```
|                                                                      |
| Subtables Found in: Time_Stamp                                       |
|                                                                      |
| Format:       Unsigned decimal integer                               |
|                                                                      |
|______________________________________________________________________|
```

```
___ Time_Stamp_Day ______________________________________________________
|                                                                      |
|                                                                      |
| Description:  The day.                                               |
|                                                                      |
| Verbs Supplied on: Reply_to_Retrieve, Reply_to_Send,                 |
|               Notification_of_Arrival, Reporting_Installation,       |
|               Reporting_Initiation, Reporting_Removal,               |
|               Reporting_Acceptance, Activation_Acceptance,           |
|               Reporting_Activation                                   |
|                                                                      |
| Verbs Returned on: None                                              |
|                                                                      |
| Subtables Found in: Time_Stamp                                       |
|                                                                      |
| Format:       Unsigned decimal integer                               |
|                                                                      |
|______________________________________________________________________|
```

```
___ Time_Stamp_Hour _____________________________________________________
|                                                                      |
|                                                                      |
| Description:  The hour.                                              |
|                                                                      |
| Verbs Supplied on: Reply_to_Retrieve, Reply_to_Send,                 |
|               Notification_of_Arrival, Reporting_Installation,       |
|               Reporting_Initiation, Reporting_Removal,               |
|               Reporting_Acceptance, Activation_Acceptance,           |
|               Reporting_Activation                                   |
|                                                                      |
| Verbs Returned on: None                                              |
|                                                                      |
| Subtables Found in: Time_Stamp                                       |
|                                                                      |
| Format:       Unsigned decimal integer                               |
|                                                                      |
|______________________________________________________________________|
```

```
___ Time_Stamp_Minute ___________________________________________________
|                                                                      |
|                                                                      |
| Description:  The minute.                                            |
|                                                                      |
| Verbs Supplied on: Reply_to_Retrieve, Reply_to_Send,                 |
|               Notification_of_Arrival, Reporting_Installation,       |
|               Reporting_Initiation, Reporting_Removal,               |
|               Reporting_Acceptance, Activation_Acceptance,           |
|               Reporting_Activation                                   |
|                                                                      |
| Verbs Returned on: None                                              |
|                                                                      |
| Subtables Found in: Time_Stamp                                       |
|                                                                      |
| Format:       Unsigned decimal integer                               |
|                                                                      |
|______________________________________________________________________|
```

```
___ Time_Stamp_Second ___________________________________________________
|                                                                      |
|                                                                      |
| Description:  The second.                                            |
|                                                                      |
| Verbs Supplied on: Reply_to_Retrieve, Reply_to_Send,                 |
|               Notification_of_Arrival, Reporting_Installation,       |
|               Reporting_Initiation, Reporting_Removal,               |
|               Reporting_Acceptance, Activation_Acceptance,           |
|               Reporting_Activation                                   |
|                                                                      |
| Verbs Returned on: None                                              |
|                                                                      |
| Subtables Found in: Time_Stamp                                       |
|                                                                      |
| Format:       Unsigned decimal integer                               |
|                                                                      |
```

Time_Stamp_Second_Hundredths

Description: Hundredths of a second.

Verbs Supplied on: Reply_to_Retrieve, Reply_to_Send, Notification_of_Arrival, Reporting_Installation, Reporting_Initiation, Reporting_Removal, Reporting_Acceptance, Activation_Acceptance, Reporting_Activation

Verbs Returned on: None

Subtables Found in: Time_Stamp

Format: Unsigned decimal integer

Installation_Results

Description: Specifies the status of installation and when it is effective.

Verbs Supplied on: Reporting_Installation

Verbs Returned on: None

Subtables Found in: None

Installation_Status

Description: Specifies whether or not installation was successful.

Verbs Supplied on: Reporting_Installation

Verbs Returned on: None

Subtables Found in: None

Format: Enumeration

Possible Values Meaning or Usage

SUCCESSFUL

NOT_SUCCESSFUL Failed with backout

NOT_SUCCESSFUL_UNRECOVERED Failed with abend

NOT_ATTEMPTED Not attempted and will not attempt.

WILL_ATTEMPT Will attempt.

When_Effective

Description: Specifies when the changed components will be in use.

Verbs Supplied on: Reporting_Installation, Reporting_Removal

Verbs Returned on: None

Subtables Found in: None

Format: Enumeration

Possible Values Meaning or Usage

```
IN_USE         Changed components are now in use.  This setting is not
               allowed when Activation_Use (TRIAL) or
               Alter_Active_Components (NO) is specified on the request.

ACTIVATION_REQUIRED Components are changed, but activation is required to
               put them in use.

NOT_APPLICABLE Not applicable, because the request was not attempted.
```

Initiation_Results

```
Description:  Specifies the status of initiation.

Verbs Supplied on: Reporting_Initiation

Verbs Returned on: None

Subtables Found in: None
```

Job_Element_State

```
Description:  Specifies the state of the command procedure (also
              referred to as job element) to be initiated.

Verbs Supplied on: Reporting_Initiation

Verbs Returned on: None

Subtables Found in: None

Format:       Enumeration
```

Possible Values Meaning or Usage

```
ACCEPTED       Will attempt

ENDED          Sucessfully or Unsuccsessfully
```

Completion_Code

```
Description:  Specifies the completion code returned from the
              initiated command procedure. Its value is zeros when the
              Job_Element_State is ACCEPTED.

Verbs Supplied on: Reporting_Initiation

Verbs Returned on: None

Subtables Found in: None

Format:       Signed binary integer (zero origin)
```

When_Secondary_Effective

```
Description:  Specifies when the changed components will be in use
              after secondary installation.

Verbs Supplied on: Reporting_Removal

Verbs Returned on: None

Subtables Found in: None
```

```
| Format:       Enumeration                                              |
|                                                                        |
|____________________________________________________________________|
```

Possible Values Meaning or Usage

```
IN_USE         Changed components are now in use.

ACTIVATION_REQUIRED Components are changed, but activation is required to
               put them in use.
```

___ Pre-Test_Status _____

```
|                                                                        |
|                                                                        |
| Description:  Specifies whether or not the pre-test was successful.    |
|                                                                        |
| Verbs Supplied on: Reporting_Installation                              |
|                                                                        |
| Verbs Returned on: None                                                |
|                                                                        |
| Subtables Found in: None                                               |
|                                                                        |
| Format:       Enumeration                                              |
|                                                                        |
|____________________________________________________________________|
```

Possible Values Meaning or Usage

```
SUCCESSFUL

NOT_SUCCESSFUL

NOT_ATTEMPTED  Not attempted and will not attempt.
```

___ Post-Test_Status _____

```
|                                                                        |
|                                                                        |
| Description:  Specifies whether or not the post-test was successful.   |
|                                                                        |
| Verbs Supplied on: Reporting_Installation, Reporting_Removal           |
|                                                                        |
| Verbs Returned on: None                                                |
|                                                                        |
| Subtables Found in: None                                               |
|                                                                        |
| Format:       Enumeration                                              |
|                                                                        |
|____________________________________________________________________|
```

Possible Values Meaning or Usage

```
SUCCESSFUL

NOT_SUCCESSFUL

NOT_ATTEMPTED  Not attempted and will not attempt.
```

___ Automatic_Removal_Results _____

```
|                                                                        |
|                                                                        |
| Description:  Specifies the status of automatic removal and when it is |
|               effective.                                               |
|                                                                        |
| Verbs Supplied on: Reporting_Installation                              |
|                                                                        |
| Verbs Returned on: None                                                |
|                                                                        |
| Subtables Found in: None                                               |
|                                                                        |
|____________________________________________________________________|
```

___ Automatic_Removal_Status _____

```
|                                                                    |
|                                                                    |
| Description:  Specifies whether or not automatic removal was       |
|               successful.                                          |
|                                                                    |
| Verbs Supplied on: Reporting_Installation                          |
|                                                                    |
| Verbs Returned on: None                                            |
|                                                                    |
| Subtables Found in: None                                           |
|                                                                    |
| Format:       Enumeration                                          |
|                                                                    |
|____________________________________________________________________|
```

Possible Values Meaning or Usage

SUCCESSFUL

NOT_SUCCESSFUL Failed with backout

NOT_SUCCESSFUL_UNRECOVERED Failed with abend

NOT_ATTEMPTED Not attempted and will not attempt.

___ Automatic_Acceptance_Status _____

```
|                                                                    |
|                                                                    |
| Description:  Specifies whether or not automatic acceptance was    |
|               successful.                                          |
|                                                                    |
| Verbs Supplied on: Reporting_Installation                          |
|                                                                    |
| Verbs Returned on: None                                            |
|                                                                    |
| Subtables Found in: None                                           |
|                                                                    |
| Format:       Enumeration                                          |
|                                                                    |
|____________________________________________________________________|
```

Possible Values Meaning or Usage

SUCCESSFUL

NOT_SUCCESSFUL

NOT_ATTEMPTED Not attempted and will not attempt.

WILL_ATTEMPT Will attempt at the end of the delay specified.

___ Removability_Status _____

```
|                                                                    |
|                                                                    |
| Description:  Specifies whether the change file was installed      |
|               removably or not.                                    |
|                                                                    |
|               An entry point does not make a decision of which way to |
|               install a change file independent from the focal-point |
|               request. However, this status information is returned in |
|               the report.                                          |
|                                                                    |
| Verbs Supplied on: Reporting_Installation                          |
|                                                                    |
| Verbs Returned on: None                                            |
|                                                                    |
| Subtables Found in: None                                           |
|                                                                    |
| Format:       Enumeration                                          |
|                                                                    |
|____________________________________________________________________|
```

Possible Values Meaning or Usage

INSTALLED_REMOVABLY

INSTALLED_NONREMOVABLY

NOT_INSTALLED The installation was not successful.

___ Activation_Use_Status ___

```
Description:  Specifies whether the change file was installed on trial
              or in production.

              An entry point does not make a decision of which way to
              install a change file independent from the focal-point
              request. However, this status information is returned in
              the report.

Verbs Supplied on: Reporting_Installation

Verbs Returned on: None

Subtables Found in: None

Format:       Enumeration
```

Possible Values Meaning or Usage

TRIAL

PRODUCTION

NOT_INSTALLED The installation was not successful.

___ Removal_Results ___

```
Description:  Specifies the status of removal and when it is
              effective.

Verbs Supplied on: Reporting_Removal

Verbs Returned on: None

Subtables Found in: None
```

___ Removal_Status ___

```
Description:  Specifies whether or not removal was successful.

Verbs Supplied on: Reporting_Removal

Verbs Returned on: None

Subtables Found in: None

Format:       Enumeration
```

Possible Values Meaning or Usage

SUCCESSFUL

NOT_SUCCESSFUL Failed with backout

NOT_SUCCESSFUL_UNRECOVERED Failed with abend

NOT_ATTEMPTED Not attempted and will not attempt.

WILL_ATTEMPT Will attempt.

___ Acceptance_Results ___

Description: Specifies the status of acceptance.

Verbs Supplied on: Reporting_Acceptance

Verbs Returned on: None

Subtables Found in: None

___ Acceptance_Status ___

Description: Status.

Verbs Supplied on: Reporting_Acceptance

Verbs Returned on: None

Subtables Found in: None

Format: Enumeration

Possible Values

SUCCESSFUL

NOT_SUCCESSFUL

NOT_ATTEMPTED Not attempted and will not attempt.

WILL_ATTEMPT Will attempt.

___ Activation_Acceptance_Status ___

Description: Specifies whether or not activation will be attempted.

Verbs Supplied on: Activation_Acceptance

Verbs Returned on: None

Subtables Found in: None

Format: Enumeration

Possible Values Meaning or Usage

NOT_ATTEMPED Not attempted and will not attempt

WILL_ATTEMPT Will attempt

___ Additional_Reports ___

Description: Indicates that one or more additional reports will be generated as a result of the requested operation.

It is present when additional reports are to be generated.

Verbs Supplied on: Activation_Acceptance

Verbs Returned on: None

Subtables Found in: None

Format: Enumeration

Possible Values

YES

NO

Activation_Status

Description: Specifies whether or not activation was successful.

Verbs Supplied on: Reporting_Activation

Verbs Returned on: None

Subtables Found in: None

Format: Enumeration

Possible Values

SUCCESSFUL

NOT_SUCCESSFUL

Activation_Stage

Description: Indicates what stage of entry point activation has been achieved.

It is present when activation status is successful.

Verbs Supplied on: Reporting_Activation

Verbs Returned on: None

Subtables Found in: None

Format: Enumeration

Possible Values Meaning or Usage

INITIAL Initial connectivity complete

HARDWARE Processor harware active

SOFTWARE Operating system, communications subsystem and network management subsystem active

Change_Management_Activation_Use_Status

Description: Specifies whether trial components were searched and used if found during activation and optionally, whether a special activation-related condition exists at the reporting entry point. Alternately, it may indicate that change management or the change management activation use function is not supported in the entry point.

It is present when activation status is successful.

An entry point does not make a decision of which way to activate itself, independent from the focal point-request. However, this status information is returned in the report.

Verbs Supplied on: Reporting_Activation

Verbs Returned on: None

Subtables Found in: None

___ C_M_Activation_Use_Status_Report _____

Description: Specifies the reported status.

Verbs Supplied on: Reporting_Activation

Verbs Returned on: None

Subtables Found in: Change_Management_Activation_Use_Status

Format Enumeration

Possible Values Meaning or Usage

TRIAL_AND_PRODUCTION Trial components were searched and used if found.

PRODUCTION_ONLY Trial components were not searched.

NOT_APPLICABLE Change_Management_Activation_Use is not supported by the reporting entry point.

___ C_M_Activation_Use_Status_Condition _____

Description: Specifies whether a special activation related condition exists at the reporting entry point.

When absent, no special condition is being reported.

Verbs Supplied on: Reporting_Activation

Verbs Returned on: None

Subtables Found in: Change_Management_Activation_Use_Status

Format Enumeration

Possible Values Meaning or Usage

YES One of the following special conditions is being reported:

When C_M_Activation_Use_Status_Report is "Trial and Production," only production components were found despite searching for trial versions.

When C_M_Activation_Use_Status_Report is "Production Only," production activation was locally requested to override a previous failing trial activation.

NO No special condition is being reported.

___ Secondary_Installation_Results _____

Description: Specifies that one or more change files were installed as the result of a removal of other change files. This parameter is returned only when the removal status is successful.

Verbs Supplied on: Reporting_Removal

Verbs Returned on: None

Subtables Found in: None

___ Secondary_Installation_Status _____

```
|                                                                      |
|                                                                      |
| Description:  Specifies that secondary installation was successful.  |
|                                                                      |
| Verbs Supplied on: Reporting_Removal                                 |
|                                                                      |
| Verbs Returned on: None                                              |
|                                                                      |
| Subtables Found in: None                                             |
|                                                                      |
| Format:       Enumeration                                            |
|                                                                      |
|______________________________________________________________________|
```

Possible Values

SUCCESSFUL

___ Secondary_Activation_Use_Status _____

```
|                                                                      |
|                                                                      |
| Description:  Specifies whether secondary installation was on trial or |
|               in production.                                         |
|                                                                      |
| Verbs Supplied on: Reporting_Removal                                 |
|                                                                      |
| Verbs Returned on: None                                              |
|                                                                      |
| Subtables Found in: None                                             |
|                                                                      |
| Format:       Enumeration                                            |
|                                                                      |
|______________________________________________________________________|
```

Possible Values

TRIAL

PRODUCTION

___ Object_Class _____

```
|                                                                      |
|                                                                      |
| Description:  The data object class, as defined by SNA/FS, of the    |
|               referenced object or objects.                          |
|                                                                      |
|               This parameter is supplied if all of the SNA/FS global |
|               names do not have either "MCODE" or "MCUST" as their   |
|               first token.                                           |
|                                                                      |
| Verbs Supplied on: Retrieve, Send, Delete, Send_and_Install,         |
|               Send_and_Initiate, Install, Initiate, Remove, Accept,  |
|               Reply_to_Retrieve, Reply_to_Send, Reply_to_Delete,     |
|               Notification_of_Arrival, Reporting_Installation,       |
|               Reporting_Initiation, Reporting_Removal,               |
|               Reporting_Acceptance                                   |
|                                                                      |
| Verbs Returned on: None                                              |
|                                                                      |
| Subtables Found in: None                                             |
|                                                                      |
| Format:       As defined by the Data_Object_Class subfield of the    |
|               SNA/FS Data_Object_Acceptance GDS variable variable with |
|               registered codepoint X'1537'                           |
|                                                                      |
|______________________________________________________________________|
```

___ FS_Action_Summary _____

```
|                                                                      |
|                                                                      |
```

```
| Description:  SNA/FS action summary, as defined by SNA/FS.              |
|                                                                         |
| Verbs Supplied on: Reply_to_Retrieve, Reply_to_Send,                    |
|               Notification_of_Arrival, Reporting_Installation,          |
|               Reporting_Initiation                                      |
|                                                                         |
| Verbs Returned on: None                                                 |
|                                                                         |
| Subtables Found in: None                                                |
|                                                                         |
|_________________________________________________________________________|
```

___ **SNA_Condition_Report** ___

```
|                                                                         |
| Description:  SNA Condition Report, as defined by SNA/FS.               |
|                                                                         |
| Verbs Supplied on: Reply_to_Retrieve, Reply_to_Send,                    |
|               Notification_of_Arrival, Reporting_Installation,          |
|               Reporting_Initiation, Reporting_Removal,                  |
|               Reporting_Acceptance, Activation_Acceptance,              |
|               Reporting_Activation                                      |
|                                                                         |
|               This parameter is present once for each condition report |
|               returned in either the agent object or server object.     |
|                                                                         |
| Verbs Returned on: None                                                 |
|                                                                         |
| Subtables Found in: None                                                |
|                                                                         |
|_________________________________________________________________________|
```

___ **Detailed_Data** ___

```
|                                                                         |
| Description:  Reports conditions unique to entry-point                  |
|               implementations.                                          |
|                                                                         |
| Verbs Supplied on: Reporting_Installation, Reporting_Initiation,        |
|               Reporting_Removal, Reporting_Acceptance,                  |
|               Activation_Acceptance, Reporting_Activation               |
|                                                                         |
|               This parameter is present once for each such item         |
|               returned in either the agent object or server object.     |
|                                                                         |
| Verbs Returned on: None                                                 |
|                                                                         |
| Subtables Found in: None                                                |
|                                                                         |
|_________________________________________________________________________|
```

___ **Return_Code** ___

```
|                                                                         |
| Description:  The results of verb execution.                            |
|                                                                         |
| Verbs Supplied on: None                                                 |
|                                                                         |
| Verbs Returned on: Retrieve, Send, Send_and_Install,                    |
|               Send_and_Initiate, Install, Initiate, Remove, Accept,     |
|               Activate, Reply_to_Retrieve, Reply_to_Send,               |
|               Notification_of_Arrival, Reporting_Installation,          |
|               Reporting_Initiation, Reporting_Removal,                  |
|               Reporting_Acceptance, Activation_Acceptance,              |
|               Reporting_Activation,                                     |
|                                                                         |
| Subtables Found in: None                                                |
|                                                                         |
| Format:       Enumeration                                               |
|                                                                         |
|_________________________________________________________________________|
```

Possible Values

```
OK
FS_ORIGIN_EXCEPTION
DS_ORIGIN_EXCEPTION
```

C.0 Appendix C. SNA/FS File Names Defined by SNA/MS

The total length of all ten tokens is restricted to (64-n+1) bytes, where n = number of tokens specified. This is a practical constraint, not the natural architectural limit.

Tokens shown as product specifiable may be specified using SNA/FS guidelines (for example, regarding hierarchical levels of authority), however, this is not required.

Implementations may choose to assign more stringent matching requirements when they create files, but not less stringent ones. For example, an implementation creating a file may assign an attribute of MUST_MATCH to a token described in the architecture as having an attribute of NEED_NOT_MATCH (but not the other way around).

SNA/MS supports two categories of global names: change management structured and enterprise structured.

A change management structured name has a format that is predefined by SNA/MS and always contains an object sub class descriptor as its first token. SNA/MS supports change management structured names only for objects belonging to the microcode or microcode customizing data object sub class, and thus change management structured names always have either MCODE or MCUST as their first token, respectively. Conversely, if a name has MCODE or MCUST as its first token, the name is considered to be change management structured and must name a data object from the microcode or microcode customizing data object sub class, respectively.

An enterprise structured name is defined by the enterprise that owns the object. Its first token should be the owner's Enterprise ID used in the the structured network IDs registered by IBM for the enterprise. Enterprise structured names are supported by SNA/MS for all object classes, including the microcode and microcode-customizing data sub classes. An enterprise structured name for an object belonging to the executable or processable object class may optionally contain, as a subset of its tokens, one of the architecturally defined subtrees defined by SNA/MS beginning on page C.0. Architecturally defined subtrees may appear in the low-order portion of the name, following the enterprise ID and one or more additional tokens that define the changed component's containment hierarchy. Architecturally defined subtrees may optionally be followed by one or more qualifying version tokens.

Change Management Structured Names

Table C-1. Identification Tokens for Microcode Objects Using Change Management Structured Names

Token number	Token Attributes (assigned when file is created)	Contents
1	Must match, not generable, type unspecified	C'MCODE'
2	Must match, not generable, type unspecified	Machine type
3	Need not match, not generable, type unspecified	Model number. If none assigned, or if this change is for all model numbers, C'NA'.
4	Need not match, not generable, type unspecified	One of the following: ° C'PATCH' ° C'MCF' ° C'SUFFIXEC' ° C'MAINTEC' ° C'FUNCTEC' ° C'FEATURE'
5	Need not match, not generable, type unspecified	Microcode change name (e.g., patch or MCF number, EC level)
6 - 10	Product specifiable	Product specifiable

Table C-2. Identification Tokens for Microcode Customizing Data Objects Using Change Management Structured Names

Token number	Token Attributes (assigned when file is created)	Contents
1	Must match, not generable, type unspecified	C'MCUST'

2	Must match, not generable, type unspecified	Machine type
3	Need not match, not generable, type unspecified	Model number. If none assigned, or if this change is for all model numbers, C'NA'.
4	Need not match, not generable, type is NETID	NETID portion of location name of node for which this customizing data was created. If not applicable, C'NA'.
5	Need not match, not generable, type is LUNAME	LUNAME portion of location name of node for which this customizing data was created. If not applicable, C'NA'.
6	Need not match, not generable, type unspecified	Product defined, but must be specified to ensure uniqueness
7 - 10	Product specifiable	Product specifiable

Note: In existing networks where LU names or network IDs are defined using SNA character set A (includes character set AR, plus the special characters @, #, and $), the LU names or network IDs may contain any of the three special characters; however, these characters may not be available on keyboards in every country and should not be used in new LU names or network IDs.

Enterprise Structured Names

Table C-3. Identification Tokens for All Objects Using Enterprise Structured Names

Token number	**Token Attributes (assigned when file is created)**	**Contents**
1	Must match, not generable, not subtree ID, unordered - type unspecified	Enterprise ID (from the Structured Netid registered with IBM)
2 - 10	User specifiable	User specifiable

The Enterprise ID token of the enterprise structured name consists of a 2-byte country code followed by a 4-byte enterprise code.

The structured network ID is defined as follows:

```
countrycode(2 bytes)||enterprisecode(4 bytes)||networksuffix(2 bytes)
```

The enterprise ID is defined as follows (notice that the 2-byte network suffix is not part of it):

```
countrycode(2 bytes)||enterprisecode(4 bytes)
```

The user-specifiable tokens of the enterprise structured name contain the one or more component name tokens and may contain, as a subset, one of the architecturally defined subtrees as defined in the tables on the following pages. An architecturally defined subtree may be used in global names that represent a *change* (to be consolidated) or a *changed* (consolidated into) object. Architecturally defined subtrees are identified by the presence of an SNA/FS registered subtree identifier (STI) token having the STI token attribute as defined by SNA/FS. SNA/MS defines the manner in which a *change* object having one of various architecturally defined subtrees in its name applies to a *changed* object having the *group* architecturally defined subtree in its name.

While microcode and microcode-customization change objects are supported by enterprise structured names, the ability to manage changes using the MCODE and MCUST change management structured names is unaffected.

The management of changes using enterprise structured names and architecturally defined subtrees is based on the notion of the SNA/FS group object and the following change management concepts apply:

- A clear distinction is made between a distributed change (i.e., to be consolidated) object and the target (i.e., consolidated into) object.

The target object is also known as the *group* object.

° Except in preparation nodes, change objects are consolidated into the group (i.e., changed) object.

° Although the group is a persistent (as opposed to change) object, there is no persistence of any installed child objects (i.e., elements of the group) of the installed parent group object.

° There are no change management operations upon consolidated child objects. l Such operations act only upon parent group (i.e., consolidated into) objects.

° The token in the name tree at which a group object is created can vary from node to node and time to time without affecting the change management operations that change the groups.

° All objects in a family share the same level number space. Parent changes with a level that is less than already installed child changes are permitted only if system dependent logic can ensure that no child back-levelling will result.

° Group objects are never transported.

Change Management Architecturally Defined Subtrees

Table C-4. Identification Tokens of the Architecturally Defined Subtree for Group Objects

Token number	**Token Attributes (assigned when file is created)**	**Contents**
1	Must match, not generable, subtree ID, unordered - type unspecified	C'GRP'
2	Need not match, not generable, not subtree ID, unordered LUNAME	The LU name of the focal point issuing the most recent command affecting this group object
3	Need not match, not generable, not subtree ID, ordered date	The date of the most recent command affecting this group object
4	Need not match, not generable, not subtree ID, ordered numeric	The command sequence number of the most recent command affecting this group object

For enterprise structured names representing changed (i.e., consolidated into) objects, this is the SNA/MS definition of the SNA/FS owned GRP subtree for group objects. The GRP subtree identifier token is registered by SNA/FS.

The existence of a group object and its elemental (i.e., children and grandchildren) objects is implied by the component tokens in the SNA/FS hierarchical name. All change management operations for the components represented by the group's component name are targeted to the group object. While SNA/FS supports group objects that contain other group objects, SNA/MS does not support managing changes on nested groups. Group object names include a GRP token at the end of their common root token string. Elementary objects do not include a GRP token.

Group objects must follow the same rules as do elementary objects with respect to supporting the function of the SNA/FS global catalog. Specifically, group object names must be unique and object immutability must be enforced. It is the responsibility of the name tree designer and the creator of the objects named with GRP subtrees to ensure that subtree tokens provide for unique object names. For SNA/MS, the subtree tokens as defined above satisfy this requirement.

Refresh Change Objects

Table C-5. Identification Tokens of the Architecturally Defined Subtree for

Token number	**Token Attributes (assigned when file is created)**	**Contents**
1	Must match, not generable, subtree ID, unordered - type unspecified	C'REF'

2	Need not match, not generable, not subtree ID, ordered numeric	Level: The maintenance level to which this change object refreshes

For enterprise structured names representing change (i.e., to be consolidated) objects, this is the definition of the SNA/MS owned REF subtree to be used in distributed *refresh* change objects intended to operate on SNA/MS created group software, microcode and data objects. The REF subtree identifier token is registered by SNA/FS.

The REF subtree is used in the *refresh* type of distributed change objects. These objects create or replace all or some of the contents of a group object, depending upon the granularity of the component name in the name of the distributed refresh change object. When replacing all the contents, the resultant level of the group object is indicated by the *level* token in the REF subtree in the name of the refresh change object. When elemental (i.e., child or grandchild) refresh change objects are installed, their installation does not affect the level of the parent group as a whole, but instead affects the level of the child or grandchild portion of the group object as indicated by the *level* token.

When the parent group is refreshed, the previous parent refresh and all subsequent related change entries, including those for children and grandchildren, are deleted from the SNA/FS global catalog.

Update Change Objects

Table C-6. Identification Tokens of the Architecturally Defined Subtree for

Token number	**Token Attributes (assigned when file is created)**	**Contents**
1	Must match, not generable, subtree ID, unordered - type unspecified	C'UPD'
2	Need not match, not generable, not subtree ID, ordered numeric	Oldlvl: The lowest maintenance level from which this change can update
3	Need not match, not generable, not subtree ID, ordered numeric	Newlvl: The maintenance level to which this change object updates

For enterprise structured names representing change (i.e., to be consolidated) objects, this is the definition of the SNA/MS owned UPD subtree to be used in distributed *update* change objects intended to operate on SNA/MS created group software, microcode and data objects. The UPD subtree identifier token is registered by SNA/FS.

The UPD subtree is used in the *update* type of distributed change objects. These objects update the level of all or some of the contents of a group object, depending upon the granularity of the component name in the name of the distributed update change object. When updating all the contents, the resultant level of the group object is indicated by the *newlvl* token of the UPD subtree in the name of the installed update change object. When elemental (i.e., child or grandchild) update change objects are installed, their installation does not affect the level of the parent group as a whole, but instead updates the level of the child or grandchild portion of the group object as indicated by the *newlvl* token.

The net effect of installing an update change object that raises the level of a group object or portion thereof to *newlvl* is the same as having installed a refresh change object for the same component with the *level* token of the REF subtree having the same value as the *newlvl* token of the UPD subtree.

In order for an update to change the level of a group object or some part thereof, the existing level of whatever is being updated must be greater than or equal to the *oldlvl* token but less than the *newlvl* token of the UPD subtree specified for the update change object.

Fix Change Objects

Table C-7. Identification Tokens of the Architecturally Defined Subtree for

Token number	**Token Attributes (assigned when file is created)**	**Contents**
1	Must match, not generable, subtree ID, unordered - type unspecified	C'FIX'

2	Need not match, not generable, not subtree ID, ordered numeric	Fixlvl: The maintenance level to which this fix applies
3	Must match, not generable, not subtree ID, unordered - type unspecified	Fixid: The fix identifier

For enterprise structured names representing change (i.e., to be consolidated) objects, this is the definition of the SNA/MS owned FIX subtree to be used in distributed *fix* change objects intended to operate on SNA/MS created group software, microcode and data objects. The FIX subtree identifier token is registered by SNA/FS.

The FIX subtree is used in the *fix* type of distributed change objects. These objects fix a specific problem affecting all or some of the contents of a group object, depending upon the granularity of the component name in the name of the distributed fix change object. The level of a group object or its effected portion is never modified by the installation of a fix change object, regardless of whether it was installed as a group or elemental (i.e., child or grandchild) change.

Fixes do not delete previous related change entries from the SNA/FS catalog.

In order to fix a problem effecting a group object or some part thereof, the fix identifier (e.g., PTF number) must be specified by the *fixid* token of the FIX subtree and the existing level of whatever is being fixed must be specified in the *fixlvl* token.

D.0 Appendix D. MS Transaction Program Names

Table D-1. Architecturally-Defined Management Services Transaction Programs

Name (hex)	Descriptive name	Notes
	MS Service Transaction Programs	
These are programs that are the targets for the APPC ALLOCATE verb, so these names will appear in an LU 6.2 Attach.		
X'23F0F0F0'	MSU_HANDLER	Performs application program routing by MS major vector key (not current level of Multiple Domain Support)
X'23F0F0F1'	MDS_RECEIVE	Multiple Domain Support transaction program for receiving data.
X'23F0F0F2'	MDS_HP_RECEIVE	Multiple Domain Support transaction program for receiving data over user-mode sessions as defined in MULTIPLE_DOMAIN_SUPPORT Optional Subset 1 (High-Performance Option).
	MS Cross-Discipline Application Programs	
These are programs that are targets for MDS-MUs. These names will appear only in the MDS Routing Information (X'1311') GDS variable. These programs are never the target for an APPC ALLOCATE verb, so they will not appear in an LU 6.2 Attach.		
X'23F0F1F0'	MDS_ROUTER	
X'23F0F1F1'	MS_CAPABILITIES	
	MS Discipline-Specific Application Programs	
These are programs that are targets for MDS-MUs. These names will appear only in the MDS Routing Information (X'1311') GDS variable. These programs are never the target for an APPC ALLOCATE verb, so they will not appear in an LU 6.2 Attach.		
X'23F0F1F4'	EP_COMMON_OPS_SERVICES	
X'23F0F1F5'	COMMON_OPS_SERVICES_NETOP	

X'23F0F1F6'	EP_OPERATIONS_MGMT	
X'23F0F1F7'	OPERATIONS_MGMT_NETOP	Also used as MS category code by MS_CAPS function set.
X'23F0F3F0'	EP_ALERT	
X'23F0F3F1'	ALERT_NETOP	Also used as MS category code by MS_CAPS function set.

As new Function Set Groups are defined, additional names will be added.

| E.0 Appendix E. FSM Notation

Subtopics:

- E.1 FSM Notation

| E.1 FSM Notation

| A finite-state machine (FSM) is a combination of processing and memory, where the memory consists of the *state* of the FSM. The state can take one of a small number of named values (the *state names*). An FSM is defined by a matrix that lists the states and specifies the processing to be performed when the FSM is called. This processing typically depends on the current state of the FSM and on the input passed to the FSM, and may change the FSM state (resulting in a *state transition*) and produce output. Within this matrix definition, each state is given a number as well as its name, for notational convenience.

| Within FSMs, a naming convention using periods (.) is employed to denote subcomponents of a structure. Another special character, the underscore (_), appears in names to indicate a name phrase rather than decomposition.

| The following operations may be performed on an FSM:

| ° Call. Processing is performed as defined in the FSM definition for the existing combination of current state and input. This may involve a state transition.

| ° State test. The current state of the FSM is tested for equality or inequality with a specified value. No state change occurs.

| An FSM is represented by a state-transition matrix. The syntax of the state-transition matrix FSM definition is shown in Figure E-1. The column headings give the FSM state names, while the row headings name the inputs to the FSMs. The matrix elements--(row,column) intersections--define the state transitions and output actions.

| Horizontal lines are used to group input lines together to improve readability. Their location has no bearing on the FSM function. For compactness, mnemonic abbreviations are used in the matrices.

| An FSM comes into existence initialized to state 1. If another state is to be the initial state, the FSM is initialized explicitly by calling the FSM with an appropriate signal.

| Calling an FSM executes the FSM; i.e., an FSM action code is selected based on the current state of the FSM and the input line that is *true*. The input line evaluation uses the parameters or signal passed to the FSM. The FSM is scanned for a *true* input line from top to bottom of the matrix. No more than one input line in a matrix is *true* during a scan.

| If the next-state indicator is a number n, the FSM enters state n. If the next-state indicator is a cannot-occur indicator (/), this is an execution-time error; calls of the FSM cannot encounter this indicator because previous logic has filtered out the input for that state of the FSM.

| If no input line is *true*, the call of the FSM acts as if a no-state-change indicator (-) were encountered.

```
|    fname:
|     ___________________________ _______________ _________
|    |            STATE NAMES____ÿ |     snam      |         |
|    |                             |               |         |
|    |                             |  | . . . |    |         |
|    |                             |  |       |    |  . . .  |
|    | INPUTS     STATE NUMBERS__ÿ |     snum      |         |
|    |_____________________________|_______________|_________|
|    | ic [ ,ic ] . . .            |      ac       |         |
|    |                             |               |         |
| |  | ic [ ,ic ] . . .            |      ac       |         |  |
| |  |_____________________________|_______________|_________|  |
| |  | ic [ ,ic ] . . .            |      ac       |         |  |
| |  |         . . .               |               |         |  |
| |  | ic [ ,ic ] . . .            |      ac       |         |  |
| |  |                             |               |         |  |
|    |_____________________________|_______________|_________|
|    |
| |  |_____ ____________________________________________|  |
| |  | OUTPUT | FUNCTION                                |  |
| |  |  CODE  |                                         |  |
| |  |_______|_________________________________________|  |
| |  |  oc_1  | Output logic statements                 |  |
| |  |        |                                         |  |
| || |_______|_________________________________________| ||
| || |        |   . . .                                 | ||
| || |_______|_________________________________________| ||
| || |  oc_n  | Output logic statements                 | ||
| || |        |                                         | ||
| |  |        |                                         |  |
|    |_______|_________________________________________|


| Legend:


|   [   ] = optional parameter
|   fname = FSM name
|   snam = state name component
|   snum = state number
|   ic = input condition name
|   ac = action code
|   oc = output code


| An action code (ac) has the following possible syntax:


|   n[oc]   normal state transition to state n
|           (corresponding to some snum) and execution of
|           the function identified by oc


|   -[oc]   same-state transition--in the same state


|      /    "cannot occur" condition, no state change


|      >    error condition, no state change



| Figure  E-1. Syntax of an FSM State-Transition Matrix
```

ABBREVIATIONS Acronyms and Abbreviations

ACTLU activate logical unit

ACTPU activate physical unit

APPC advanced program-to-program communication

APPN Advanced Peer-to-Peer Networking

AUOWC Agent Unit-of-Work Correlator

BF boundary function

CD change direction

CNR carrier-to-noise ratio

CP control point

CPMS control point management services

CP-MSU control point management services unit

CRC cyclic redundancy check

CSMA/CD carrier-sense multiple-access with collision detection

CTS clear to send

DCE data circuit-terminating equipment

DLC data link control

DS distribution services

DSE data service element

DSU distribution service unit

DTE data terminal equipment

EB end bracket

EC engineering change

EN end node

| **ENCP** end node control point

EP entry point

FM function management

FP focal point

FRMR Frame Reject

FS file services

FSM finite-state machine

GDS general data stream

GMT Greenwich Mean Time

ID identification

IML initial microprogram load

IPL initial program load

| **IPM** ISOLATED PACING MESSAGE

| **IPR** ISOLATED PACING RESPONSE

| **ISR** intermediate session routing

LAN local-area network

LAPB link access procedure, balanced

LCS link connection subsystem

LEN low-entry networking

LLC logical link control

LMS local management services

LU logical unit

MAC medium-access control

MDS multiple-domain support

MDS-MU multiple-domain support message unit

MCF microcode fix

MES miscellaneous equipment specification

MS management services

MSU management services unit

MU message unit

MV major vector

NAU network addressable unit (in the context of subarea networking); *also* network accessible unit (in the context of APPN networking)

NETID network identification

NMVT Network Management Vector Transport

NN network node

| **NNCP** network node control point

NRM normal response mode

NS network services

NSD National Service Division

OEM original/other equipment manufacturer

| **PCID** procedure correlation identifier

PRID procedure-related identifier

PSID product set identification

PU physical unit

PUMS physical unit management services

QPI query product identification

RS ring station

| **RSCV** Route Selection control vector

RSP response

RTI routing and targeting instructions (GDS variable)

RTM response-time monitor

RTS request to send

RU request-response unit

SAA Systems Application Architecture

SDLC Synchronous Data Link Control

SF	subfield
SNA	Systems Network Architecture
SNACR	SNA Condition Report
SNA/DS	SNA/Distribution Services
SNA/FS	SNA/File Services
SNA/MS	SNA/Management Services
SNRM	Set Normal Response Mode
SOC	sphere of control
SSCP	system services control point
SV	subvector
TP	transaction program
UOW	unit of work
UPM	undefined protocol machine
VAR	value added remarketer
XID	Exchange Identification

GLOSSARY Glossary

This glossary defines terms used in this book. It should be considered an adjunct to the glossary in the following publications:

SNA Concepts and Products

SNA Type 2.1 Node Reference

Dictionary of Computing

A

accept. To cancel removability of a change (see Install). Releases resources required to maintain the removability of a change.

agent unit of work correlator. The agent unit of work correlator (X'1549') GDS variable, used for correlation of application program messages by multiple-domain support.

Alert. A record sent to an Alert focal point to communicate the existence of an Alert condition.

Alert condition. A problem or impending problem for which some or all of the process of problem determination/diagnosis/resolution is expected to require action at a problem management focal point.

assigned focal point. A focal point at which a node is included in a sphere of control by explicit definition at either the focal point or SOC node. When a node's inclusion in the sphere of control is entered at the focal point, the focal point is termed an *explicit* focal point for that SOC node. When a node's inclusion in the sphere of control is defined at the SOC node, the focal point is termed an *implicit* focal point for that node. Contrast with *default* focal point.

B

backup focal point. A focal point that provides management services support for a particular category for a node in the event of a communications failure with the primary focal point. Contrast with *primary* focal point. Both assigned focal points (explicit and implicit) and default focal points may have backup counterparts.

C

change. An alteration (addition, deletion, or modification) of one or more information system components, of one of the following types: hardware (may include microcode), or software (system or application program). The term *change* also refers to an SNA/File Services data object containing microcode, microcode customizing data, software, software customizing data, application programs data, procedures, or documentation.

change management. The process of planning (e.g., scheduling) and controlling (e.g., distributing, installing, and tracking) changes in an SNA network.

change management plan. A plan is a sequence of change management requests to be performed by a focal point with regard to one or more entry points. It is prepared by the network planner.

configuration management. The control of information necessary to identify both physical and logical information system resources and their relationship to one another.

corequisite change file. A change file that must be installed together with another change file, as part of the same process.

control point management services (CPMS). A component of a control point, consisting of management services function sets, that provides facilities to assist in performing problem management, performance and accounting management, change management, and configuration management.

control point management services unit (CP-MSU). The message unit that contains management services data and flows between management services function sets. This message unit is in general data stream (GDS) format. For more information on GDS format refer to *Systems Network Architecture Formats*. *See also* network management vector transport and management services unit.

D

default focal point. A focal point that provides management services support for those nodes that have not been assigned a focal point. The set of nodes in the sphere of control of a default focal point is not defined either at the focal point itself or at the SOC nodes. A default focal point exchanges MS Capabilities with all network nodes known to it, and only those nodes which have not established a relationship with another focal point accept the request. Contrast with *assigned* focal point.

default/replacement code point. A two-byte code point in which the first byte indexes text providing a high-level description of a type of condition and the second byte indexes text providing a more specific description of a particular condition of that type.

delayed Alert. An Alert reporting an Alert condition that prevented the Alert sender from sending that Alert to a control point. The fact that a delayed Alert is sent implies that the Alert condition it reports is no longer impacting availability. *See also* held Alert.

delete. In the context of SNA/MS, a protocol boundary verb to delete a change file.

E

engineering change (EC). A change to hardware or microcode to correct one or more design defects, or to modify the design. There are three types of ECs: *suffix, maintenance, and functional*.

entry point. An entry point is an SNA node that provides distributed network management support. It may be a type 2.0 or type 2.1 node. It sends SNA-formatted network management data about itself and the resources it controls to a focal point for centralized processing, and it receives and executes focal point initiated requests to manage and control its resources.

error. The smallest detectable anomaly or exception that can occur in an information system. Errors may be caused by hardware, software, microcode, media, or external causes, e.g., people or environmental abnormalities.

error condition. A state whose existence is indicated by (1) the occurrence of an error, and, possibly, (2) additional specific events, often errors resulting from attempts to recover from the initial error. Associated with each type of error is a recovery procedure to be followed when an error of that type occurs. The results of this recovery procedure determine whether the error condition is to be classified as *recoverable* or *irrecoverable*. An error condition is classified as *irrecoverable* if the operation being attempted when the error was detected is not completed without the use of recovery techniques external both to the component detecting the error, e.g., a hardware adapter, and to its controlling component, e.g., microcode controlling the adapter. A *recoverable* error condition is one in which the operation being attempted at the time the error was detected is completed by using a recovery procedure internal either to the component detecting the error or to its controlling component.

explicit focal point. An assigned focal point for which the set of nodes to be included in its sphere of control is defined locally. An explicit focal point initiates the MS Capabilities exchange that brings a node into its sphere of control. Contrast with *implicit* focal point.

F

feature change. A change to hardware (or microcode) to add new capabilities. Sometimes called a *sales change*.

firmware. *See* microcode.

focal point. A focal point is an entry point that provides centralized management and control for other entry points within its sphere of control for one or more network management categories.

function set. *See* management services function set.

function set group. *See* management services function set group.

functional EC. An EC that enhances reliability/availability/serviceability (RAS) or adds new capabilities (in addition to correcting one or more design defects).

G

general function set. *See* general management services function set.

general management services function set. A management services function set that transports management services data or provides a database function for this data in a control point.

H

hardware. Physical equipment used in data processing, as opposed to programs, procedures, rules, and associated documentation.

held Alert. An Alert that an Alert sender was unable to send to a control point immediately. The sending of a held Alert implies nothing about whether the Alert condition it reports is still impacting availability. *See also* delayed Alert.

I

impending problem. An error condition that has the potential for resulting in a loss of availability of a system resource to an end user.

implicit focal point. An assigned focal point for which the nodes to be included in its sphere of control are defined at the SOC nodes. The MS Capabilities exchange that brings a node into the sphere of control of an implicit focal point is initiated by the SOC node. Contrast with *explicit* focal point.

indicated resource. The SNA or non-SNA entity most closely associated with an Alert condition.

install. In the context of SNA/MS, to alter all network components necessary to effect a change. A change may be installed removably or nonremovably.

L

local management services (LMS). Those functions within various node components which provide for the collection and exchange of management data with the management services component of the same node.

M

machine level control (MLC). A complete history of engineering and feature changes to a hardware product. This term is often misused to describe the entire range of activities involved in processing a user's order for a machine.

maintenance EC. An EC correcting one or more hardware or microcode design defects. Usually corrects more defects than a suffix EC and has undergone more testing.

management services. *See* SNA/MS.

management services function set. A collection of specific services that together perform an overall management services function for a physical unit or a control point (e.g., Alert). Function sets are composed of a base and optional subsets.

management services function set group. The components or subset of components providing the function described by the corresponding function set.

management services unit (MSU). A generic term for major-vector encoded management services data, regardless of the method used to transport the data. Thus, management services unit includes major vectors transported within the network management vector transport (NMVT) the control point management services unit (CP-MSU), or the multiple-domain support message unit (MDS-MU).

microcode. Microinstructions used in a product as an alternative to hard-wired circuitry to implement functions of a processor or other system component. (*See* microinstruction.)

microcode fix (MCF). A change to correct a single microcode design defect. Higher quality than a *patch*, and intended for broad distribution. Considered functionally equivalent to a software PTF.

microinstruction. An instruction that controls data flow and sequencing in a processor at a more fundamental level than a machine instruction.

miscellaneous equipment specification (MES). A request from marketing that results in a feature change. Also, the procedure to install the feature change.

multiple-domain support (MDS). A technique for transporting management services data between management services function sets over LU-LU and CP-CP sessions. *See also* multiple-domain support message unit

multiple-domain support message unit (MDS-MU). The message unit that contains management services data and flows between management services function sets over the LU-LU and CP-CP sessions used by multiple-domain support. This message unit, as well as the actual management services data that it contains, is in general data stream (GDS) format. *See also* network management vector transport, control point management services unit, and management services unit.

N

nested focal point. A focal point that is itself in the sphere of control of another focal point. Contrast with *nesting* focal point.

nesting focal point. A focal point that has another focal point in its sphere of control (although this may not be explicitly known to the nesting focal point). Contrast with *nested* focal point.

network management vector transport (NMVT). A management services RU that flows over an active session between the physical unit management services component and the control point management services component of a control point. *See also* control point management services unit and management services unit.

network operator. A person or program responsible for controlling the operation of all or part of a network.

network management. The process of planning, organizing, and controlling a communication-oriented data processing or information system.

network planner. A person or program responsible for planning the configuration (and changes) of all or part of a network.

P

patch. A change to correct a single defect, in hardware, microcode, or software. Temporary in nature, usually applied in *corrective mode*, i.e. to solve a problem that has actually been experienced at the site installing the patch.

performance and accounting management. The process of quantifying, measuring, reporting, and controlling the responsiveness, availability, utilization, and costs of an information system.

physical unit management services (PUMS). A component of the PU, consisting of management services function sets, that provides management services for a node and its associated resources. PUMS is dependent on control point management services (CPMS) for providing some information and control (e.g., configuration knowledge of the connection between itself and its control points). Capabilities provided by the PUMS include receiving requests from a CPMS, passing these requests to the appropriate local management services (LMS) for processing, and sending data to a CPMS.

prerequisite change file. A change file that must be installed prior to the subsequent installation of another change file.

primary focal point. A focal point understood to be the preferred source of management services support for a particular category. Contrast with *backup* focal point.

problem. An error condition resulting in a loss of availability of a system resource to an end user.

problem bypass and recovery. The process of providing partial or complete circumvention of a problem, usually prior to the final resolution of the problem.

problem determination. The process of isolating a problem to the failing hardware device, software product, microcode component, medium, or external cause in order to identify the organization responsible for problem diagnosis.

problem diagnosis. The process of determining the precise cause of a hardware, software, microcode, medium, or externally caused problem and the precise action required to resolve the problem.

problem management. The management discipline that manages a problem from its detection through it final resolution. Problem management is composed of the following:

Problem determination Problem diagnosis Problem bypass and recovery Problem resolution Problem tracking and control

problem resolution. The process of taking action to correct the error condition detected as a problem or impending problem.

problem tracking and control. The process of tracking a problem until its resolution.

production. An installation option that results in altered components being installed for production use. Such components will be activated for any activation request pertaining to them.

program. A set of actions or instructions that a machine is capable of interpreting and executing. In the context of SNA/MS, there are two types of programs: software and microcode.

| **pseudo focal point**. An APPN network node that performs the forwarding | functions of a nested focal point to a nesting subarea focal point when an | APPN focal point is not available.

Q

quiescing. The process of bringing a device or system to a halt by rejection of new requests for work.

R

request for price quotation (RPQ). A feature change that is created specially for a limited set of users. Popular RPQs become orderable feature changes. RPQs are assigned numbers.

remove. In the context of SNA/MS, to return all components previously altered in connection with a change to their condition prior to the installation of the change.

retrieve. In the context of SNA/MS, a protocol boundary verb to retrieve a change file.

Routing Element Name (REN). In SNA/DS, the second part of the two-part DSU name. The routing element name (REN) must be unique within its particular group (RGN). This is typically, but not necessarily, the LU name.

Routing Group Name (RGN). In SNA/DS, the first part of the two-part DSU name. This is typically, but not necessarily, the network ID.

S

send. In the context of SNA/MS, a protocol boundary verb to send a change file.

SOC node. A node directly in the sphere of control of a focal point. A SOC node has exchanged MS Capabilities with its focal point. An APPN end node may optionally be a SOC node, if it supports the function to exchange | MS Capabilities. Support for being a SOC node is a base requirement for | APPN network nodes.

SNA/Management Services (SNA/MS). The services provided to assist in management of SNA networks.

software. Programs, procedures, rules, and any associated documentation pertaining to the operation of a computer system.

In the context of SNA/MS, a special kind of software, microcode, is defined (*see* microcode, program.)

source update. A change to programming source statements (prior to assembly).

sphere of control (SOC). For each category of network management focal point, the user-specified set of nodes for which a focal point provides management services. For example, the nodes that rely on a particular focal point to process problem determination data are in that problem management focal point's sphere of control. A node may have different focal points for different categories of management data (e.g., a problem management focal point may be different from a performance management focal point for a given domain).

specialized function set. *See* specialized management services function set.

specialized management services function set. A management services function set that analyzes data for a particular management services function. In a

control point, it also interacts with a network operator.

suffix EC. An EC correcting one or more hardware or microcode design defects. (*See* Maintenance EC.)

T

test. In the context of SNA management services, to exercise as many altered components as feasible to observe whether a change has produced the desired result (typically, an absence of error conditions).

trial. An installation option that results in altered components superseding components installed for production use. Such components can only be activated if use of both trial and production components is specified in the activation request.

U

unit of work correlator. *see* agent unit of work correlator

V

vital product data (VPD). Information imbedded in a product that is in machine-readable form and will be used to uniquely identify a product or product instance or to provide other information.

Note: The information is *vital* only in the context of how it is used (for example, to determine service entitlements, or by specific SNA management services categories like problem or change management). There may be other data describing the product kept by a configuration management focal point that is equally *vital* to some category, but is not kept by the product and so does not fall within this definition.

INDEX Index

IBM Library Server Print Preview

DOCNUM = SC30-3346-05
DATETIME = 09/27/93 17:10:48
BLDVERS = 1.2
TITLE = SNA/Management Services Reference
AUTHOR =
COPYR = © Copyright IBM Corp. 1986, 1993
PATH = /home/webapps/epubs/htdocs/book