

SNMP MIB Extension for the X.25 Packet Layer

Status of this Memo

This RFC specifies an IAB standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "IAB Official Protocol Standards" for the standardization state and status of this protocol. Distribution of this memo is unlimited.

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 the Packet Layer of X.25. The objects defined here, along with the objects in the "SNMP MIB Extension for LAPB" [9] and the "Definitions of Managed Objects for RS-232-like Hardware Devices" [8], combine to allow management of an X.25 protocol stack.

Table of Contents

1. The Network Management Framework	2
2. Objects	2
2.1 Format of Definitions	3
3. Overview	3
3.1 Informal Overview	3
3.2 Textual Conventions	4
3.3 Structure of MIB	4
3.4 Tables	5
3.5 Table Usage	6
3.6 Conformance	6
4. Object Definitions	7
5. Appendix: Revision History	62
July 30 1992	62
June 26 1992	62
June 1992	63
April 1992	63
February 1992	65
October 1991	65
June 1991	66
April 1991	66
6. Acknowledgements	66

7. References	67
8. Security Considerations	68
9. Author's Address	69

1. 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 STD 16/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. STD 16/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". STD 17/RFC 1213 [5] defines MIB-II, an evolution of MIB-I with changes to incorporate implementation experience and new operational requirements.

STD 15/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.

2. 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.

2.1. Format of Definitions

Section 4 contains the specification of all object types defined 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].

3. Overview

3.1. Informal Overview

This section describes how the objects defined below relate with other MIBs. This section is only informational to help understand how the pieces fit together.

The objects defined below are used in conjunction with MIB-II and other MIBs such as the LAPB MIB [9]. A system with a complete X.25 stack running over a synchronous line will have at least two interfaces in the ifTable defined in MIB-II. There will be an interface for LAPB and another interface for the packet layer of X.25. There will also be objects defined in the RS-232-like MIB for the physical sync line.

Each software interface identifies the layer below it used to send and receive packets. The X.25 MIB object, defined below, x25OperDataLinkId, specifies an instance of lapbAdmnIndex for the LAPB interface under that X.25. The LAPB object, lapbOperPortId, identifies an instance of the rs232PortIndex for the the Sync line used by LAPB.

For X.25 running over LAPB over Ethernet, the lapbOperPortId would identify the instance of ifIndex for the Ethernet interface.

Each X.25 subnetwork will have separate entries in the ifTable. Thus a system with two X.25 lines would have two ifTable entries for the two X.25 packet layers and two other entries for the two LAPB interfaces. Each X.25 Packet Layer MIB would identify the instance of the LAPB MIB for the interface below it. Each LAPB MIB would identify the Sync line below it. The system would also have two entries in the rs232PortTable and rs232SyncPortTable for the two physical lines.

Since the ifTable as defined in MIB-II is device independent, it doesn't have anything specific for any type of interface. The

objects below define the X.25 packet layer specific information for an interface of type X.25. Different X.25 interfaces can also be differentiated by matching the values of ifIndex with x25AdmnIndex.

3.2. Textual Conventions

This MIB introduces a new data type as a textual convention for use with X.25. This textual convention enhances the readability of the specification and can ease comparison with other specifications if appropriate. It should be noted that the introduction of such textual conventions has no effect on either the syntax nor the semantics of any managed objects. These conventions are merely an artifact of the explanatory method used. Objects defined in terms of one of these methods are always encoded by means of the rules that define the primitive type. Hence, no changes to the SMI or the SNMP are necessary to accommodate these textual conventions which are adopted merely for the convenience of readers and writers in pursuit of the elusive goal of clear, concise, and unambiguous MIB documents.

This MIB introduces the data type of:

X121Address

3.3. Structure of MIB

Instances of the objects defined below represent attributes of an X.25 Packet Layer interface. At present these interfaces are identified by an ifType object in the Internet-standard MIB-II [5] of:

ddn-x25(4), and
rfc887-x25(5).

For these interfaces, the value of the ifSpecific variable in the MIB-II [5] has the OBJECT IDENTIFIER value:

x25 OBJECT IDENTIFIER ::= { transmission 5 }

The objects defined below are similar to those defined in a draft ISO document for X.25 management [11]. Some object definitions also reference the ISO specification for X.25 [10] to specify the section that will give the reader additional information about the object. Access to those documents maybe useful (but isn't essential) to understand the names and semantics of some objects. The similarity of these objects with the ISO objects minimizes the instrumentation required by those systems that support both OSI and TCP/IP management protocols.

Since the objects defined here are extensions to the Internet Standard MIB [2] and thus also an extension of the second version, MIB-II [5], the objects defined here explicitly do not duplicate objects defined in existing standards. In some instances clarification of how to apply those objects has been given.

The relationship between an X.25 Packet Layer interface and an interface in the context of the Internet-standard MIB [5] is one-to-one. As such, the value of an ifIndex object instance can be directly used to identify corresponding instances of the objects defined below.

3.4. Tables

The objects below form several tables. These tables are:

- x25AdmnTable
- x25OperTable
- x25StatTable
- x25ChannelTable
- x25CircuitTable
- x25ClearedCircuitTable
- x25CallParmTable

The x25AdmnTable defines objects for the parameters of an X.25 interface which the administrator can read and set. These objects are used at interface initialization time to start the interface. Once the interface has started, changes to the objects in the Administration table may not take affect until the interface is re-initialized.

The x25OperTable defines objects that report the current parameters used by a running interface. These objects are read-only.

The x25StatTable defines objects that report operational statistics for an X.25 interface. These are read-only counters of events that occurred at the interface.

The x25ChannelTable defines objects to allow the administrator to manage the division of channel numbers.

The x25CircuitTable defines objects that return information about existing X.25 circuits. These entries result from calls placed or answered by the PLE or from PVCs.

The x25ClearedCircuitTable contains objects for recording the termination information from circuits that cleared abnormally.

The x25CallParmTable defines the call parameters used to call other systems. This table contains call parameter entries which are referenced by other tables. For example, the x25AdmnTable has one object that identifies the entry in the table for the default PLE parameters. The x25CircuitTable has one object that identifies the entry in the x25CallParmTable for the parameters in use by that circuit. Other MIBs may also reference entries to identify call parameters to use to make X.25 calls.

3.5. Table Usage

Different tables provide different functions. The administrator sets the starting X.25 parameters in the x25AdmnTable for the X.25 PLE; these objects include a reference to the x25CallParmTable entry to identify the default call parameters for the PLE. Once all the parameters are set, the administrator initializes the interface. As part of initializing the interface, the operating parameters are copied into the interface from the x25AdmnTable; these parameters are viewable by getting the objects in the x25OperTable. (The interface maybe started by setting the value of ifAdminStatus to up.) If any PVCs are configured, their parameters can be set in the the x25CircuitTable before initializing the interface; this should be done in conjunction with configuring higher layer entities to use the PVCs via the MIBs for those entities.

Once the PLE completes initialization, it makes additional entries in the x25CircuitTable for calls placed or answered. When a circuit is cleared, the status of the entry for the circuit is set to closed and, if the clear is abnormal, an entry will also be made in the x25ClearedCircuitTable. An entry in the x25CircuitTable with a status of closed maybe deleted by the agent at its convenience. A closed entry will always be reused at the time the PLE re-allocates the channel number of the entry for another call. The call parameters used for a circuit can be found by looking in the x25CircuitTable and following the x25CircuitCallParamId pointer to the entry in the x25CallParmTable that contains the parameters.

There are no mechanisms in the X.25 MIB for telling the PLE to place an X.25 call. Such mechanisms belong in the MIBs for the higher layer entities that use the X.25 circuits.

3.6. Conformance

All the objects defined here are mandatory. To claim conformance with this MIB an implementation must support all objects. However some objects pertain to features that are optional. There are values defined for those objects that indicate the implementation does not support the optional feature. The agent for such an implementation

must support reading the object and return the value that indicates the optional feature isn't supported and reject set requests to change the object.

Some optional features have more than one object that pertain to it (window rotation has a timer, a count, and a counter for timer runouts). In such case, any object which indicates the optional feature isn't supported is sufficient to indicate the feature isn't supported and the values of the other objects relative to that feature are undefined.

4. Object Definitions

```
RFC1382-MIB DEFINITIONS ::= BEGIN
```

```
IMPORTS
```

```
    Counter, Gauge, TimeTicks
```

```
        FROM RFC1155-SMI
```

```
    OBJECT-TYPE
```

```
        FROM RFC-1212
```

```
    DisplayString, transmission
```

```
        FROM RFC1213-MIB
```

```
    TRAP-TYPE
```

```
        FROM RFC-1215
```

```
    EntryStatus
```

```
        FROM RFC1271-MIB
```

```
    PositiveInteger,
```

```
    IfIndexType
```

```
        FROM RFC1381-MIB;
```

```
x25      OBJECT IDENTIFIER ::= { transmission 5 }
```

```
-- Support of the X25 subtree and all subtrees under it
-- is mandatory for all agents of system that implement X.25.
```

```
X121Address ::= OCTET STRING (SIZE(0..17))
```

```
-- 0 to 17 bytes in length containing the ASCII
```

```
-- characters [0-9], each octet contains one digit
```

```
-- of the address.
```

```
-- #####
```

```
--           X.25 Administration Table
```

```
-- #####
```

```
x25AdmnTable OBJECT-TYPE
```

SYNTAX SEQUENCE OF X25AdmnEntry
 ACCESS not-accessible
 STATUS mandatory
 DESCRIPTION

"This table contains the administratively set configuration parameters for an X.25 Packet Level Entity (PLE).

Most of the objects in this table have corresponding objects in the x25OperTable. This table contains the values as last set by the administrator. The x25OperTable contains the values actually in use by an X.25 PLE.

Changing an administrative value may or may not change a current operating value. The operating value may not change until the interface is restarted. Some implementations may change the values immediately upon changing the administrative table. All implementations are required to load the values from the administrative table when initializing a PLE."

::= { x25 1 }

x25AdmnEntry OBJECT-TYPE

SYNTAX X25AdmnEntry
 ACCESS not-accessible
 STATUS mandatory
 DESCRIPTION
 "Entries of x25AdmnTable."
 INDEX { x25AdmnIndex }
 ::= { x25AdmnTable 1 }

X25AdmnEntry ::= SEQUENCE {
 x25AdmnIndex
 IfIndexType,
 x25AdmnInterfaceMode
 INTEGER,
 x25AdmnMaxActiveCircuits
 INTEGER,
 x25AdmnPacketSequencing
 INTEGER,
 x25AdmnRestartTimer
 PositiveInteger,
 x25AdmnCallTimer
 PositiveInteger,


```

x25AdmnResetTimer
    PositiveInteger,
x25AdmnClearTimer
    PositiveInteger,
x25AdmnWindowTimer
    PositiveInteger,
x25AdmnDataRxmtTimer
    PositiveInteger,
x25AdmnInterruptTimer
    PositiveInteger,
x25AdmnRejectTimer
    PositiveInteger,
x25AdmnRegistrationRequestTimer
    PositiveInteger,
x25AdmnMinimumRecallTimer
    PositiveInteger,
x25AdmnRestartCount
    INTEGER,
x25AdmnResetCount
    INTEGER,
x25AdmnClearCount
    INTEGER,
x25AdmnDataRxmtCount
    INTEGER,
x25AdmnRejectCount
    INTEGER,
x25AdmnRegistrationRequestCount
    INTEGER,
x25AdmnNumberPVCs
    INTEGER,
x25AdmnDefCallParamId
    OBJECT IDENTIFIER,
x25AdmnLocalAddress
    X121Address,
x25AdmnProtocolVersionSupported
    OBJECT IDENTIFIER
}

x25AdmnIndex OBJECT-TYPE
    SYNTAX  IfIndexType
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The ifIndex value for the X.25 Interface."
    ::= { x25AdmnEntry 1 }

x25AdmnInterfaceMode OBJECT-TYPE
    SYNTAX  INTEGER {

```

```

        dte (1),
        dce (2),
        dxe (3)
    }
ACCESS read-write
STATUS mandatory
DESCRIPTION
    "Identifies DCE/DTE mode in which the
    interface operates.  A value of dxe
    indicates the mode will be determined by XID
    negotiation."
REFERENCE "10733 5.9 interfaceMode"
::= { x25AdmnEntry 2 }

x25AdmnMaxActiveCircuits OBJECT-TYPE
SYNTAX INTEGER (0..4096)
ACCESS read-write
STATUS mandatory
DESCRIPTION
    "The maximum number of circuits this PLE can
    support; including PVCs."
REFERENCE "10733 5.9 maxActiveCircuits;
    See ISO 8208, Section 3.7"
::= { x25AdmnEntry 3 }

x25AdmnPacketSequencing OBJECT-TYPE
SYNTAX INTEGER {
    modulo8 (1),
    modulo128 (2)
}
ACCESS read-write
STATUS mandatory
DESCRIPTION
    "The modulus of the packet sequence number
    space."
REFERENCE "10733 extendedPacketSequencing;
    See ISO 8208 Section 7.1.1"
::= { x25AdmnEntry 4 }

x25AdmnRestartTimer OBJECT-TYPE
SYNTAX PositiveInteger
ACCESS read-write
STATUS mandatory
DESCRIPTION
    "The T20 restart timer in milliseconds."
REFERENCE "10733 5.9 restartTime
    See ISO 8208 Section 4.1, table 26"
::= { x25AdmnEntry 5 }

```

x25AdmnCallTimer OBJECT-TYPE
SYNTAX PositiveInteger
ACCESS read-write
STATUS mandatory
DESCRIPTION
 "The T21 Call timer in milliseconds."
REFERENCE "10733 callTime;
 See ISO 8208 Section 5.2.1, table 26"
::= { x25AdmnEntry 6 }

x25AdmnResetTimer OBJECT-TYPE
SYNTAX PositiveInteger
ACCESS read-write
STATUS mandatory
DESCRIPTION
 "The T22 Reset timer in milliseconds."
REFERENCE "10733 resetTime;
 See ISO 8208 Section 8.1, table 26"
::= { x25AdmnEntry 7 }

x25AdmnClearTimer OBJECT-TYPE
SYNTAX PositiveInteger
ACCESS read-write
STATUS mandatory
DESCRIPTION
 "The T23 Clear timer in milliseconds."
REFERENCE "10733 clearTime;
 See ISO 8208 Section 5.5.1, table 26"
::= { x25AdmnEntry 8 }

x25AdmnWindowTimer OBJECT-TYPE
SYNTAX PositiveInteger
ACCESS read-write
STATUS mandatory
DESCRIPTION
 "The T24 window status transmission timer in
 milliseconds. A value of 2147483647
 indicates no window timer in use."
REFERENCE "10733 5.10.1 windowTime (opt);
 See ISO 8208 Section 11.2.2, table 26"
::= { x25AdmnEntry 9 }

x25AdmnDataRxmtTimer OBJECT-TYPE
SYNTAX PositiveInteger
ACCESS read-write
STATUS mandatory
DESCRIPTION
 "The T25 data retransmission timer in

milliseconds. A value of 2147483647 indicates no data retransmission timer in use."

REFERENCE "10733 5.10.1 dataRetransmissionTime (opt);
See ISO 8208 Section 11.2.1, table 26"

::= { x25AdmnEntry 10 }

x25AdmnInterruptTimer OBJECT-TYPE

SYNTAX PositiveInteger

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The T26 interrupt timer in milliseconds. A value of 2147483647 indicates no interrupt timer in use."

REFERENCE "10733 interruptTime;
See ISO 8208 Section 6.8.1, table 26"

::= { x25AdmnEntry 11 }

x25AdmnRejectTimer OBJECT-TYPE

SYNTAX PositiveInteger

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The T27 Reject retransmission timer in milliseconds. A value of 2147483647 indicates no reject timer in use."

REFERENCE "10733 5.10.1 dataRejectTime (opt);
See ISO 8208 Section 13.4.1, table 26"

::= { x25AdmnEntry 12 }

x25AdmnRegistrationRequestTimer OBJECT-TYPE

SYNTAX PositiveInteger

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The T28 registration timer in milliseconds. A value of 2147483647 indicates no registration timer in use."

REFERENCE "10733 5.8.1 registrationRequestTime (opt)
See ISO 8208 Section 13.1.1.1, table 26"

::= { x25AdmnEntry 13 }

x25AdmnMinimumRecallTimer OBJECT-TYPE

SYNTAX PositiveInteger

ACCESS read-write

STATUS mandatory

DESCRIPTION

```
        "Minimum time interval between unsuccessful
        call attempts in milliseconds."
REFERENCE "10733 5.9 minimum RecallTimer"
::= { x25AdmnEntry 14 }

x25AdmnRestartCount OBJECT-TYPE
    SYNTAX  INTEGER (0..65535)
    ACCESS  read-write
    STATUS  mandatory
    DESCRIPTION
        "The R20 restart retransmission count."
REFERENCE "10733 5.9 restartCount;
        See ISO 8208 Section 4.1, table 27"
::= { x25AdmnEntry 15 }

x25AdmnResetCount OBJECT-TYPE
    SYNTAX  INTEGER (0..65535)
    ACCESS  read-write
    STATUS  mandatory
    DESCRIPTION
        "The r22 Reset retransmission count."
REFERENCE "10733 resetCount;
        See section ISO 8208 8.1, table 27"
::= { x25AdmnEntry 16 }

x25AdmnClearCount OBJECT-TYPE
    SYNTAX  INTEGER (0..65535)
    ACCESS  read-write
    STATUS  mandatory
    DESCRIPTION
        "The r23 Clear retransmission count."
REFERENCE "10733 clearCount;
        See ISO 8208 Section 5.5.1, table 27"
::= { x25AdmnEntry 17 }

x25AdmnDataRxmtCount OBJECT-TYPE
    SYNTAX  INTEGER (0..65535)
    ACCESS  read-write
    STATUS  mandatory
    DESCRIPTION
        "The R25 Data retransmission count.  This
        value is irrelevant if the
        x25AdmnDataRxmtTimer indicates no timer in
        use."
REFERENCE "10733 5.10.1 dataRetransmissionCount (opt)
        See ISO 8208 Section 11.2.1, table 27"
::= { x25AdmnEntry 18 }
```

x25AdmnRejectCount OBJECT-TYPE

SYNTAX INTEGER (0..65535)

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The R27 reject retransmission count. This value is irrelevant if the x25AdmnRejectTimer indicates no timer in use."

REFERENCE "10733 5.10.1 dataRejectCount (opt)"

::= { x25AdmnEntry 19 }

x25AdmnRegistrationRequestCount OBJECT-TYPE

SYNTAX INTEGER (0..65535)

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The R28 Registration retransmission Count. This value is irrelevant if the x25AdmnRegistrationRequestTimer indicates no timer in use."

REFERENCE "10733 5.8.1 registrationRequestCount (opt);

See ISO 8208 Section 13.1.1.1, table 27"

::= { x25AdmnEntry 20 }

x25AdmnNumberPVCs OBJECT-TYPE

SYNTAX INTEGER (0..4096)

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The number of PVC configured for this PLE. The PVCs use channel numbers from 1 to this number."

::= { x25AdmnEntry 21 }

x25AdmnDefCallParamId OBJECT-TYPE

SYNTAX OBJECT IDENTIFIER

ACCESS read-write

STATUS mandatory

DESCRIPTION

"This identifies the instance of the x25CallParmIndex for the entry in the x25CallParmTable which contains the default call parameters for this PLE."

::= { x25AdmnEntry 22 }

x25AdmnLocalAddress OBJECT-TYPE

SYNTAX X121Address

```

ACCESS    read-write
STATUS    mandatory
DESCRIPTION
    "The local address for this PLE subnetwork.
    A zero length address maybe returned by PLEs
    that only support PVCs."
REFERENCE "10733 5.9 localDTEAddress"
::= { x25AdmnEntry 23 }

x25AdmnProtocolVersionSupported OBJECT-TYPE
SYNTAX    OBJECT IDENTIFIER
ACCESS    read-write
STATUS    mandatory
DESCRIPTION
    "Identifies the version of the X.25 protocol
    this interface should support.  Object
    identifiers for common versions are defined
    below in the x25ProtocolVersion subtree."
REFERENCE "10733 5.9 protocolVersionSupported"
::= { x25AdmnEntry 24 }

-- #####
--           X.25 Operational Table
-- #####

x25OperTable OBJECT-TYPE
SYNTAX    SEQUENCE OF X25OperEntry
ACCESS    not-accessible
STATUS    mandatory
DESCRIPTION
    "The operation parameters in use by the X.25
    PLE."
::= { x25 2 }

x25OperEntry OBJECT-TYPE
SYNTAX    X25OperEntry
ACCESS    not-accessible
STATUS    mandatory
DESCRIPTION
    "Entries of x25OperTable."
INDEX { x25OperIndex }
::= { x25OperTable 1 }

X25OperEntry ::= SEQUENCE {
    x25OperIndex
        IfIndexType,
    x25OperInterfaceMode

```

```
        INTEGER,
x25OperMaxActiveCircuits
        INTEGER,
x25OperPacketSequencing
        INTEGER,
x25OperRestartTimer
        PositiveInteger,
x25OperCallTimer
        PositiveInteger,
x25OperResetTimer
        PositiveInteger,
x25OperClearTimer
        PositiveInteger,
x25OperWindowTimer
        PositiveInteger,
x25OperDataRxmtTimer
        PositiveInteger,
x25OperInterruptTimer
        PositiveInteger,
x25OperRejectTimer
        PositiveInteger,
x25OperRegistrationRequestTimer
        PositiveInteger,
x25OperMinimumRecallTimer
        PositiveInteger,
x25OperRestartCount
        INTEGER,
x25OperResetCount
        INTEGER,
x25OperClearCount
        INTEGER,
x25OperDataRxmtCount
        INTEGER,
x25OperRejectCount
        INTEGER,
x25OperRegistrationRequestCount
        INTEGER,
x25OperNumberPVCs
        INTEGER,
x25OperDefCallParamId
        OBJECT IDENTIFIER,
x25OperLocalAddress
        X121Address,
x25OperDataLinkId
        OBJECT IDENTIFIER,
x25OperProtocolVersionSupported
        OBJECT IDENTIFIER
}
```



```
x25OperIndex OBJECT-TYPE
    SYNTAX  IfIndexType
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The ifIndex value for the X.25 interface."
    ::= { x25OperEntry 1 }

x25OperInterfaceMode OBJECT-TYPE
    SYNTAX  INTEGER {
        dte (1),
        dce (2),
        dxe (3)
    }
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "Identifies DCE/DTE mode in which the
         interface operates.  A value of dxe
         indicates the role will be determined by XID
         negotiation at the Link Layer and that
         negotiation has not yet taken place."
    REFERENCE "10733 5.9 interfaceMode"
    ::= { x25OperEntry 2 }

x25OperMaxActiveCircuits OBJECT-TYPE
    SYNTAX  INTEGER (0..4096)
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "Maximum number of circuits this PLE can
         support."
    REFERENCE "10733 5.9 maxActiveCircuits
         See ISO 8208, Section 3.7"
    ::= { x25OperEntry 3 }

x25OperPacketSequencing OBJECT-TYPE
    SYNTAX  INTEGER {
        modulo8 (1),
        modulo128 (2)
    }
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The modulus of the packet sequence number
         space."
    REFERENCE "10733 extendedPacketSequencing;
         See ISO 8208 Section 7.1.1"
```

```
 ::= { x25OperEntry 4 }

x25OperRestartTimer OBJECT-TYPE
    SYNTAX  PositiveInteger
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The T20 restart timer in milliseconds."
    REFERENCE "10733 5.9 restartTime;
        See ISO 8208 Section 4.1, table 26"
    ::= { x25OperEntry 5 }

x25OperCallTimer OBJECT-TYPE
    SYNTAX  PositiveInteger
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The T21 Call timer in milliseconds."
    REFERENCE "10733 callTime;
        See ISO 8208 Section 5.2.1, table 26"
    ::= { x25OperEntry 6 }

x25OperResetTimer OBJECT-TYPE
    SYNTAX  PositiveInteger
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The T22 Reset timer in milliseconds."
    REFERENCE "10733 resetTime;
        See ISO 8208 Section 8.1, table 26"
    ::= { x25OperEntry 7 }

x25OperClearTimer OBJECT-TYPE
    SYNTAX  PositiveInteger
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The T23 Clear timer in milliseconds."
    REFERENCE "10733 clearTime;
        See ISO 8208 Section 5.5.1, table 26"
    ::= { x25OperEntry 8 }

x25OperWindowTimer OBJECT-TYPE
    SYNTAX  PositiveInteger
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The T24 window status transmission timer
```

```
        milliseconds.  A value of 2147483647
        indicates no window timer in use."
REFERENCE "10733 5.10.1 windowTime (opt);
        See ISO 8208 Section 11.2.2, table 26"
 ::= { x25OperEntry 9 }

x25OperDataRxmtTimer OBJECT-TYPE
    SYNTAX  PositiveInteger
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The T25 Data Retransmission timer in
        milliseconds.  A value of 2147483647
        indicates no data retransmission timer in
        use."
REFERENCE "10733 5.10.1 dataRetransmissionTime (opt);
        See ISO 8208 Section 11.2.1, table 26"
 ::= { x25OperEntry 10 }

x25OperInterruptTimer OBJECT-TYPE
    SYNTAX  PositiveInteger
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The T26 Interrupt timer in milliseconds.  A
        value of 2147483647 indicates interrupts are
        not being used."
REFERENCE "10733 interruptTime;
        See ISO 8208 Section 6.8.1, table 26"
 ::= { x25OperEntry 11 }

x25OperRejectTimer OBJECT-TYPE
    SYNTAX  PositiveInteger
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The T27 Reject retransmission timer in
        milliseconds.  A value of 2147483647
        indicates no reject timer in use."
REFERENCE "10733 5.10.1 dataRejectTime (opt);
        See ISO 8208 Section 13.4.1, table 26"
 ::= { x25OperEntry 12 }

x25OperRegistrationRequestTimer OBJECT-TYPE
    SYNTAX  PositiveInteger
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
```

```
        "The T28 registration timer in milliseconds.
        A value of 2147483647 indicates no
        registration timer in use."
REFERENCE "10733 5.8.1 registrationRequestTime (opt);
        See ISO 8208      Section 13.1.1.1, table 26"
::= { x25OperEntry 13 }

x25OperMinimumRecallTimer OBJECT-TYPE
    SYNTAX  PositiveInteger
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "Minimum time interval between unsuccessful
        call attempts in milliseconds."
REFERENCE "10733 5.9 minimum RecallTimer"
::= { x25OperEntry 14 }

x25OperRestartCount OBJECT-TYPE
    SYNTAX  INTEGER (0..65535)
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The R20 restart retransmission count."
REFERENCE "10733 5.9 restartCount
        See ISO 8208      Section 4.1, table 27"
::= { x25OperEntry 15 }

x25OperResetCount OBJECT-TYPE
    SYNTAX  INTEGER (0..65535)
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The r22 Reset retransmission count."
REFERENCE "10733 resetCount;
        See section ISO 8208 8.1, table 27"
::= { x25OperEntry 16 }

x25OperClearCount OBJECT-TYPE
    SYNTAX  INTEGER (0..65535)
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The r23 Clear retransmission count."
REFERENCE "10733 clearCount;
        See ISO 8208 Section 5.5.1, table 27"
::= { x25OperEntry 17 }

x25OperDataRxmtCount OBJECT-TYPE
```

```

SYNTAX  INTEGER (0..65535)
ACCESS  read-only
STATUS  mandatory
DESCRIPTION
    "The R25 Data retransmission count.  This
    value is undefined if the
    x25OperDataRxmtTimer indicates no timer in
    use."
REFERENCE "10733 5.10.1 dataRetransmissionCount (opt);
    See ISO 8208 Section 11.2.1, table 27"
::= { x25OperEntry 18 }

```

x25OperRejectCount OBJECT-TYPE

```

SYNTAX  INTEGER (0..65535)
ACCESS  read-only
STATUS  mandatory
DESCRIPTION
    "The R27 reject retransmission count.  This
    value is undefined if the x25OperRejectTimer
    indicates no timer in use."
REFERENCE "10733 5.10.1 dataRejectCount (opt)"
::= { x25OperEntry 19 }

```

x25OperRegistrationRequestCount OBJECT-TYPE

```

SYNTAX  INTEGER (0..65535)
ACCESS  read-only
STATUS  mandatory
DESCRIPTION
    "The R28 Registration retransmission Count.
    This value is undefined if the
    x25OperRegistrationRequestTimer indicates no
    timer in use."
REFERENCE "10733 5.8.1 registrationRequestCount (opt);
    See ISO 8208 Section 13.1.1.1, table 27"
::= { x25OperEntry 20 }

```

x25OperNumberPVCs OBJECT-TYPE

```

SYNTAX  INTEGER (0..4096)
ACCESS  read-only
STATUS  mandatory
DESCRIPTION
    "The number of PVC configured for this PLE.
    The PVCs use channel numbers from 1 to this
    number."
::= { x25OperEntry 21 }

```

x25OperDefCallParamId OBJECT-TYPE

```

SYNTAX  OBJECT IDENTIFIER

```

```

ACCESS    read-only
STATUS    mandatory
DESCRIPTION
    "This identifies the instance of the
    x25CallParmIndex for the entry in the
    x25CallParmTable that contains the default
    call parameters for this PLE."
::= { x25OperEntry 22 }

x25OperLocalAddress OBJECT-TYPE
    SYNTAX  X121Address
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The local address for this PLE subnetwork.
        A zero length address maybe returned by PLEs
        that only support PVCs."
    REFERENCE "10733 5.9 localDTEAddress"
    ::= { x25OperEntry 23 }

x25OperDataLinkId OBJECT-TYPE
    SYNTAX  OBJECT IDENTIFIER
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "This identifies the instance of the index
        object in the first table of the most device
        specific MIB for the interface used by this
        PLE."
    ::= { x25OperEntry 24 }

x25OperProtocolVersionSupported OBJECT-TYPE
    SYNTAX  OBJECT IDENTIFIER
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "Identifies the version of the X.25 protocol
        this interface supports. Object identifiers
        for common versions are defined below in the
        x25ProtocolVersion subtree."
    REFERENCE "10733 5.9 protocolVersionSupported"
    ::= { x25OperEntry 25 }

--      MIB-II also provides:

-- ifDescr:
-- On an X.25 interface this must include sufficient

```

```

-- information to enable the system's administrator
-- to determine the appropriate configuration
-- information on a system having multiple X.25
-- subnetworks.

-- ifType: ddN-x25 or rfc877-x25
--     an interface of type ddN-x25 will use an algorithm to
--     translate between X.121 address and IP addresses.
--     An interface of type rfc877-x25 will use a
--     configuration table to translate between X.121
--     addresses and IP addresses.

-- ifMtu: the maximum PDU a higher layer can pass to X.25 or
-- receive from X.25

-- ifSpeed:
-- This will be the value of the local clock for this line.
-- A value of zero indicates external clocking.

-- ifAdminStatus:

-- ifOperStatus

-- ifLastChange

-- #####
--           X.25 Statistics Table
-- #####

x25StatTable OBJECT-TYPE
    SYNTAX  SEQUENCE OF X25StatEntry
    ACCESS  not-accessible
    STATUS  mandatory
    DESCRIPTION
        "Statistics information about this X.25
        PLE."
    ::= { x25 3 }

x25StatEntry OBJECT-TYPE
    SYNTAX  X25StatEntry
    ACCESS  not-accessible
    STATUS  mandatory
    DESCRIPTION
        "Entries of the x25StatTable."
        INDEX { x25StatIndex }
    ::= { x25StatTable 1 }

```

```
X25StatEntry ::= SEQUENCE {  
    x25StatIndex  
        IfIndexType,  
    x25StatInCalls  
        Counter,  
    x25StatInCallRefusals  
        Counter,  
    x25StatInProviderInitiatedClears  
        Counter,  
    x25StatInRemotelyInitiatedResets  
        Counter,  
    x25StatInProviderInitiatedResets  
        Counter,  
    x25StatInRestarts  
        Counter,  
    x25StatInDataPackets  
        Counter,  
    x25StatInAccusedOfProtocolErrors  
        Counter,  
    x25StatInInterrupts  
        Counter,  
    x25StatOutCallAttempts  
        Counter,  
    x25StatOutCallFailures  
        Counter,  
    x25StatOutInterrupts  
        Counter,  
    x25StatOutDataPackets  
        Counter,  
    x25StatOutgoingCircuits  
        Gauge,  
    x25StatIncomingCircuits  
        Gauge,  
    x25StatTwowayCircuits  
        Gauge,  
    x25StatRestartTimeouts  
        Counter,  
    x25StatCallTimeouts  
        Counter,  
    x25StatResetTimeouts  
        Counter,  
    x25StatClearTimeouts  
        Counter,  
    x25StatDataRxmtTimeouts  
        Counter,  
    x25StatInterruptTimeouts  
        Counter,  
    x25StatRetryCountExceeded
```



```

        Counter,
x25StatClearCountExceededs
        Counter
    }

x25StatIndex OBJECT-TYPE
    SYNTAX  IfIndexType
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The ifIndex value for the X.25 interface."
    ::= { x25StatEntry 1 }

x25StatInCalls OBJECT-TYPE
    SYNTAX  Counter
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The number of incoming calls received."
    ::= { x25StatEntry 2 }

x25StatInCallRefusals OBJECT-TYPE
    SYNTAX  Counter
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The number of incoming calls refused.  This
        includes calls refused by the PLE and by
        higher layers.  This also includes calls
        cleared because of restricted fast select."
    ::= { x25StatEntry 3 }

x25StatInProviderInitiatedClears          OBJECT-TYPE
    SYNTAX  Counter
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The number of clear requests with a cause
        code other than DTE initiated."
        REFERENCE "10733 providerInitiatedDisconnect"
    ::= { x25StatEntry 4 }

x25StatInRemotelyInitiatedResets          OBJECT-TYPE
    SYNTAX  Counter
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The number of reset requests received with

```

cause code DTE initiated."
REFERENCE "10733 remotelyInitiatedResets"
::= { x25StatEntry 5 }

x25StatInProviderInitiatedResets OBJECT-TYPE
SYNTAX Counter
ACCESS read-only
STATUS mandatory
DESCRIPTION
"The number of reset requests received with
cause code other than DTE initiated."
REFERENCE "10733 ProviderInitiatedResets"
::= { x25StatEntry 6 }

x25StatInRestarts OBJECT-TYPE
SYNTAX Counter
ACCESS read-only
STATUS mandatory
DESCRIPTION
"The number of remotely initiated (including
provider initiated) restarts experienced by
the PLE excluding the restart associated
with bringing up the PLE interface. This
only counts restarts received when the PLE
already has an established connection with
the remove PLE."
REFERENCE "10733 5.9 remotelyInitiatedRestarts"
::= { x25StatEntry 7 }

x25StatInDataPackets OBJECT-TYPE
SYNTAX Counter
ACCESS read-only
STATUS mandatory
DESCRIPTION
"The number of data packets received."
REFERENCE "10733 5.9 dataPacketsReceived."
::= { x25StatEntry 8 }

x25StatInAccusedOfProtocolErrors OBJECT-TYPE
SYNTAX Counter
ACCESS read-only
STATUS mandatory
DESCRIPTION
"The number of packets received containing a
procedure error cause code. These include
clear, reset, restart, or diagnostic
packets."
REFERENCE "CD 10733 5.9 accusedOfProtocolError"

```
 ::= { x25StatEntry 9 }

x25StatInInterrupts OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of interrupt packets received by
        the PLE or over the PVC/VC."
    REFERENCE "10733 interruptPacketsReceived"
    ::= { x25StatEntry 10 }

x25StatOutCallAttempts OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of calls attempted."
    REFERENCE "10733 5.9 callAttempts"
    ::= { x25StatEntry 11 }

x25StatOutCallFailures OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of call attempts which failed.
        This includes calls that were cleared
        because of restrictive fast select."
    ::= { x25StatEntry 12 }

x25StatOutInterrupts OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of interrupt packets send by the
        PLE or over the PVC/VC."
    REFERENCE "10733 InterruptPacketsSent"
    ::= { x25StatEntry 13 }

x25StatOutDataPackets OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of data packets sent by this
        PLE."
```

REFERENCE "10733 dataPacketSent"
 ::= { x25StatEntry 14 }

x25StatOutgoingCircuits OBJECT-TYPE
SYNTAX Gauge
ACCESS read-only
STATUS mandatory
DESCRIPTION
 "The number of active outgoing circuits.
 This includes call requests sent but not yet
 confirmed. This does not count PVCs."
 ::= { x25StatEntry 15 }

x25StatIncomingCircuits OBJECT-TYPE
SYNTAX Gauge
ACCESS read-only
STATUS mandatory
DESCRIPTION
 "The number of active Incoming Circuits.
 This includes call indications received but
 not yet acknowledged. This does not count
 PVCs."
 ::= { x25StatEntry 16 }

x25StatTwowayCircuits OBJECT-TYPE
SYNTAX Gauge
ACCESS read-only
STATUS mandatory
DESCRIPTION
 "The number of active two-way Circuits.
 This includes call requests sent but not yet
 confirmed. This does not count PVCs."
 ::= { x25StatEntry 17 }

x25StatRestartTimeouts OBJECT-TYPE
SYNTAX Counter
ACCESS read-only
STATUS mandatory
DESCRIPTION
 "The number of times the T20 restart timer
 expired."
REFERENCE "10733 5.9 restartTimeouts"
 ::= { x25StatEntry 18 }

x25StatCallTimeouts OBJECT-TYPE
SYNTAX Counter
ACCESS read-only
STATUS mandatory

```
DESCRIPTION
    "The number of times the T21 call timer
    expired."
REFERENCE "10733 5.9 callTimeouts"
::= { x25StatEntry 19 }

x25StatResetTimeouts    OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of times the T22 reset timer
        expired."
    REFERENCE "10733 5.9 resetTimeouts"
    ::= { x25StatEntry 20 }

x25StatClearTimeouts    OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of times the T23 clear timer
        expired."
    REFERENCE "10733 5.9 clearTimeouts"
    ::= { x25StatEntry 21 }

x25StatDataRxmtTimeouts OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of times the T25 data timer
        expired."
    REFERENCE "10733 5.9 dataRetransmissionsTimerExpiries"
    ::= { x25StatEntry 22 }

x25StatInterruptTimeouts OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of times the T26 interrupt timer
        expired."
    REFERENCE "10733 5.9 interruptTimerExpires"
    ::= { x25StatEntry 23 }

x25StatRetryCountExceededs OBJECT-TYPE
    SYNTAX Counter
```

```

ACCESS    read-only
STATUS    mandatory
DESCRIPTION
    "The number of times a retry counter was
    exhausted."
REFERENCE "10733 5.9 retryCountsExceeded"
::= { x25StatEntry 24 }

x25StatClearCountExceeded OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of times the R23 clear count was
        exceeded."
    REFERENCE "10733 5.9 clearCountsExceeded"
    ::= { x25StatEntry 25 }

--      MIB-II also contains:

-- ifInOctets: Number of data octets delivered to upper
-- layer entities.

-- ifInUcastPkts: Number of packets with a clear M-bit
-- delivered to higher layer entities.

-- ifDiscards: Number of packets dropped for lack of buffering

-- ifInErrors: Number of packets received containing errors
-- REFERENCE ProtocolErrorsDetectedLocally

-- ifInUnknownProtos: Number of packets with unknown circuit
-- identifier.

-- ifOutOctets: Number of data octets delivered by
--      X.25 to upper layers.

-- ifOutUcastPkts: Number of packets with a clear M-bit
-- received from higher layer entities.

-- #####
--                        X.25 Channel Table
-- #####

x25ChannelTable OBJECT-TYPE
    SYNTAX SEQUENCE OF X25ChannelEntry

```

```

ACCESS    not-accessible
STATUS    mandatory
DESCRIPTION
    "These objects contain information about the
    channel number configuration in an X.25 PLE.
    These values are the configured values.
    changes in these values after the interfaces
    has started may not be reflected in the
    operating PLE."
REFERENCE "See ISO 8208, Section 3.7"
::= { x25 4 }

```

```

x25ChannelEntry OBJECT-TYPE
    SYNTAX  X25ChannelEntry
    ACCESS  not-accessible
    STATUS  mandatory
    DESCRIPTION
        "Entries of x25ChannelTable."
    REFERENCE "This provides the information available
        in 10733 logicalChannelAssignments."
    INDEX { x25ChannelIndex }
    ::= { x25ChannelTable 1 }

```

```

X25ChannelEntry ::= SEQUENCE {
    x25ChannelIndex
        IfIndexType,
    x25ChannelLIC
        INTEGER,
    x25ChannelHIC
        INTEGER,
    x25ChannelLTC
        INTEGER,
    x25ChannelHTC
        INTEGER,
    x25ChannelLOC
        INTEGER,
    x25ChannelHOC
        INTEGER
}

```

```

x25ChannelIndex OBJECT-TYPE
    SYNTAX  IfIndexType
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The ifIndex value for the X.25 Interface."
    ::= { x25ChannelEntry 1 }

```

```
x25ChannelLIC OBJECT-TYPE
    SYNTAX  INTEGER (0..4095)
    ACCESS  read-write
    STATUS  mandatory
    DESCRIPTION
        "Lowest Incoming channel."
    ::= { x25ChannelEntry 2 }

x25ChannelHIC OBJECT-TYPE
    SYNTAX  INTEGER (0..4095)
    ACCESS  read-write
    STATUS  mandatory
    DESCRIPTION
        "Highest Incoming channel.  A value of zero
        indicates no channels in this range."
    ::= { x25ChannelEntry 3 }

x25ChannelLTC OBJECT-TYPE
    SYNTAX  INTEGER (0..4095)
    ACCESS  read-write
    STATUS  mandatory
    DESCRIPTION
        "Lowest Two-way channel."
    ::= { x25ChannelEntry 4 }

x25ChannelHTC OBJECT-TYPE
    SYNTAX  INTEGER (0..4095)
    ACCESS  read-write
    STATUS  mandatory
    DESCRIPTION
        "Highest Two-way channel.  A value of zero
        indicates no channels in this range."
    ::= { x25ChannelEntry 5 }

x25ChannelLOC OBJECT-TYPE
    SYNTAX  INTEGER (0..4095)
    ACCESS  read-write
    STATUS  mandatory
    DESCRIPTION
        "Lowest outgoing channel."
    ::= { x25ChannelEntry 6 }

x25ChannelHOC OBJECT-TYPE
    SYNTAX  INTEGER (0..4095)
    ACCESS  read-write
    STATUS  mandatory
    DESCRIPTION
        "Highest outgoing channel.  A value of zero
```


indicates no channels in this range."
 ::= { x25ChannelEntry 7 }

```
-- #####
--           X25 Per Circuits Information Table
-- #####
```

```
x25CircuitTable OBJECT-TYPE
    SYNTAX  SEQUENCE OF X25CircuitEntry
    ACCESS  not-accessible
    STATUS  mandatory
    DESCRIPTION
        "These objects contain general information
        about a specific circuit of an X.25 PLE."
    ::= { x25 5 }
```

```
x25CircuitEntry OBJECT-TYPE
    SYNTAX  X25CircuitEntry
    ACCESS  not-accessible
    STATUS  mandatory
    DESCRIPTION
        "Entries of x25CircuitTable."
    INDEX { x25CircuitIndex,
            x25CircuitChannel }
    ::= { x25CircuitTable 1 }
```

```
X25CircuitEntry ::= SEQUENCE {
    x25CircuitIndex
        IfIndexType,
    x25CircuitChannel
        INTEGER,
    x25CircuitStatus
        INTEGER,
    x25CircuitEstablishTime
        TimeTicks,
    x25CircuitDirection
        INTEGER,
    x25CircuitInOctets
        Counter,
    x25CircuitInPdus
        Counter,
    x25CircuitInRemotelyInitiatedResets
        Counter,
    x25CircuitInProviderInitiatedResets
        Counter,
```

```

x25CircuitInInterrupts
    Counter,
x25CircuitOutOctets
    Counter,
x25CircuitOutPdus
    Counter,
x25CircuitOutInterrupts
    Counter,
x25CircuitDataRetransmissionTimeouts
    Counter,
x25CircuitResetTimeouts
    Counter,
x25CircuitInterruptTimeouts
    Counter,
x25CircuitCallParamId
    OBJECT IDENTIFIER,
x25CircuitCalledDteAddress
    X121Address,
x25CircuitCallingDteAddress
    X121Address,
x25CircuitOriginallyCalledAddress
    X121Address,
x25CircuitDescr
    DisplayString
}

x25CircuitIndex OBJECT-TYPE
    SYNTAX  IfIndexType
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The ifIndex value for the X.25 Interface."
    ::= { x25CircuitEntry 1 }

x25CircuitChannel OBJECT-TYPE
    SYNTAX  INTEGER (0..4095)
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The channel number for this circuit."
    ::= { x25CircuitEntry 2 }

x25CircuitStatus OBJECT-TYPE
    SYNTAX  INTEGER {
        invalid (1),
        closed (2),
        calling (3),
        open (4),
    }
    -- state table states
    -- (p1)
    -- (p2,p3,p5)
    -- (p4)

```

```

        clearing (5),          -- (p6,p7)
        pvc (6),
        pvcResetting (7),
        startClear (8),       -- Close cmd
        startPvcResetting (9), -- Reset cmd
        other (10)
    }
ACCESS read-write
STATUS mandatory
DESCRIPTION
    "This object reports the current status of
    the circuit.

```

An existing instance of this object can only be set to startClear, startPvcResetting, or invalid. An instance with the value calling or open can only be set to startClear and that action will start clearing the circuit. An instance with the value PVC can only be set to startPvcResetting or invalid and that action resets the PVC or deletes the circuit respectively. The values startClear or startPvcResetting will never be returned by an agent. An attempt to set the status of an existing instance to a value other than one of these values will result in an error.

A non-existing instance can be set to PVC to create a PVC if the implementation supports dynamic creation of PVCs. Some implementations may only allow creation and deletion of PVCs if the interface is down. Since the instance identifier will supply the PLE index and the channel number, setting this object alone supplies sufficient information to create the instance. All the DEFVAL clauses for the other objects of this table are appropriate for creating a PVC; PLEs creating entries for placed or accepted calls will use values appropriate for the call rather than the value of the DEFVAL clause. Two managers trying to create the same PVC can determine from the return code which manager succeeded and which failed (the failing manager fails because it can not set a value of PVC for an existing object).

An entry in the closed or invalid state may be deleted or reused at the agent's convenience. If the entry is kept in the closed state, the values of the parameters associated with the entry must be correct. Closed implies the values in the circuit table are correct.

The value of invalid indicates the other values in the table are invalid. Many agents may never return a value of invalid because they dynamically allocate and free unused table entries. An agent for a statically configured systems can return invalid to indicate the entry has not yet been used so the counters contain no information."

REFERENCE "See ISO 8208,
table 33 for (p<n>) state table"
::= { x25CircuitEntry 3 }

x25CircuitEstablishTime OBJECT-TYPE

SYNTAX TimeTicks

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The value of sysUpTime when the channel was associated with this circuit. For outgoing SVCs, this is the time the first call packet was sent. For incoming SVCs, this is the time the call indication was received. For PVCs this is the time the PVC was able to pass data to a higher layer entity without loss of data."

::= { x25CircuitEntry 4 }

x25CircuitDirection OBJECT-TYPE

SYNTAX INTEGER {
incoming (1),
outgoing (2),
pvc (3)
}

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The direction of the call that established this circuit."

REFERENCE "10733 direction"

```
DEFVAL { pvc }
 ::= { x25CircuitEntry 5 }

    -- X25 Circuit data flow statistics

x25CircuitInOctets OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of octets of user data delivered
        to upper layer."
    REFERENCE "5.11 octetsReceivedCounter"
    ::= { x25CircuitEntry 6 }

x25CircuitInPdus OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of PDUs received for this
        circuit."
    REFERENCE "10733 5.11 dataPacketsReceived"
    ::= { x25CircuitEntry 7 }

x25CircuitInRemotelyInitiatedResets OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of Resets received for this
        circuit with cause code of DTE initiated."
    REFERENCE "10733 remotelyInitiatedResets"
    ::= { x25CircuitEntry 8 }

x25CircuitInProviderInitiatedResets OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of Resets received for this
        circuit with cause code other than DTE
        initiated."
    REFERENCE "10733 ProviderInitiatedResets"
    ::= { x25CircuitEntry 9 }

x25CircuitInInterrupts OBJECT-TYPE
    SYNTAX Counter
```

```
ACCESS    read-only
STATUS    mandatory
DESCRIPTION
    "The number of interrupt packets received
    for this circuit."
REFERENCE "10733 interruptPacketsReceived"
::= { x25CircuitEntry 10 }

x25CircuitOutOctets OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of octets of user data sent for
        this circuit."
    REFERENCE "10733 5.11 octetsSentCounter"
    ::= { x25CircuitEntry 11 }

x25CircuitOutPdus OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of PDUs sent for this circuit."
    REFERENCE "10733 5.11 dataPacketsSent"
    ::= { x25CircuitEntry 12 }

x25CircuitOutInterrupts OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of interrupt packets sent on
        this circuit."
    REFERENCE "10733 interruptPacketsSent"
    ::= { x25CircuitEntry 13 }

-- X25 circuit timer statistics

x25CircuitDataRetransmissionTimeouts OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of times the T25 data
        retransmission timer expired for this
        circuit."
```

REFERENCE "10733 5.11 dataRetransmissionTimerExpiries"
::= { x25CircuitEntry 14 }

x25CircuitResetTimeouts OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of times the T22 reset timer
expired for this circuit."

REFERENCE "10733 5.11 resetTimeouts"

::= { x25CircuitEntry 15 }

x25CircuitInterruptTimeouts OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of times the T26 Interrupt timer
expired for this circuit."

REFERENCE "10733 interruptTimerExpiries"

::= { x25CircuitEntry 16 }

x25CircuitCallParamId OBJECT-TYPE

SYNTAX OBJECT IDENTIFIER

ACCESS read-write

STATUS mandatory

DESCRIPTION

"This identifies the instance of the
x25CallParmIndex for the entry in the
x25CallParmTable which contains the call
parameters in use with this circuit. The
entry referenced must contain the values
that are currently in use by the circuit
rather than proposed values. A value of
NULL indicates the circuit is a PVC or is
using all the default parameters."

DEFVAL { {0 0} }

::= { x25CircuitEntry 17 }

x25CircuitCalledDteAddress OBJECT-TYPE

SYNTAX X121Address

ACCESS read-write

STATUS mandatory

DESCRIPTION

"For incoming calls, this is the called
address from the call indication packet.
For outgoing calls, this is the called

```

        address from the call confirmation packet.
        This will be zero length for PVCs."
REFERENCE "10733 calledDTEAddress"
DEFVAL { ''h }
::= { x25CircuitEntry 18 }

x25CircuitCallingDteAddress OBJECT-TYPE
    SYNTAX  X121Address
    ACCESS  read-write
    STATUS  mandatory
    DESCRIPTION
        "For incoming calls, this is the calling
        address from the call indication packet.
        For outgoing calls, this is the calling
        address from the call confirmation packet.
        This will be zero length for PVCs."
REFERENCE "10733 callingDTEAddress"
DEFVAL { ''h }
::= { x25CircuitEntry 19 }

x25CircuitOriginallyCalledAddress OBJECT-TYPE
    SYNTAX  X121Address
    ACCESS  read-write
    STATUS  mandatory
    DESCRIPTION
        "For incoming calls, this is the address in
        the call Redirection or Call Deflection
        Notification facility if the call was
        deflected or redirected, otherwise it will
        be called address from the call indication
        packet.  For outgoing calls, this is the
        address from the call request packet.  This
        will be zero length for PVCs."
REFERENCE "10733 originallyCalledAddress"
DEFVAL { ''h }
::= { x25CircuitEntry 20 }

x25CircuitDescr OBJECT-TYPE
    SYNTAX  DisplayString (SIZE (0..255))
    ACCESS  read-write
    STATUS  mandatory
    DESCRIPTION
        "A descriptive string associated with this
        circuit.  This provides a place for the
        agent to supply any descriptive information
        it knows about the use or owner of the
        circuit.  The agent may return the process
        identifier and user name for the process

```


using the circuit. Alternative the agent may return the name of the configuration entry that caused a bridge to establish the circuit. A zero length value indicates the agent doesn't have any additional information."

```
DEFVAL { ''h }
::= { x25CircuitEntry 21 }
```

```
-- #####
--                               The Cleared Circuit Table
-- #####
```

x25ClearedCircuitEntriesRequested OBJECT-TYPE

SYNTAX PositiveInteger

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The requested number of entries for the agent to keep in the x25ClearedCircuit table."

```
::= { x25 6 }
```

x25ClearedCircuitEntriesGranted OBJECT-TYPE

SYNTAX PositiveInteger

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The actual number of entries the agent will keep in the x25ClearedCircuit Table."

```
::= { x25 7 }
```

x25ClearedCircuitTable OBJECT-TYPE

SYNTAX SEQUENCE OF X25ClearedCircuitEntry

ACCESS not-accessible

STATUS mandatory

DESCRIPTION

"A table of entries about closed circuits. Entries must be made in this table whenever circuits are closed and the close request or close indication packet contains a clearing cause other than DTE Originated or a Diagnostic code field other than Higher Layer Initiated disconnection-normal. An agent may optionally make entries for normal closes (to record closing facilities or

other information).

Agents will delete the oldest entry in the table when adding a new entry would exceed agent resources. Agents are required to keep the last entry put in the table and may keep more entries. The object x25OperClearEntriesGranted returns the maximum number of entries kept in the table."

REFERENCE "See ISO 8208 Section 12.2.3.1.1
and 12.2.3.1.2"

::= { x25 8 }

x25ClearedCircuitEntry OBJECT-TYPE
SYNTAX X25ClearedCircuitEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION
"Information about a cleared circuit."
INDEX { x25ClearedCircuitIndex }
::= { x25ClearedCircuitTable 1 }

X25ClearedCircuitEntry ::= SEQUENCE {
x25ClearedCircuitIndex
PositiveInteger,
x25ClearedCircuitPleIndex
IfIndexType,
x25ClearedCircuitTimeEstablished
TimeTicks,
x25ClearedCircuitTimeCleared
TimeTicks,
x25ClearedCircuitChannel
INTEGER,
x25ClearedCircuitClearingCause
INTEGER,
x25ClearedCircuitDiagnosticCode
INTEGER,
x25ClearedCircuitInPdus
Counter,
x25ClearedCircuitOutPdus
Counter,
x25ClearedCircuitCalledAddress
X121Address,
x25ClearedCircuitCallingAddress
X121Address,
x25ClearedCircuitClearFacilities
OCTET STRING

```

    }

x25ClearedCircuitIndex OBJECT-TYPE
    SYNTAX  PositiveInteger
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "An index that uniquely distinguishes one
        entry in the clearedCircuitTable from
        another.  This index will start at
        2147483647 and will decrease by one for each
        new entry added to the table.  Upon reaching
        one, the index will reset to 2147483647.
        Because the index starts at 2147483647 and
        decreases, a manager may do a getnext on
        entry zero and obtain the most recent entry.
        When the index has the value of 1, the next
        entry will delete all entries in the table
        and that entry will be numbered 2147483647."
    ::= { x25ClearedCircuitEntry 1 }

x25ClearedCircuitPleIndex OBJECT-TYPE
    SYNTAX  IfIndexType
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The value of ifIndex for the PLE which
        cleared the circuit that created the entry."
    ::= { x25ClearedCircuitEntry 2 }

x25ClearedCircuitTimeEstablished OBJECT-TYPE
    SYNTAX  TimeTicks
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The value of sysUpTime when the circuit was
        established.  This will be the same value
        that was in the x25CircuitEstablishTime for
        the circuit."
    ::= { x25ClearedCircuitEntry 3 }

x25ClearedCircuitTimeCleared OBJECT-TYPE
    SYNTAX  TimeTicks
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The value of sysUpTime when the circuit was
        cleared.  For locally initiated clears, this

```

will be the time when the clear confirmation was received. For remotely initiated clears, this will be the time when the clear indication was received."

::= { x25ClearedCircuitEntry 4 }

x25ClearedCircuitChannel OBJECT-TYPE

SYNTAX INTEGER (0..4095)

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The channel number for the circuit that was cleared."

::= { x25ClearedCircuitEntry 5 }

x25ClearedCircuitClearingCause OBJECT-TYPE

SYNTAX INTEGER (0..255)

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The Clearing Cause from the clear request or clear indication packet that cleared the circuit."

REFERENCE "See ISO 8208 Section 12.2.3.1.1"

::= { x25ClearedCircuitEntry 6 }

x25ClearedCircuitDiagnosticCode OBJECT-TYPE

SYNTAX INTEGER (0..255)

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The Diagnostic Code from the clear request or clear indication packet that cleared the circuit."

REFERENCE "See ISO 8208 Section 12.2.3.1.2"

::= { x25ClearedCircuitEntry 7 }

x25ClearedCircuitInPdus OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of PDUs received on the circuit."

::= { x25ClearedCircuitEntry 8 }

x25ClearedCircuitOutPdus OBJECT-TYPE

SYNTAX Counter

```

ACCESS    read-only
STATUS    mandatory
DESCRIPTION
    "The number of PDUs transmitted on the
    circuit."
 ::= { x25ClearedCircuitEntry 9 }

x25ClearedCircuitCalledAddress OBJECT-TYPE
SYNTAX    X121Address
ACCESS    read-only
STATUS    mandatory
DESCRIPTION
    "The called address from the cleared
    circuit."
 ::= { x25ClearedCircuitEntry 10 }

x25ClearedCircuitCallingAddress OBJECT-TYPE
SYNTAX    X121Address
ACCESS    read-only
STATUS    mandatory
DESCRIPTION
    "The calling address from the cleared
    circuit."
 ::= { x25ClearedCircuitEntry 11 }

x25ClearedCircuitClearFacilities OBJECT-TYPE
SYNTAX    OCTET STRING (SIZE (0..109))
ACCESS    read-only
STATUS    mandatory
DESCRIPTION
    "The facilities field from the clear request
    or clear indication packet that cleared the
    circuit.  A size of zero indicates no
    facilities were present."
 ::= { x25ClearedCircuitEntry 12 }

-- #####
--                               The Call Parameter Table
-- #####

x25CallParmTable OBJECT-TYPE
SYNTAX    SEQUENCE OF X25CallParmEntry
ACCESS    not-accessible
STATUS    mandatory
DESCRIPTION

```

"These objects contain the parameters that can be varied between X.25 calls. The entries in this table are independent of the PLE. There exists only one of these tables for the entire system. The indexes for the entries are independent of any PLE or any circuit. Other tables reference entries in this table. Entries in this table can be used for default PLE parameters, for parameters to use to place/answer a call, for the parameters currently in use for a circuit, or parameters that were used by a circuit.

The number of references to a given set of parameters can be found in the x25CallParmRefCount object sharing the same instance identifier with the parameters. The value of this reference count also affects the access of the objects in this table. An object in this table with the same instance identifier as the instance identifier of an x25CallParmRefCount must be considered associated with that reference count. An object with an associated reference count of zero can be written (if its ACCESS clause allows it). An object with an associated reference count greater than zero can not be written (regardless of the ACCESS clause). This ensures that a set of call parameters being referenced from another table can not be modified or changed in a way inappropriate for continued use by that table."

```
::= { x25 9 }
```

```
x25CallParmEntry OBJECT-TYPE
```

```
SYNTAX X25CallParmEntry
```

```
ACCESS not-accessible
```

```
STATUS mandatory
```

```
DESCRIPTION
```

```
"Entries of x25CallParmTable."
```

```
INDEX { x25CallParmIndex }
```

```
::= { x25CallParmTable 1 }
```

```
X25CallParmEntry ::= SEQUENCE {
```

```
    x25CallParmIndex
```

```
    PositiveInteger,
```

```
x25CallParmStatus
    EntryStatus,
x25CallParmRefCount
    PositiveInteger,
x25CallParmInPacketSize
    INTEGER,
x25CallParmOutPacketSize
    INTEGER,
x25CallParmInWindowSize
    INTEGER,
x25CallParmOutWindowSize
    INTEGER,
x25CallParmAcceptReverseCharging
    INTEGER,
x25CallParmProposeReverseCharging
    INTEGER,
x25CallParmFastSelect
    INTEGER,
x25CallParmInThruPutClasSize
    INTEGER,
x25CallParmOutThruPutClasSize
    INTEGER,
x25CallParmCug
    DisplayString,
x25CallParmCugoa
    DisplayString,
x25CallParmBcug
    DisplayString,
x25CallParmNui
    OCTET STRING,
x25CallParmChargingInfo
    INTEGER,
x25CallParmRpoa
    DisplayString,
x25CallParmTrnstDly
    INTEGER,
x25CallParmCallingExt
    DisplayString,
x25CallParmCalledExt
    DisplayString,
x25CallParmInMinThuPutCls
    INTEGER,
x25CallParmOutMinThuPutCls
    INTEGER,
x25CallParmEndTrnsDly
    OCTET STRING,
x25CallParmPriority
    OCTET STRING,
```

```
x25CallParmProtection
    DisplayString,
x25CallParmExptData
    INTEGER,
x25CallParmUserData
    OCTET STRING,
x25CallParmCallingNetworkFacilities
    OCTET STRING,
x25CallParmCalledNetworkFacilities
    OCTET STRING
}
```

x25CallParmIndex OBJECT-TYPE

SYNTAX PositiveInteger

ACCESS read-only

STATUS mandatory

DESCRIPTION

"A value that distinguishes this entry from another entry. Entries in this table are referenced from other objects which identify call parameters.

It is impossible to know which other objects in the MIB reference entries in the table by looking at this table. Because of this, changes to parameters must be accomplished by creating a new entry in this table and then changing the referencing table to identify the new entry.

Note that an agent will only use the values in this table when another table is changed to reference those values. The number of other tables that reference an index object in this table can be found in x25CallParmRefCount. The value of the reference count will affect the writability of the objects as explained above.

Entries in this table which have a reference count of zero maybe deleted at the convenue of the agent. Care should be taken by the agent to give the NMS sufficient time to create a reference to newly created entries.

Should a Management Station not find a free index with which to create a new entry, it may feel free to delete entries with a

reference count of zero. However in doing so the Management Station much realize it may impact other Management Stations."

```
 ::= { x25CallParmEntry 1 }
```

x25CallParmStatus OBJECT-TYPE

SYNTAX EntryStatus

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The status of this call parameter entry.

See RFC 1271 for details of usage."

```
 ::= { x25CallParmEntry 2 }
```

x25CallParmRefCount OBJECT-TYPE

SYNTAX PositiveInteger

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of references know by a management station to exist to this set of call parameters. This is the number of other objects that have returned a value of, and will return a value of, the index for this set of call parameters. Examples of such objects are the x25AdmnDefCallParamId, x25OperDataLinkId, or x25AdmnDefCallParamId objects defined above."

```
 ::= { x25CallParmEntry 3 }
```

x25CallParmInPacketSize OBJECT-TYPE

SYNTAX INTEGER (0..4096)

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The maximum receive packet size in octets for a circuit. A size of zero for a circuit means use the PLE default size. A size of zero for the PLE means use a default size of 128."

REFERENCE "10733 proposedPacketSize;

See ISO 8208 Section 15.2.2.1.1"

DEFVAL { 128 }

```
 ::= { x25CallParmEntry 4 }
```

x25CallParmOutPacketSize OBJECT-TYPE

SYNTAX INTEGER (0..4096)

ACCESS read-write

```
STATUS    mandatory
DESCRIPTION
    "The maximum transmit packet size in octets
    for a circuit.  A size of zero for a circuit
    means use the PLE default size.  A size of
    zero for the PLE default means use a default
    size of 128."
REFERENCE "10733 proposedPacketSize;
    See ISO 8208 Section 15.2.2.1.1"
DEFVAL { 128 }
::= { x25CallParmEntry 5 }

x25CallParmInWindowSize OBJECT-TYPE
SYNTAX    INTEGER (0..127)
ACCESS    read-write
STATUS    mandatory
DESCRIPTION
    "The receive window size for a circuit.  A
    size of zero for a circuit means use the PLE
    default size.  A size of zero for the PLE
    default means use 2."
REFERENCE "10733 proposedWindowSize;
    See ISO 8208 Section 15.2.2.1.2"
DEFVAL { 2 }
::= { x25CallParmEntry 6 }

x25CallParmOutWindowSize OBJECT-TYPE
SYNTAX    INTEGER (0..127)
ACCESS    read-write
STATUS    mandatory
DESCRIPTION
    "The transmit window size for a circuit.  A
    size of zero for a circuit means use the PLE
    default size.  A size of zero for the PLE
    default means use 2."
REFERENCE "10733 proposedWindowSize;
    See ISO 8208 Section 15.2.2.1.2"
DEFVAL { 2 }
::= { x25CallParmEntry 7 }

x25CallParmAcceptReverseCharging OBJECT-TYPE
SYNTAX    INTEGER {
        default (1),
        accept (2),
        refuse (3),
        neverAccept (4)
    }
ACCESS    read-write
```

STATUS mandatory
DESCRIPTION

"An enumeration defining if the PLE will accept or refuse charges. A value of default for a circuit means use the PLE default value. A value of neverAccept is only used for the PLE default and indicates the PLE will never accept reverse charging. A value of default for a PLE default means refuse."

REFERENCE "10733 acceptReverseCharging"

DEFVAL { refuse }

::= { x25CallParmEntry 8 }

x25CallParmProposeReverseCharging OBJECT-TYPE

SYNTAX INTEGER {
 default (1),
 reverse (2),
 local (3)
}

ACCESS read-write

STATUS mandatory

DESCRIPTION

"An enumeration defining if the PLE should propose reverse or local charging. The value of default for a circuit means use the PLE default. The value of default for the PLE default means use local."

REFERENCE "10733 proposedPacketSize;

See ISO 8208 Section 15.2.2.6"

DEFVAL { local }

::= { x25CallParmEntry 9 }

x25CallParmFastSelect OBJECT-TYPE

SYNTAX INTEGER {
 default (1),
 notSpecified (2),
 fastSelect (3),
 restrictedFastResponse (4),
 noFastSelect (5),
 noRestrictedFastResponse (6)
}

ACCESS read-write

STATUS mandatory

DESCRIPTION

"Expresses preference for use of fast select facility. The value of default for a circuit is the PLE default. A value of

default for the PLE means noFastSelect. A value of noFastSelect or noRestrictedFastResponse indicates a circuit may not use fast select or restricted fast response."

REFERENCE "10733 fastSelect;
Sec ISO 8208 Section 15.2.2.6"
DEFVAL { noFastSelect }
::= { x25CallParmEntry 10 }

x25CallParmInThruPutClasSize OBJECT-TYPE

SYNTAX INTEGER {
tcReserved1 (1),
tcReserved2 (2),
tc75 (3),
tc150 (4),
tc300 (5),
tc600 (6),
tc1200 (7),
tc2400 (8),
tc4800 (9),
tc9600 (10),
tc19200 (11),
tc48000 (12),
tc64000 (13),
tcReserved14 (14),
tcReserved15 (15),
tcReserved0 (16),
tcNone (17),
tcDefault (18)
}

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The incoming throughput class to negotiate. A value of tcDefault for a circuit means use the PLE default. A value of tcDefault for the PLE default means tcNone. A value of tcNone means do not negotiate throughput class."

REFERENCE "See ISO 8208 Section 15.2.2.2, table 18"
DEFVAL { tcNone }
::= { x25CallParmEntry 11 }

x25CallParmOutThruPutClasSize OBJECT-TYPE

SYNTAX INTEGER {
tcReserved1 (1),
tcReserved2 (2),

```

        tc75 (3),
        tc150 (4),
        tc300 (5),
    tc600 (6),
        tc1200 (7),
        tc2400 (8),
        tc4800 (9),
        tc9600 (10),
        tc19200 (11),
        tc48000 (12),
        tc64000 (13),
        tcReserved14 (14),
        tcReserved15 (15),
        tcReserved0 (16),
        tcNone (17),
        tcDefault (18)
    }
ACCESS read-write
STATUS mandatory
DESCRIPTION
    "The outgoing throughput class to negotiate.
    A value of tcDefault for a circuit means use
    the PLE default. A value of tcDefault for
    the PLE default means use tcNone. A value
    of tcNone means do not negotiate throughput
    class."
REFERENCE "See ISO 8208 Section 15.2.2.2, table 18"
DEFVAL { tcNone }
::= { x25CallParmEntry 12 }

x25CallParmCug OBJECT-TYPE
SYNTAX DisplayString (SIZE(0..4))
ACCESS read-write
STATUS mandatory
DESCRIPTION
    "The Closed User Group to specify. This
    consists of two or four octets containing
    the characters 0 through 9. A zero length
    string indicates no facility requested. A
    string length of three containing the
    characters DEF for a circuit means use the
    PLE default, (the PLE default parameter may
    not reference an entry of DEF.)"
REFERENCE "See ISO 8208 Section 15.2.2.3"
DEFVAL { ''h }
::= { x25CallParmEntry 13 }

x25CallParmCugoa OBJECT-TYPE

```

```

SYNTAX  DisplayString (SIZE(0..4))
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
    "The Closed User Group with Outgoing Access
    to specify. This consists of two or four
    octets containing the characters 0 through
    9. A string length of three containing the
    characters DEF for a circuit means use the
    PLE default (the PLE default parameters may
    not reference an entry of DEF). A zero
    length string indicates no facility
    requested."
REFERENCE "See ISO 8208 Section 15.2.2.4"
DEFVAL { ''h }
 ::= { x25CallParmEntry 14 }

```

```

x25CallParmBcug OBJECT-TYPE
    SYNTAX  DisplayString (SIZE(0..3))
    ACCESS  read-write
    STATUS  mandatory
    DESCRIPTION
        "The Bilateral Closed User Group to specify.
        This consists of two octets containing the
        characters 0 through 9. A string length of
        three containing the characters DEF for a
        circuit means use the PLE default (the PLE
        default parameter may not reference an entry
        of DEF). A zero length string indicates no
        facility requested."
    REFERENCE "See ISO 8208 Section 15.2.2.5"
    DEFVAL { ''h }
    ::= { x25CallParmEntry 15 }

```

```

x25CallParmNui OBJECT-TYPE
    SYNTAX  OCTET STRING (SIZE(0..108))
    ACCESS  read-write
    STATUS  mandatory
    DESCRIPTION
        "The Network User Identifier facility. This
        is binary value to be included immediately
        after the length field. The PLE will supply
        the length octet. A zero length string
        indicates no facility requested. This value
        is ignored for the PLE default parameters
        entry."
    REFERENCE "See ISO 8208 Section 15.2.2.7"
    DEFVAL { ''h }

```

```

 ::= { x25CallParmEntry 16 }

x25CallParmChargingInfo OBJECT-TYPE
    SYNTAX INTEGER {
        default (1),
        noFacility (2),
        noChargingInfo (3),
        chargingInfo (4)
    }
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "The charging Information facility. A value
        of default for a circuit means use the PLE
        default. The value of default for the
        default PLE parameters means use noFacility.
        The value of noFacility means do not include
        a facility."
    REFERENCE "See ISO 8208 Section 15.2.2.8"
    DEFVAL { noFacility }
    ::= { x25CallParmEntry 17 }

x25CallParmRpoa OBJECT-TYPE
    SYNTAX DisplayString (SIZE(0..108))
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "The RPOA facility. The octet string
        contains n * 4 sequences of the characters
        0-9 to specify a facility with n entries.
        The octet string containing the 3 characters
        DEF for a circuit specifies use of the PLE
        default (the entry for the PLE default may
        not contain DEF). A zero length string
        indicates no facility requested."
    REFERENCE "See ISO 8208, section 15.2.2.9"
    DEFVAL { ''h }
    ::= { x25CallParmEntry 18 }

x25CallParmTrnstDly OBJECT-TYPE
    SYNTAX INTEGER (0..65537)
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "The Transit Delay Selection and Indication
        value. A value of 65536 indicates no
        facility requested. A value of 65537 for a
        circuit means use the PLE default (the PLE

```

default parameters entry may not use the value 65537). The value 65535 may only be used to indicate the value in use by a circuit."

REFERENCE "See ISO 8208, Section 15.2.2.13"

DEFVAL { 65536 }

::= { x25CallParmEntry 19 }

-- The following parameters are for CCITT facilities.

x25CallParmCallingExt OBJECT-TYPE

SYNTAX DisplayString (SIZE(0..40))

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The Calling Extension facility. This contains one of the following:

A sequence of hex digits with the value to be put in the facility. These digits will be converted to binary by the agent and put in the facility. These octets do not include the length octet.

A value containing the three character DEF for a circuit means use the PLE default, (the entry for the PLE default parameters may not use the value DEF).

A zero length string indicates no facility requested."

REFERENCE "See ISO 8208 Section 15.3.2.1"

DEFVAL { ''h }

::= { x25CallParmEntry 20 }

x25CallParmCalledExt OBJECT-TYPE

SYNTAX DisplayString (SIZE(0..40))

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The Called Extension facility. This contains one of the following:

A sequence of hex digits with the value to be put in the facility. These digits will be converted to binary by the agent and put in the facility. These octets do not include

the length octet.

A value containing the three character DEF for a circuit means use the PLE default, (the entry for the PLE default parameters may not use the value DEF).

A zero length string indicates no facility requested."

REFERENCE "See ISO 8208 Section 15.3.2.2"

DEFVAL { 'h' }

::= { x25CallParmEntry 21 }

x25CallParmInMinThuPutCls OBJECT-TYPE

SYNTAX INTEGER (0..17)

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The minimum input throughput Class. A value of 16 for a circuit means use the PLE default (the PLE parameters entry may not use this value). A value of 17 indicates no facility requested."

REFERENCE "See ISO 8208 Section 15.3.2.3"

DEFVAL { 17 }

::= { x25CallParmEntry 22 }

x25CallParmOutMinThuPutCls OBJECT-TYPE

SYNTAX INTEGER (0..17)

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The minimum output throughput Class. A value of 16 for a circuit means use the PLE default (the PLE parameters entry may not use this value). A value of 17 indicates no facility requested."

REFERENCE "See ISO 8208 Section 15.3.2.3"

DEFVAL { 17 }

::= { x25CallParmEntry 23 }

x25CallParmEndTrnsDly OBJECT-TYPE

SYNTAX OCTET STRING (SIZE(0..6))

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The End-to-End Transit Delay to negotiate. An octet string of length 2, 4, or 6

contains the facility encoded as specified in ISO/IEC 8208 section 15.3.2.4. An octet string of length 3 containing the three character DEF for a circuit means use the PLE default (the entry for the PLE default can not contain the characters DEF). A zero length string indicates no facility requested."

REFERENCE "See ISO 8208 Section 15.3.2.4"

DEFVAL { 'h }

::= { x25CallParmEntry 24 }

x25CallParmPriority OBJECT-TYPE

SYNTAX OCTET STRING (SIZE(0..6))

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The priority facility to negotiate. The octet string encoded as specified in ISO/IEC 8208 section 15.3.2.5. A zero length string indicates no facility requested. The entry for the PLE default parameters must be zero length."

REFERENCE "See ISO 8208 Section 15.3.2.5"

DEFVAL { 'h }

::= { x25CallParmEntry 25 }

x25CallParmProtection OBJECT-TYPE

SYNTAX DisplayString (SIZE(0..108))

ACCESS read-write

STATUS mandatory

DESCRIPTION

"A string contains the following:
A hex string containing the value for the protection facility. This will be converted from hex to the octets actually in the packet by the agent. The agent will supply the length field and the length octet is not contained in this string.

An string containing the 3 characters DEF for a circuit means use the PLE default (the entry for the PLE default parameters may not use the value DEF).

A zero length string mean no facility requested."

REFERENCE "See ISO 8208 Section 15.3.2.5"

```

DEFVAL { ''h }
::= { x25CallParmEntry 26 }

x25CallParmExptData OBJECT-TYPE
    SYNTAX  INTEGER {
        default (1),
        noExpeditedData (2),
        expeditedData (3)
    }
    ACCESS  read-write
    STATUS  mandatory
    DESCRIPTION
        "The Expedited Data facility to negotiate.
        A value of default for a circuit means use
        the PLE default value. The entry for the
        PLE default parameters may not have the
        value default."
    REFERENCE "See ISO 8208 Section 15.3.2.7"
    DEFVAL { noExpeditedData }
    ::= { x25CallParmEntry 27 }

x25CallParmUserData      OBJECT-TYPE
    SYNTAX  OCTET STRING (SIZE (0..128))
    ACCESS  read-write
    STATUS  mandatory
    DESCRIPTION
        "The call user data as placed in the packet.
        A zero length string indicates no call user
        data. If both the circuit call parameters
        and the PLE default have call user data
        defined, the data from the circuit call
        parameters will be used. If only the PLE
        has data defined, the PLE entry will be
        used. If neither the circuit call
        parameters or the PLE default entry has a
        value, no call user data will be sent."
    REFERENCE "See ISO 8208 Section 12.2.1.1.6, 12.2.1.2"
    DEFVAL { ''h }
    ::= { x25CallParmEntry 28 }

x25CallParmCallingNetworkFacilities OBJECT-TYPE
    SYNTAX  OCTET STRING (SIZE (0..108))
    ACCESS  read-write
    STATUS  mandatory
    DESCRIPTION
        "The calling network facilities. The
        facilities are encoded here exactly as
        encoded in the call packet. These

```

facilities do not include the marker facility code.

A zero length string in the entry for the parameter to use when establishing a circuit means use the PLE default. A zero length string in the entry for PLE default parameters indicates no default facilities."

REFERENCE "See ISO 8206 Section 15.1, category b"

DEFVAL { ''h }

::= { x25CallParmEntry 29 }

x25CallParmCalledNetworkFacilities OBJECT-TYPE

SYNTAX OCTET STRING (SIZE (0..108))

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The called network facilities. The facilities are encoded here exactly as encoded in the call packet. These facilities do not include the marker facility code.

A zero length string in the entry for the parameter to use when establishing a circuit means use the PLE default. A zero length string in the entry for PLE default parameters indicates no default facilities."

REFERENCE "See ISO 8206 Section 15.1, category c"

DEFVAL { ''h }

::= { x25CallParmEntry 30 }

```
-- #####
-- X.25 Traps
-- #####
```

x25Restart TRAP-TYPE

ENTERPRISE x25

VARIABLES { x25OperIndex }

DESCRIPTION

"This trap means the X.25 PLE sent or received a restart packet. The restart that brings up the link should not send a x25Restart trap so the interface should send a linkUp trap. Sending this trap means the agent does not send a linkDown and linkUp trap."

::= 1

```

x25Reset          TRAP-TYPE
                  ENTERPRISE  x25
                  VARIABLES { x25CircuitIndex,
                              x25CircuitChannel }
                  DESCRIPTION
                      "If the PLE sends or receives a reset, the
                       agent should send an x25Reset trap."
                  ::= 2

-- #####
--                               X.25 Protocol Version Identifiers
-- #####

x25ProtocolVersion OBJECT IDENTIFIER
    ::= { x25 10 }

    -- X.25 CCITT 1976 version.
x25protocolCcittV1976 OBJECT IDENTIFIER
    ::= { x25ProtocolVersion 1 }

    -- X.25 CCITT 1980 version.
x25protocolCcittV1980 OBJECT IDENTIFIER
    ::= { x25ProtocolVersion 2 }

    -- X.25 CCITT 1984 version.
x25protocolCcittV1984 OBJECT IDENTIFIER
    ::= { x25ProtocolVersion 3 }

    -- X.25 CCITT 1988 version.
x25protocolCcittV1988 OBJECT IDENTIFIER
    ::= { x25ProtocolVersion 4 }

    -- X.25 1987 version of ISO 8208.
x25protocolIso8208V1987 OBJECT IDENTIFIER
    ::= { x25ProtocolVersion 5 }

    -- X.25 1989 version of ISO 8208.
x25protocolIso8208V1989 OBJECT IDENTIFIER
    ::= { x25ProtocolVersion 6 }

-- #####

END

```

5. Appendix: Revision History

July 30 1992

The July, 1992 release (Editor's Internal Reference Number 2.14) made the following changes:

The syntax of the index objects for tables that are congruent with the MIB-II ifTable were changed to ifIndexType.

The x25CallParmRefCount object was added to the x25CallParmTable.

The description of the x25CallParmTable and x25CallParmIndex objects were changed to only allow writing an entry with a zero reference count.

A requirement for conformance was added after the definition of x25 in the ASN.1 definition.

June 26 1992

The June 29, 1992 release (Editor's Internal Reference Number 2.12) made the following changes:

The range of x25ChannelLIC was changed from (0..4096) to (0..4095).

The range of x25ChannelHIC was changed from (0..4096) to (0..4095).

The range of x25ChannelLTC was changed from (0..4096) to (0..4095).

The range of x25ChannelHTC was changed from (0..4096) to (0..4095).

The range of x25ChannelLOC was changed from (0..4096) to (0..4095).

The range of x25ChannelHOC was changed from (0..4096) to (0..4095).

The range of x25CircuitChannel was changed from (1..4096) to (0..4095).

The range of x25ClearedCircuitChannel was changed from

(1..4096) to (0..4095).

June 1992

The June 92 release (Editor's Internal Reference Number 2.11) made the following changes:

A value of dxs was defined for x25AdmnInterfaceMode and x25OperInterfaceMode.

The objects in the x25ChannelTable can now have a value of zero to indicate no channels configured in the range.

The length of an X121Address was extended to 17 to accommodate the 1988 CCITT X.25 standard.

Some object descriptions have been expanded and simplified, these include: all the channel table objects except the index, x25AdmnDataRxmtCount, x25AdmnRejectCount, x25AdmnRegistrationRequestCount, x25OperDataRxmtCount, x25OperRejectCount, x25OperRegistrationRequestCount, x25CircuitEstablishTime, x25ClearedCircuitTimeEstablished, x25ClearedCircuitTimeCleared, x25CallParmIndex, x25CallParmInPacketSize, x25CircuitCalledAddress, x25CircuitOriginalCalledAddress, x25CircuitCallingAddress, x25CallParmFastSelect, x25CallParmCug, x25CallParmCugoa, x25CallParmBcug, x25CallParmNui, x25CallParmRpoa, x25CallParmCallingExt, x25CallParmCalledExt, x25CallParmProtection, x25StatInCallRefusals and x25CallParmOutPacketSize.

The x25StatNumberPvcs object was deleted and x25AdmnNumberPVCs and x25OperNumberPVCs objects added.

The object x25StatOutDataPackets was added.

The object x25AdmnProtocolVersionSupported as added.

The x25CircuitRemoteDteAddress was deleted.

Some ASN.1 errors were corrected.

April 1992

The April release (Editor's Internal Reference Number 2.8) made many changes to incorporate the comments of the working group meeting in March 1992.

All reference comments were changed to reference fields.

The type PositiveInteger was imported from the RFC1381-MIB and used for all index and timer values.

The x25PleTable was split into the x25AdmnTable, x25OperTable, and x25StatTable.

The timer and counter objects from the x25CircuitTable were moved to the x25AdmnTable and replicated in the x25OperTable

The objects in the x25CircuitTable were reordered to put the non-integer objects at the end of the table for easier implementation.

The called and calling extension character set was extended to include a-f, and A-F.

Additional states were added to the x25CircuitStatus object.

Additional values were added to x25CircuitDirection, x25CircuitCallParamId, and the addresses in the Circuit Table for PVCs.

The length of the X25Address was changed to 0..15.

The objects x25ClearedCircuitTimeEstablished, x25ClearedCircuitInPdus, and x25ClearedCircuitOutPdus were added to the x25ClearedCircuitTable.

The name of the x25CircuitName was changed to x25CircuitDescr and the description was expanded.

The access of the x25CircuitCallParamId was changed to read-only.

The x25ClearedCircuitCodes object was split into the x25ClearedCircuitClearingCause and x25ClearedCircuitDiagnosticCode objects.

The semantics of the x25ClearedCircuitIndex was redefined.

Some of the description clauses were changed in an attempt to add clarity.

DEFVAL clauses were added to most objects in the x25CallParmTable.

Additional text was added to the description section to provide an overview of the tables of the MIB.

The minimum allowable value for maximum active circuits was changed from one to zero.

February 1992

The February release (Editor's Internal Reference Number 1.14) made many changes.

Many of the tables were combined. For example, the x25InfoTable, x25PktStatTable, and x25TmrStatTable were combined into the x25PleTable. The x25ConInfoTable, x25ConStatTable, and x25ConTimrTable were combined into the x25CircuitTable.

The objects for call parameters were drastically reworked. All call parameters were combined in the x25CallParmTable. Any table, such as the x25PleTable or x25CircuitTable, that needs to reference call parameters identifies an entry in the new table. As part of this the x25ConDefTable was deleted and replaced with the x25PleDefCallParamId.

The x25PvcTable was deleted; the x25CircuitStatus object provides similar information about PVCs.

The x25ClearedCircuitTable was added to record the status code of cleared circuits.

Many object definitions were restructured. For example, the time units for timers was changed from 1/100ths of a second to milliseconds. Some indexes into tables were replaced with object identifiers.

Much of the introductory text was changed and the references were changed to match.

October 1991

The October release (Editor Internal Reference Number 1.10) made the following changes:

Changed x25ConInfoStatus to clarify the description and

the pvcResetting(5) value was changed to pvcResetting(6) to avoid a conflict with a previous use of the number 5.

The name of the counter object x25TmrStatRetryCountsExceeded was changed to x25TmrStatRetryCountExceededs.

The name of the counter object x25TmrStatClearCountsExceeded was changed to x25TmrStatClearCountExceededs.

All occurrence of Guage was changed to Gauge.

Added the x25CallFcltyTable, x25CallFcltyCcittTable, and x25CallParamTable.

June 1991

The June release corrected some syntax errors and cleaned up some other minor things.

April 1991

The April 26 release of this document was the first release. That version was derived from the ISO work on network layer management as presented in ISO/IEC 10733 [11]

6. 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, Snmp Research
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, Clearpoint
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

In addition, the contributions of the following individuals are also acknowledged:

John Harper, DEC
Chairman of the ISO committee for
Network Level Management Information

7. 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., Fedor, M., Schoffstall, M., 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, "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: MIB-II", 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., Editor, "SNMP MIB extension for LAPB", RFC 1381, Data General Corporation, November 1992.
- [10] "Information technology - - Data communication - X.25 Packet layer Protocol for Data Terminal Equipment", International Organization for Standardization, International Standard 8208, March 1990.
- [11] "Information Technology - Telecommunications and information exchange between systems - Elements of Management Information Related to OSI network Layer Standards", Committee Draft International Standard 10733, November 1990.

8. Security Considerations

Security issues are not discussed in this memo.

9. Authors' Addresses

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

Phone: (919)248-8421
EMail: throop@dg-rtp.dg.com