SNMP MIB extension for Multiprotocol Interconnect over X.25

January 11, 1993

Dean D. Throop

Data General Corporation
62 Alexander Dr.
Research Triangle Park, NC 27709

throop@dg-rtp.dg.com

1. Status of this Memo

This document is an Internet Draft. Internet Drafts are working documents of the Internet Engineering Task Force (IETF), its Areas, and its Working Groups. Note that other groups may also distribute working documents as Internet Drafts).

Internet Drafts are draft documents valid for a maximum of six months. Internet Drafts may be updated, replaced, or obsoleted by other documents at any time. It is not appropriate to use Internet Drafts as reference material or to cite them other than as a "working draft" or "work in progress."

Please check the I-D abstract listing contained in each Internet Draft directory to learn the current status of this or any other Internet Draft.

This document was produced by the x25mib working group in conjunction with the Large Public Data Networks Working Group. Eventually this document will be submitted to the RFC editor as an extension to the SNMP MIB. Distribution of this memo is unlimited. Please send comments to the x25mib working group at:

x25mib@dg-rtp.dg.com

1.1. Revision History

January 1993

The January 1993 release (Editor's Internal Reference Number 2.13) incorporated the following comments from the SNMP directorate:

The overview section was expanded to have two subsections to better state the relationship this MIB has with RFC 1356. The MIB Context subsection was added.

Object grouping was clarified by introducing the mioxPle and mioxPeer object identifiers and text was added to identify all objects as required.

A conformance statement was added for the mioxPeerStatus object and the mioxPeerX25CircuitId object.

The descriptions of mioxPleRefusedConnections, mioxPleEnAddrToX121LkupFlrs, mioxPleX121ToEnAddrLkupFlrs, mioxPleCollisionRetryTimer, mioxPeerX25CallParamId, mioxPeerX25CircuitId, mioxPeerEncType were changed to improve clarity.

Obsolete references were updated to new references as appropriate.

Some minor typographical errors were corrected.

June 1992

The June 1992 release (Editor's Internal Reference Number 2.4) incorporated several comments of the mailing list. These changes are as follows:

The range and description of mioxPleMaxCircuits was expanded.

The following objects were added: mioxPleEnAddrToX121LkupFlrTime, mioxPleX121ToEnAddrLkupFlrTime, mioxPleQbitFailures, mioxPleQbitFailureRemoteAddress, mioxPleQbitFailureTime, mioxPleMinimumOpenTimer, mioxPleInactivityTimer, mioxPleHoldDownTimer, mioxPleCollisionRetryTimer.

The following objects were deleted: mioxPeerMinimumOpenTimer, mioxPeerHoldDownTimer, mioxPeerQbitErrors.

May 1992

The April 1992 release (Editor's Internal Reference Number 2.2) incorporated the comments of the March working group meeting. These changes are as follows:

The overview section was expanded to better explain the relationship between the objects defined in this MIB and other MIB extensions.

The name of the MIB was change from IP over X.25 to Multiprotocol Interconnect over X.25.

All references to IP addresses were changed to Encapsulation Addresses.

All references to X25Address were changed to X121Address.

The ioxInfoTable was renamed the mioxPleTable because it contains information relative to a PLE.

The ioxConTable was renamed the mioxPeerTable.

The mioxPeerStatus object was added.

The mioxPeerMaxCircuits object was added.

The mioxPeerIfIndex object was added.

The mioxPeerQbitErrors object was added.

The mioxPeerConnectSeconds object was added.

The mioxPeerDescr object was added.

The mioxPeerEncTable was added.

Some objects were re-ordered and some descriptions were expanded.

February 1992

The February 1992 release (Editor's Internal Reference Number 1.17) made the following changes:

- The ioxInfoDefaultParamId object was added to the ioxInfoTable.

- The ioxConX25Channel object was deleted from the ioxConTable and replaced with the ioxConX25CircuitId object.

- The ioxConX25Address object was added to the ioxConTable.

- The ioxConX25FcltyIndex, ioxConX25fcltyCcittIndex, and ioxConX25CallParamIndex objects were deleted from the ioxConTable and replaced with the ioxConX25CallParamId object.

- The ioxConEncapsulation and ioxConHoldDownTimer objects were added to the ioxConTable.

- The text at the beginning of the document was changed.

- The references were changed to match the new text.

October 1991

The October 91 revision of this document (Editor's internal reference 1.14) had the following changes:

- The object ioxInfoAddressXlationFailures was split into ioxInfoIpToX25LookupFailures and ioxInfoX25ToIpLookupFailures. The objects ioxInfoLastFailedIpAddress and ioxInfoLastFailedX25Address were added. This provides information to allow for correction of errors as well as detection.

- The objects ioxConX25FcltyIndex, ioxConX25fcltyCcittIndex, and ioxConX25CallParamIndex were added.

June 1991

The June revision of this document was synthesized from various ideas on how to manage IP over X.25. This initial release of this document serves as a basis of discussion in the X25mib working group.

2. Abstract

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in TCP/IP-based internets. In particular, it defines objects for managing Multiprotocol Interconnect (including IP) traffic carried over X.25. The objects defined here, along with the objects in the "SNMP MIB extension for the Packet Layer of X.25"[10], "SNMP MIB extension for LAPB"[9], and the "Definitions of Managed Objects for RS-232-like Hardware Devices"[8], combine to allow management of the traffic over an X.25 protocol stack.

This memo does not specify a standard for the Internet community.

3. The Network Management Framework

The Internet-standard Network Management Framework consists of three components. These components give the rules for defining objects, the definitions of objects, and the protocol for manipulating objects.

The network management framework structures objects in an abstract information tree. The branches of the tree name objects and the leaves of the tree contain the values manipulated to effect management. This tree is called the Management Information Base or MIB. The concepts of this tree are given in RFC 1155 "The Structure of Management Information" or SMI [1]. The SMI defines the trunk of the tree and the types of objects used when defining the leaves. RFC 1212, "Towards Concise MIB Definitions" [4], defines a more concise description mechanism that preserves all the principals of the SMI.

The core MIB definitions for the Internet suite of protocols can be found in RFC 1156 [2] "Management Information Base for Network Management of TCP/IP-based internets". RFC 1213 [5] defines MIB-II, an evolution of MIB-I with changes to incorporate implementation experience and new operational requirements.

RFC 1157 [3] defines the SNMP protocol itself. The protocol defines how to manipulate the objects in a remote MIB.

The tree structure of the MIB allows new objects to be defined for the purpose of experimentation and evaluation.

4. Objects

The definition of an object in the MIB requires an object name and type. Object names and types are defined using the subset of Abstract Syntax Notation One (ASN.1) [6] defined in the SMI [1]. Objects are named using ASN.1 object identifiers, administratively assigned names, to specify object types. The object name, together with an optional object instance, uniquely identifies a specific instance of an object. For human convenience, we often use a textual string, termed the OBJECT DESCRIPTOR, to also refer to objects.

Objects also have a syntax that defines the abstract data structure corresponding to that object type. The ASN.1 language [6] provides the primitives used for this purpose. The SMI [1] purposely restricts the ASN.1 constructs which may be used for simplicity and ease of implementation. The encoding of an object type simply describes how to represent an object using ASN.1 encoding rules [7], for purposes of dealing with the SNMP protocol.

4.1. Format of Definitions

Section 6 contains the specification of all object types contained in this MIB module. The object types are defined using the conventions defined in the SMI, as amended by the extensions specified in "Towards Concise MIB Definitions" [4].

5. Overview

5.1. MIB Context

Instances of the objects defined below provide management information for Multiprotocol Interconnect traffic on X.25 as defined in RFC 1356 [11]. That RFC describes how X.25 can be used to exchange IP or network level protocols. The multiprotocol packets (IP, CLNP, ES-IS, or SNAP) are encapsulated in X.25 frames for transmission between nodes. All nodes that implement RFC 1356 must implement this MIB.

The objects in this MIB apply to the software in the node that manages X.25 connections and performs the protocol encapsulation. A node in this usage maybe the end node source or destination host for the packet, or it may be a router or bridge responsible for forwarding the packet. Since RFC 1356 requires X.25, nodes that implement RFC 1356 must also implement the X.25 MIB, RFC 1382.

This MIB only applies to Multiprotocol Interconnect over X.25 service. It does not apply to other software that may also use X.25 (for example PAD). Thus the presence, absence, or operation of such software will not directly affect any of these objects. (However connections in use by that software will appear in the X.25 MIB).

5.2. Structure of MIB objects

The objects of this MIB are organized into three tables: the mioxPleTable, the mioxPeerTable, and the mioxPeerEncTable. All objects in all tables are mandatory for conformance with this MIB.

The mioxPleTable defines information relative to an interface used to carry Multiprotocol Interconnect traffic over X.25. Such interfaces are identified by an ifType object in the Internet-standard MIB [5] of ddn-x25 or rfc877-x25. Interfaces of type ddn-x25 have a self contained algorithm for translating between IP addresses and X.121 addresses. Interfaces of type rfc877-x25 do not have such an algorithm. Note that not all X.25 Interfaces will be used to carry Multiprotocol Interconnect traffic. Those interfaces not carrying such traffic will not have entries in the mioxPleTable. The entries in the mioxPleTable are only for

interfaces that do carry Multiprotocol Interconnect traffic over X.25. Interfaces that do have entries in the mioxPleTable have mioxPleIndex object instance identifiers that match the values of their respective ifIndex object instance identifiers. This relationship allows the value of an index object instance from the mioxPleTable below to be directly used to identify the corresponding instances of the objects for the interface to X.25.

The mioxPeerTable contains information needed to contact an X.25 **Peer to exchange packets. This includes information such** as the X.121 address of the peer and a pointer to the X.25 call parameters needed to place the call. The instance identifiers used for the objects in this table are independent of any interface or other tables defined outside this MIB. This table contains the ifIndex value of the X.25 interface to use to call a peer.

The mioxPeerEncTable contains information about the encapsulation type used to communicate with a peer. This table is an extension of the mioxPeerTable in its instance identification. Each entry in the mioxPeerTable may have zero or more entries in this table. This table will not have any entries that do not have correspondent entries in mioxPeerTable.

6. Definitions

```
RFCmiox-MIB DEFINITIONS ::= BEGIN
        -- (Editors Internal Reference 2.42)

IMPORTS
        Counter,
        experimental,
        TimeTicks
                FROM RFC1155-SMI
        OBJECT-TYPE
                FROM RFC-1212
        DisplayString
                FROM RFC1213-MIB
        X121Address
                FROM RFC1382-MIB
        PositiveInteger
                FROM RFC1381-MIB;

                -- IP over X.25 MIB

                -- DO NOT USE THIS EXPERIMENTAL NUMBER.
miox    OBJECT IDENTIFIER ::= { experimental 12345678 }
                -- This experimental number is an
                -- invalid placeholder.
                -- DO NOT USE THIS EXPERIMENTAL NUMBER.

mioxPle         OBJECT IDENTIFIER ::= { miox 1 }
mioxPeer        OBJECT IDENTIFIER ::= { miox 2 }

-- ###############################################################
--              Ple Table
-- ###############################################################

-- Systems that implement RFC 1356 must also implement
-- all objects in this group.

mioxPleTable    OBJECT-TYPE
        SYNTAX  SEQUENCE OF MioxPleEntry
        ACCESS  not-accessible
        STATUS  mandatory
        DESCRIPTION
                "This table contains information relative to
                an interface to an X.25 Packet Level Entity
```

```
        (PLE)."
        ::= { mioxPle 1 }

mioxPleEntry    OBJECT-TYPE
        SYNTAX  MioxPleEntry
        ACCESS  not-accessible
        STATUS  mandatory
        DESCRIPTION
                "These objects manage the encapsulation of
                other protocols within X.25."
        INDEX { mioxPleIndex }
        ::= { mioxPleTable 1 }

MioxPleEntry ::= SEQUENCE {
        mioxPleIndex
                PositiveInteger,
        mioxPleMaxCircuits
                INTEGER,
        mioxPleRefusedConnections
                Counter,
        mioxPleEnAddrToX121LkupFlrs
                Counter,
        mioxPleLastFailedEnAddr
                OCTET STRING,
        mioxPleEnAddrToX121LkupFlrTime
                TimeTicks,
        mioxPleX121ToEnAddrLkupFlrs
                Counter,
        mioxPleLastFailedX121Address
                X121Address,
        mioxPleX121ToEnAddrLkupFlrTime
                TimeTicks,
        mioxPleQbitFailures
                Counter,
        mioxPleQbitFailureRemoteAddress
                X121Address,
        mioxPleQbitFailureTime
                TimeTicks,
        mioxPleMinimumOpenTimer
                PositiveInteger,
        mioxPleInactivityTimer
                PositiveInteger,
        mioxPleHoldDownTimer
                PositiveInteger,
        mioxPleCollisionRetryTimer
```

```
        PositiveInteger,
        mioxPleDefaultPeerId
                OBJECT IDENTIFIER
        }

mioxPleIndex    OBJECT-TYPE
        SYNTAX  PositiveInteger
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "An index value that uniquely identifies the
                interface to X.25 used to send and receive
                encapsulated Packets.  This value will match
                the ifIndex entry identifying the MIB-II
                ifTable objects for that interface.  This
                value ranges between 1 and ifNumber."
        ::= { mioxPleEntry 1 }

mioxPleMaxCircuits OBJECT-TYPE
        SYNTAX  INTEGER (0..2147483647)
        ACCESS  read-write
        STATUS  mandatory
        DESCRIPTION
                "The maximum number of X.25 circuits that
                can be open at one time for this interface.
                A value of zero indicates the interface will
                not allow any additional circuits (as it may
                soon be shutdown).  A value of 2147483647
                allows an unlimited number of circuits."
        ::= { mioxPleEntry 2 }

mioxPleRefusedConnections OBJECT-TYPE
        SYNTAX  Counter
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "The number of X.25 calls from a remote
                systems to this system that were cleared by
                this system.  The interface instance should
                be that of the X.25 interface in call came
                in on."
        ::= { mioxPleEntry 3 }

mioxPleEnAddrToX121LkupFlrs OBJECT-TYPE
        SYNTAX  Counter
```

```
      ACCESS    read-only
        STATUS  mandatory
        DESCRIPTION
                "The number of times a translation from an
                Encapsulated Address to an X.121 address
                failed to find a corresponding X.121
                address.  Encapsulated addresses can be
                looked up in the mioxPeerTable or translated
                via an algorithm as for the DDN.  Addresses
                that are successfully recognized do not
                increment this counter.  Addresses that are
                not recognized (reflecting an abnormal
                packet delivery condition) increment this
                counter.

                If an address translation fails, it may be
                difficult to determine which PLE entry
                should count the failure.  In such cases the
                first likely entry in this table should be
                selected.  Agents should record the failure
                even if they are unsure which PLE should be
                associated with the failure."
        ::= { mioxPleEntry 4 }

mioxPleLastFailedEnAddr OBJECT-TYPE
        SYNTAX  OCTET STRING (SIZE(2..128))
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "The last Encapsulated address that failed
                to find a corresponding X.121 address and
                caused mioxPleEnAddrToX121LkupFlrs to be
                incremented.  The first octet of this object
                contains the encapsulation type, the
                remaining octets contain the address of that
                type that failed.  Thus for an IP address,
                the length will be five octets, the first
                octet will contain 204 (hex CC), and the
                last four octets will contain the IP
                address.  For a snap encapsulation, the
                first byte would be 128 (hex 80) and the
                rest of the octet string would have the snap
                header."
        ::= { mioxPleEntry 5 }
```

```
mioxPleEnAddrToX121LkupFlrTime OBJECT-TYPE
        SYNTAX  TimeTicks
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "The most recent value of sysUpTime when the
                translation from an Encapsulated Address to
                X.121 address failed to find a corresponding
                X.121 address."
        ::= { mioxPleEntry 6 }


mioxPleX121ToEnAddrLkupFlrs OBJECT-TYPE
        SYNTAX  Counter
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "The number of times the translation from an
                X.121 address to an Encapsulated Address
                failed to find a corresponding Encapsulated
                Address.  Addresses successfully recognized
                by an algorithm do not increment this
                counter.  This counter reflects the number
                of times call acceptance encountered the
                abnormal condition of not recognizing the
                peer."
        ::= { mioxPleEntry 7 }

mioxPleLastFailedX121Address OBJECT-TYPE
        SYNTAX  X121Address
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "The last X.121 address that caused
                mioxPleX121ToEnAddrLkupFlrs to increase."
        ::= { mioxPleEntry 8 }

mioxPleX121ToEnAddrLkupFlrTime OBJECT-TYPE
        SYNTAX  TimeTicks
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "The most recent value of sysUpTime when the
                translation from an X.121 address to an
                Encapsulated Address failed to find a
```

```
        corresponding Encapsulated Address."
        ::= { mioxPleEntry 9 }

mioxPleQbitFailures OBJECT-TYPE
        SYNTAX  Counter
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "The number of times a connection was closed
                because of a Q-bit failure."
        ::= { mioxPleEntry 10 }

mioxPleQbitFailureRemoteAddress OBJECT-TYPE
        SYNTAX  X121Address
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "The remote address of the most recent
                (last) connection that was closed because of
                a Q-bit failure."
        ::= { mioxPleEntry 11 }

mioxPleQbitFailureTime OBJECT-TYPE
        SYNTAX  TimeTicks
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "The most recent value of sysUpTime when a
                connection was closed because of a Q-bit
                failure.  This will also be the last time
                that mioxPleQbitFailures was incremented."
        ::= { mioxPleEntry 12 }

mioxPleMinimumOpenTimer OBJECT-TYPE
        SYNTAX  PositiveInteger
        ACCESS  read-write
        STATUS  mandatory
        DESCRIPTION
                "The minimum time in milliseconds this
                interface will keep a connection open before
                allowing it to be closed.  A value of zero
                indicates no timer."
        DEFVAL { 0 }
        ::= { mioxPleEntry 13 }
```

```
mioxPleInactivityTimer OBJECT-TYPE
        SYNTAX  PositiveInteger
        ACCESS  read-write
        STATUS  mandatory
        DESCRIPTION
                "The amount of time time in milliseconds
                this interface will keep an idle connection
                open before closing it.  A value of
                2147483647 indicates no timer."
        DEFVAL { 10000 }
        ::= { mioxPleEntry 14 }

mioxPleHoldDownTimer    OBJECT-TYPE
        SYNTAX  PositiveInteger
        ACCESS  read-write
        STATUS  mandatory
        DESCRIPTION
                "The hold down timer in milliseconds.  This
                is the minimum amount of time to wait before
                trying another call to a host that was
                previously unsuccessful.  A value of
                2147483647 indicates the host will not be
                retried."
        DEFVAL { 0 }
        ::= { mioxPleEntry 15 }

mioxPleCollisionRetryTimer OBJECT-TYPE
        SYNTAX  PositiveInteger
        ACCESS  read-write
        STATUS  mandatory
        DESCRIPTION
                "The Collision Retry Timer in milliseconds.
                The time to delay between call attempts when
                the maximum number of circuits is exceeded
                in a call attempt."
        DEFVAL { 0 }
        ::= { mioxPleEntry 16 }


mioxPleDefaultPeerId OBJECT-TYPE
        SYNTAX  OBJECT IDENTIFIER
        ACCESS  read-write
        STATUS  mandatory
        DESCRIPTION
                "This identifies the instance of the index
```

```
        in the mioxPeerTable for the default
                parameters to use with this interface.

                The entry identified by this object may have
                a zero length Encapsulation address and a
                zero length X.121 address.

                These default parameters are used with
                connections to hosts that do not have
                entries in the mioxPeerTable.  Such
                connections occur when using ddn-x25 IP-X.25
                address mapping or when accepting
                connections from other hosts not in the
                mioxPeerTable.

                The mioxPeerEncTable entry with the same
                index as the mioxPeerTable entry specifies
                the call encapsulation types this PLE will
                accept for peers not in the mioxPeerTable.
                If the mioxPeerEncTable doesn't contain any
                entries, this PLE will not accept calls from
                entries not in the mioxPeerTable."
        ::= { mioxPleEntry 17 }


-- ################################################################
--              Peer Table
-- ################################################################

-- Systems that implement RFC 1356 must also implement
-- all objects in this group.

mioxPeerTable OBJECT-TYPE
        SYNTAX  SEQUENCE OF MioxPeerEntry
        ACCESS  not-accessible
        STATUS  mandatory
        DESCRIPTION
                "This table contains information about the
                possible peers this machine may exchange
                packets with."
        ::= { mioxPeer 1 }

mioxPeerEntry OBJECT-TYPE
        SYNTAX  MioxPeerEntry
        ACCESS  not-accessible
```

```
      STATUS    mandatory
        DESCRIPTION
                "Per peer information."
        INDEX { mioxPeerIndex   }
        ::= { mioxPeerTable 1 }


MioxPeerEntry ::= SEQUENCE {
        mioxPeerIndex
                PositiveInteger,
        mioxPeerStatus
                INTEGER,
        mioxPeerMaxCircuits
                PositiveInteger,
        mioxPeerIfIndex
                PositiveInteger,
        mioxPeerConnectSeconds
                Counter,
        mioxPeerX25CallParamId
                OBJECT IDENTIFIER,
        mioxPeerEnAddr
                OCTET STRING,
        mioxPeerX121Address
                X121Address,
        mioxPeerX25CircuitId
                OBJECT IDENTIFIER,
        mioxPeerDescr
                DisplayString
        }

mioxPeerIndex   OBJECT-TYPE
        SYNTAX  PositiveInteger
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "An index value that distinguished one entry
                from another.  This index is independent of
                any other index."
        ::= { mioxPeerEntry 1 }

-- Systems can claim conformance with this MIB without
-- implementing sets to mioxPeerStatus with a value of
-- clearCall or makeCall.
-- All other defined values must be accepted.
-- Implementors should realize that allowing these values
```

```
-- provides richer system management and implementations
-- are encouraged to accept these values.
mioxPeerStatus OBJECT-TYPE
        SYNTAX  INTEGER {
                        valid (1),
                        createRequest (2),
                        underCreation (3),
                        invalid (4),
                        clearCall (5),
                        makeCall (6)
                        }
        ACCESS  read-write
        STATUS  mandatory
        DESCRIPTION
                "This reports the status of a peer entry.

                A value of valid indicates a normal entry
                that is in use by the agent.  A value of
                underCreation indicates a newly created
                entry which isn't yet in use because the
                creating management station is still setting
                values.

                The value of invalid indicates the entry is
                no longer in use and the agent is free to
                delete the entry at any time.  A management
                station is also free to use an entry in the
                invalid state.

                Entries are created by setting a value of
                createRequest.  Only non-existent or invalid
                entries can be set to createRequest.  Upon
                receiving a valid createRequest, the agent
                will create an entry in the underCreation
                state.  This object can not be set to a
                value of underCreation directly, entries can
                only be created by setting a value of
                createRequest.  Entries that exist in other
                than the invalid state can not be set to
                createRequest.

                Entries with a value of underCreation are
                not used by the system and the management
                station can change the values of other
                objects in the table entry.  Management
```

stations should also remember to configure values in the mioxPeerEncTable with the same peer index value as this peer entry.

An entry in the underCreation state can be set to valid or invalid. Entries in the underCreation state will stay in that state until 1) the agent times them out, 2) they are set to valid, 3) they are set to invalid. If an agent notices an entry has been in the underCreation state for an abnormally long time, it may decide the management station has failed and invalidate the entry. A prudent agent will understand that the management station may need to wait for human input and will allow for that possibility in its determination of this abnormally long period.

Once a management station has completed all fields of an entry, it will set a value of valid. This causes the entry to be activated.

Entries in the valid state may also be set to makeCall or clearCall to make or clear X.25 calls to the peer. After such a set request the entry will still be in the valid state. Setting a value of makeCall causes the agent to initiate an X.25 call request to the peer specified by the entry. Setting a value of clearCall causes the agent to initiate clearing one X.25 call present to the peer. Each set request will initiate another call or clear request (up to the maximum allowed); this means that management stations that fail to get a response to a set request should query to see if a call was in fact placed or cleared before retrying the request. Entries not in the valid state can not be set to makeCall or clearCall.

The values of makeCall an clearCall provide for circuit control on devices which perform

```
        Ethernet Bridging using static circuit
                assignment without address recognition;
                other devices which dynamically place calls
                based on destination addresses may reject
                such requests.

                An agent that (re)creates a new entry
                because of a set with createRequest, should
                also (re)create a mioxPeerEncTable entry
                with the mioxPeerEncPeerIndex that matches
                the mioxPeerIndex, a mioxPeerEncEncIndex of
                1, and a mioxPeerEncType of 204 (hex CC)."
        ::= { mioxPeerEntry 2 }

mioxPeerMaxCircuits OBJECT-TYPE
        SYNTAX  PositiveInteger
        ACCESS  read-write
        STATUS  mandatory
        DESCRIPTION
                "The maximum number of X.25 circuits allowed
                to this peer."
        DEFVAL { 1 }
        ::= { mioxPeerEntry 3 }

mioxPeerIfIndex OBJECT-TYPE
        SYNTAX  PositiveInteger
        ACCESS  read-write
        STATUS  mandatory
        DESCRIPTION
                "The value of the ifIndex object for the
                interface to X.25 to use to call the peer."
        DEFVAL { 1 }
        ::= { mioxPeerEntry 4 }

mioxPeerConnectSeconds OBJECT-TYPE
        SYNTAX  Counter
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "The number of seconds a call to this peer
                was active.  This counter will be
                incremented by one for every second a
                connection to a peer was open.  If two calls
                are open at the same time, one second of
                elapsed real time will results in two
```

```
        seconds of connect time."
        ::= { mioxPeerEntry 5 }

mioxPeerX25CallParamId OBJECT-TYPE
        SYNTAX  OBJECT IDENTIFIER
        ACCESS  read-write
        STATUS  mandatory
        DESCRIPTION
                "The instance of the index object in the
                x25CallParmTable for the X.25 call
                parameters used to communicate with the
                remote host.  The well know value {0 0}
                indicates no call parameters specified."
        DEFVAL { {0 0} }
        ::= { mioxPeerEntry 6 }

mioxPeerEnAddr  OBJECT-TYPE
        SYNTAX  OCTET STRING (SIZE (0..128))
        ACCESS  read-write
        STATUS  mandatory
        DESCRIPTION
                "The Encapsulation address of the remote
                host mapped by this table entry.  A length
                of zero indicates the remote IP address is
                unknown or unspecified for use as a PLE
                default.

                The first octet of this object contains the
                encapsulation type, the remaining octets
                contain the address of that type that
                failed.  Thus for an IP address, the length
                will be five octets, the first octet will
                contain 204 (hex CC), and the last four
                octets will contain the IP address.  For a
                snap encapsulation, the first byte would be
                128 (hex 80) and the rest of the octet
                string would have the snap header."
        DEFVAL { ''h }
        ::= { mioxPeerEntry 7 }

mioxPeerX121Address OBJECT-TYPE
        SYNTAX  X121Address
        ACCESS  read-write
        STATUS  mandatory
        DESCRIPTION
```

```
        "The X.25 address of the remote host mapped
                by this table entry.  A zero length string
                indicates the X.25 address is unspecified
                for use as the PLE default."
        DEFVAL { ''h }
        ::= { mioxPeerEntry 8 }

-- Systems can claim conformance to this MIB without
-- implementing sets to mioxPeerx25CircuitId.
-- However systems that use PVCs with RFC1356
-- are encouraged to implement sets.
mioxPeerX25CircuitId OBJECT-TYPE
        SYNTAX  OBJECT IDENTIFIER
        ACCESS  read-write
        STATUS  mandatory
        DESCRIPTION
                "This object identifies the instance of the
                index for the X.25 circuit open to the peer
                mapped by this table entry.  The well knows
                value {0 0} indicates no connection
                currently active.  For multiple connections,
                this identifies the index of a multiplexing
                table entry for the connections.  This can
                only be written to configure use of PVCs
                which means the identified circuit table
                entry for a write must be a PVC."
        DEFVAL { {0 0} }
        ::= { mioxPeerEntry 9 }

mioxPeerDescr   OBJECT-TYPE
        SYNTAX  DisplayString (SIZE (0..255))
        ACCESS  read-write
        STATUS  mandatory
        DESCRIPTION
                "This object returns any identification
                information about the peer.  An agent may
                supply the comment information found in the
                configuration file entry for this peer.  A
                zero length string indicates no information
                available."
        DEFVAL { ''h }
        ::= { mioxPeerEntry 10 }
```

```
-- ###########################################################
--              Peer Encapsulation Table
-- ###########################################################

mioxPeerEncTable OBJECT-TYPE
        SYNTAX  SEQUENCE OF MioxPeerEncEntry
        ACCESS  not-accessible
        STATUS  mandatory
        DESCRIPTION
                "This table contains the list of
                encapsulations used to communicate with a
                peer.  This table has two indexes, the first
                identifies the peer, the second
                distinguishes encapsulation types.

                The peer index, mioxPeerEncPeerIndex, value
                matches the value of the mioxPeerIndex entry
                for the peer.  The second index gives the
                priority of the different encapsulations.

                The encapsulation types are ordered in
                priority order.  For calling a peer, the
                first entry (mioxPeerEncEncIndex of 1) is
                tried first.  If the call doesn't succeed
                because the remote host clears the call due
                to incompatible call user data, the next
                entry in the list is tried.  Each entry is
                tried until the list is exhausted.

                For answering a call, the encapsulation type
                requested by the peer must be found the list
                or the call will be refused.  If there are
                no entries in this table for a peer, all
                call requests from the peer will be refused.

                Objects in this table can only be set when
                the mioxPeerStatus object with the same
                index has a value of underCreation.  When
                that status object is set to invalid and
                deleted, the entry in this table with that
                peer index must also be deleted."
        ::= { mioxPeer 2 }

mioxPeerEncEntry OBJECT-TYPE
        SYNTAX  MioxPeerEncEntry
```

```
      ACCESS    not-accessible
        STATUS  mandatory
        DESCRIPTION
                "Per connection information."
        INDEX { mioxPeerEncPeerIndex, mioxPeerEncEncIndex}
        ::= { mioxPeerEncTable 1 }

MioxPeerEncEntry ::= SEQUENCE {
        mioxPeerEncPeerIndex
                PositiveInteger,
        mioxPeerEncEncIndex
                PositiveInteger,
        mioxPeerEncType
                INTEGER
        }

mioxPeerEncPeerIndex OBJECT-TYPE
        SYNTAX  PositiveInteger
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "An index value that matches the
                mioxPeerTable index of the peer to which
                these encapsulation types apply."
        ::= { mioxPeerEncEntry 1 }

mioxPeerEncEncIndex     OBJECT-TYPE
        SYNTAX  PositiveInteger
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                "The second index in the table which
                distinguishes different encapsulation
                types."
        ::= { mioxPeerEncEntry 2 }

mioxPeerEncType OBJECT-TYPE
        SYNTAX  INTEGER (0..256)
        ACCESS  read-write
        STATUS  mandatory
        DESCRIPTION
                "The value of the encapsulation type.  For
                IP encapsulation this will have a value of
                204 (hex CC).  For SNAP encapsulated
                packets, this will have a value of 128 (hex
```

```
        80).  For CLNP, ISO 8473, this will have a
                value of 129 (hex 81).  For ES-ES, ISO 9542,
                this will have a value of 130 (hex 82).  A
                value of 197 (hex C5) identifies the Blacker
                X.25 encapsulation.  A value of 0,
                identifies the Null encapsulation.

                This value can only be written when the
                matching miox PeerStatus object has a value
                of underCreation.  The mioxPeerStatus object
                that matches this entry has a mioxPeerIndex
                that matches the value of the
                mioxPeerEndPeerIndex of this entry.  Setting
                this object to a value of 256 deletes this
                entry.  All other entries in the
                mioxPeerEntTable with the same
                mioxPeerEncPeerIndex but with
                mioxPeerEncEncIndex-es higher then this
                entry will all have their
                mioxPeerEncEncIndex values decremented by
                one."
        ::= { mioxPeerEncEntry 3 }

-- ############################################################

END
```

7. Acknowledgements

This document was produced by the x25mib working group:

Fred Baker, ACC
Art Berggreen, ACC
Frank Bieser
Gary Bjerke, Tandem
Bill Bowman, HP
Christopher Bucci, Datability
Charles Carvalho, ACC
Jeff Case, University of Tennessee at Knoxville
Angela Chen, HP
Carson Cheung, BNR
Tom Daniel, Spider Systems
Chuck Davin, MIT
Billy Durham, Honeywell
Richard Fox, Synoptics
Doug Geller, Data General
Herve Goguely, LIR Corp
Andy Goldthorpe, british-telecom
Walter D. Guilarte
David Gurevich
Steve Huston, Process Software Corporation
Jon Infante, ICL
Frank Kastenholz, FTP Software
Zbigniew Kielczewski, Eicon
Cheryl Krupezak, Georgia Tech
Mats Lindstrom, Diab Data AB
Andrew Malis, BBN
Evan McGinnis, 3Com
Gary (G.P.)Mussar, BNR
Chandy Nilakantan, 3Com
Randy Pafford, Data General
Ragnar Paulson, The Software Group Limited
Dave Perkins, Synoptics
Walter Pinkarschewsky, DEC
Karen Quidley, Data General
Chris Ranch, Novell
Paul S. Rarey, DHL Systems Inc.
Jim Roche, Newbridge Research
Philippe Roger, LIR Corp.
Timon Sloane
Mike Shand, DEC
Brad Steina, Microcom

Bob Stewart, Xyplex
Tom Sullivan, Data General
Rodney Thayer, Sable Technology Corporation
Mark Therieau, Microcom
Jane Thorn, Data General
Dean Throop, Data General
Maurice Turcotte, Racal Datacom
Mike Zendels, Data General

8. References

[1] Rose M., and K. McCloghrie, "Structure and Identification of Management Information for TCP/IP-based internets", STD 16, RFC 1155, Performance Systems International, Hughes LAN Systems, May, 1990.

[2] McCloghrie K., and M. Rose, "Management Information Base for Network Management of TCP/IP-based Internets", RFC 1156, Hughes LAN Systems, Performance Systems International, May, 1990.

[3] Case, J., M.S. Fedor, M.L. Schoffstall, and J. Davin, "Simple Network Management Protocol", STD 15, RFC 1157, SNMP Research, Performance Systems International, Performance Systems International, MIT Laboratory for Computer Science, May, 1990.

[4] Rose, M. and K. McCloghrie, Editors, "Towards Concise MIB Definitions", STD 16, RFC 1212, Performance Systems International, Hughes LAN Systems, March, 1991.

[5] Rose M., Editor, "Management Information Base for Network Management of TCP/IP-based internets", STD 17, RFC 1213. Performance Systems International, March, 1991.

[6] "Information processing systems - Open Systems Interconnection - Specification of Abstract Syntax Notation One (ASN.1)", International Organization for Standardization. International Standard 8824, December, 1987.

[7] "Information processing systems - Open Systems Interconnection - Specification of Basic Encoding Rules for Abstract Notation One (ASN.1)", International Organization for Standardization. International Standard 8825, December, 1987.

[8] Stewart, B., Editor, "Definitions of Managed Objects for RS-232-like Hardware Devices", RFC 1317, Xyplex, Inc., April, 1992.

[9] Throop, D., and F. Baker, "SNMP MIB extension for X.25 LAPB", RFC 1381, Data General Corporation, Advanced Computer Communications, November, 1992.

[10] Throop, D., Editor, "SNMP MIB extension for the X.25 Packet Layer", RFC 1382, Data General Corporation, November, 1991.

[11] Malis, A., D. Robinson, R. Ullmann "Multiprotocol Interconnect on X.25 and ISDN in the Packet Mode", RFC 1356, BBN Communications, Computervision Systems Integration, Process Software Corporation, August, 1992.

9. Authors' Addresses

Dean D. Throop
Data General Corporation
62 Alexander Dr.
Research Triangle Park, NC 27709

Phone: (919) 248 6081
EMail: throop@dg-rtp.dg.com

Table of Contents