Network Working Group
Request for Comments: 1316

B. Stewart, Editor
Xyplex, Inc.

April 1992

# Definitions of Managed Objects for Character Stream Devices

## 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.

## 1. 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 the management of character stream devices.

## 2. The Network Management Framework

The Internet-standard Network Management Framework consists of three components. They are:

RFC 1155 which defines the SMI, the mechanisms used for describing and naming objects for the purpose of management. RFC 1212 defines a more concise description mechanism, which is wholly consistent with the SMI.

RFC 1156 which defines MIB-I, the core set of managed objects for the Internet suite of protocols. RFC 1213, defines MIB-II, an evolution of MIB-I based on implementation experience and new operational requirements.

RFC 1157 which defines the SNMP, the protocol used for network access to managed objects.

The Framework permits new objects to be defined for the purpose of experimentation and evaluation.

## 3. Objects

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. Objects in the MIB are

defined using the subset of Abstract Syntax Notation One (ASN.1) [7] defined in the SMI. In particular, each object has a name, a syntax, and an encoding. The name is an object identifier, an administratively assigned name, which specifies an object type.

The object type together with an object instance serves to uniquely identify a specific instantiation of the object. For human convenience, we often use a textual string, termed the OBJECT DESCRIPTOR, to also refer to the object type.

The syntax of an object type defines the abstract data structure corresponding to that object type. The ASN.1 language is used for this purpose. However, the SMI [3] purposely restricts the ASN.1 constructs which may be used. These restrictions are explicitly made for simplicity.

The encoding of an object type is simply how that object type is represented using the object type's syntax. Implicitly tied to the notion of an object type's syntax and encoding is how the object type is represented when being transmitted on the network.

The SMI specifies the use of the basic encoding rules of ASN.1 [8], subject to the additional requirements imposed by the SNMP.

### 3.1. Format of Definitions

Section 5 contains the specification of all object types contained in this MIB module. The object types are defined using the conventions defined in the SMI, as amended by the extensions specified in [9,10].

## 4. Overview

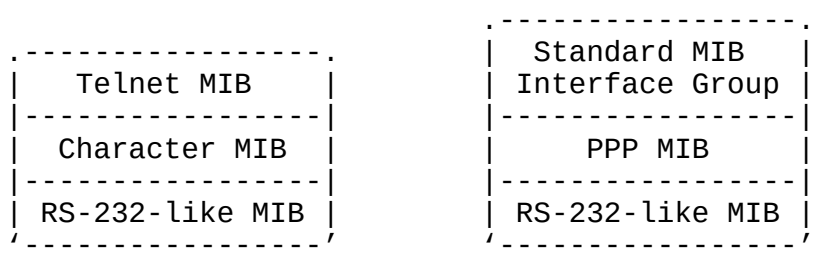
The Character MIB applies to interface ports that carry a character stream, whether physical or virtual, serial or parallel, synchronous or asynchronous. The most common example of a character port is a hardware terminal port with an RS-232 interface. Another common hardware example is a parallel printer port, say with a Centronics interface. The concept also includes virtual terminal ports, such as a software connection point for a remote console.

The Character MIB is one of a set of MIBs designed for complementary use. At this writing, the set comprises:

Character MIB
PPP MIB
RS-232-like MIB
Parallel-printer-like MIB

The RS-232-like MIB and the Parallel-printer-like MIB represent the physical layer, providing service to higher layers such as the Character MIB or PPP MIB. Further MIBs may appear above these.

The following diagram shows two possible "MIB stacks", each using the RS-232-like MIB.




The intent of the model is for the physical-level MIBs to represent the lowest level, regardless of the higher level that may be using it. In turn, separate higher level MIBs represent specific applications, such as a terminal (the Character MIB) or a network connection (the PPP MIB).

For the most part, character ports are distinct from network interfaces (which are already covered by the Interface group). In general, they are attachment points for non-network devices. The exception is a character port that can support a network protocol, such as SLIP or PPP. This implies the existence of a corresponding entry in the Interfaces table, with ifOperStatus of 'off' while the port is not running a network protocol and 'on' if it is. The intent is that such usage is exclusive of non-network character stream usage. That is, while switched to network use, charPortOperStatus would be 'down' and Character MIB operational values such as charPortInFlowState and charPortInCharacters would be inactive.

The Character MIB is mandatory for all systems that offer character ports. This includes, for example, terminal servers, general-purpose time-sharing hosts, and even such systems as a bridge with a (virtual) console port. It may or may not include character ports that do not support network sessions, depending on the system's needs.

The Character MIB's central abstraction is a port. Physical ports have a one-to-one correspondence with hardware ports. Virtual ports are software entities analogous to physical ports, but with no hardware connector.

Each port supports one or more sessions. A session represents a virtual connection that carries characters between the port and some

partner. Sessions typically operate over a stack of network protocols. A typical session, for example, uses Telnet over TCP.

The MIB comprises one base object and two tables, detailed in the following sections. The tables contain objects for ports and sessions.

The MIB intentionally contains no distinction between what is often called permanent and operational or volatile data bases. For the purposes of this MIB, handling of such distinctions is implementation specific.

## 5. Definitions

```
                RFC1316-MIB DEFINITIONS ::= BEGIN

                IMPORTS
                        Counter, TimeTicks, Gauge
                                FROM RFC1155-SMI
                        DisplayString
                                FROM RFC1213-MIB
                        OBJECT-TYPE
                                FROM RFC-1212;

        -- this is the MIB module for character stream devices

        char    OBJECT IDENTIFIER ::= { mib-2 19 }

        -- Textual Conventions

            AutonomousType    ::= OBJECT IDENTIFIER

        -- The object identifier is an independently extensible type
        -- identification value.  It may, for example indicate a
        -- particular sub-tree with further MIB definitions, or
        -- define something like a protocol type or type of
        -- hardware.

            InstancePointer   ::= OBJECT IDENTIFIER

        -- The object identifier is a pointer to a specific instance
        -- of a MIB object in this agent's implemented MIB.  By
        -- convention, it is the first object in the conceptual row
        -- for the instance.
```

```
-- the generic Character group

-- Implementation of this group is mandatory for all
-- systems that offer character ports

charNumber OBJECT-TYPE
    SYNTAX INTEGER
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of entries in charPortTable, regardless
        of their current state."
    ::= { char 1 }


-- the Character Port table

charPortTable OBJECT-TYPE
    SYNTAX SEQUENCE OF CharPortEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "A list of port entries.  The number of entries is
        given by the value of charNumber."
    ::= { char 2 }

charPortEntry OBJECT-TYPE
    SYNTAX CharPortEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "Status and parameter values for a character port."
    INDEX { charPortIndex }
    ::= { charPortTable 1 }

CharPortEntry ::=
    SEQUENCE {
        charPortIndex
            INTEGER,
        charPortName
            DisplayString,
        charPortType
            INTEGER,
        charPortHardware
            AutonomousType,
        charPortReset
            INTEGER,
        charPortAdminStatus
```

```
                    INTEGER,
                charPortOperStatus
                    INTEGER,
                charPortLastChange
                    TimeTicks,
                charPortInFlowType
                    INTEGER,
                charPortOutFlowType
                    INTEGER,
                charPortInFlowState
                    INTEGER,
                charPortOutFlowState
                    INTEGER,
                charPortInCharacters
                    Counter,
                charPortOutCharacters
                    Counter,
                charPortAdminOrigin
                    INTEGER,
                charPortSessionMaximum
                    INTEGER,
                charPortSessionNumber
                    Gauge,
                charPortSessionIndex
                    INTEGER
            }

        charPortIndex OBJECT-TYPE
            SYNTAX INTEGER
            ACCESS read-only
            STATUS mandatory
            DESCRIPTION
                "A unique value for each character port.  Its value
                ranges between 1 and the value of charNumber.  By
                convention and if possible, hardware port numbers
                come first, with a simple, direct mapping.  The
                value for each port must remain constant at least
                from one re-initialization of the network management
                agent to the next."
            ::= { charPortEntry 1 }

        charPortName OBJECT-TYPE
            SYNTAX DisplayString (SIZE (0..32))
            ACCESS read-write
            STATUS mandatory
            DESCRIPTION
                "An administratively assigned name for the port,
                typically with some local significance."
```

```
        ::= { charPortEntry 2 }

    charPortType OBJECT-TYPE
        SYNTAX INTEGER { physical(1), virtual(2) }
        ACCESS read-only
        STATUS mandatory
        DESCRIPTION
            "The port's type, 'physical' if the port represents
            an external hardware connector, 'virtual' if it does
            not."
        ::= { charPortEntry 3 }

    charPortHardware OBJECT-TYPE
        SYNTAX AutonomousType
        ACCESS read-only
        STATUS mandatory
        DESCRIPTION
            "A reference to hardware MIB definitions specific to
            a physical port's external connector.  For example,
            if the connector is RS-232, then the value of this
            object refers to a MIB sub-tree defining objects
            specific to RS-232.  If an agent is not configured
            to have such values, the agent returns the object
            identifier:

                nullHardware OBJECT IDENTIFIER ::= { 0 0 }
            "
        ::= { charPortEntry 4 }

    charPortReset OBJECT-TYPE
        SYNTAX INTEGER { ready(1), execute(2) }
        ACCESS read-write
        STATUS mandatory
        DESCRIPTION
            "A control to force the port into a clean, initial
            state, both hardware and software, disconnecting all
            the port's existing sessions.  In response to a
            get-request or get-next-request, the agent always
            returns 'ready' as the value.  Setting the value to
            'execute' causes a reset."
        ::= { charPortEntry 5 }

    charPortAdminStatus OBJECT-TYPE
        SYNTAX INTEGER { enabled(1), disabled(2), off(3),
                         maintenance(4) }
        ACCESS read-write
        STATUS mandatory
        DESCRIPTION
```

```
                "The port's desired state, independent of flow
                control.  'enabled' indicates that the port is
                allowed to pass characters and form new sessions.
                'disabled' indicates that the port is allowed to
                pass characters but not form new sessions.  'off'
                indicates that the port is not allowed to pass
                characters or have any sessions. 'maintenance'
                indicates a maintenance mode, exclusive of normal
                operation, such as running a test."
            ::= { charPortEntry 6 }

        charPortOperStatus OBJECT-TYPE
            SYNTAX INTEGER { up(1), down(2),
                             maintenance(3), absent(4), active(5) }
            ACCESS read-only
            STATUS mandatory
            DESCRIPTION
                "The port's actual, operational state, independent
                of flow control.  'up' indicates able to function
                normally.  'down' indicates inability to function
                for administrative or operational reasons.
                'maintenance' indicates a maintenance mode,
                exclusive of normal operation, such as running a
                test.  'absent' indicates that port hardware is not
                present.  'active' indicates up with a user present
                (e.g. logged in)."
            ::= { charPortEntry 7 }

        charPortLastChange OBJECT-TYPE
            SYNTAX TimeTicks
            ACCESS read-only
            STATUS mandatory
            DESCRIPTION
                "The value of sysUpTime at the time the port entered
                its current operational state.  If the current state
                was entered prior to the last reinitialization of
                the local network management subsystem, then this
                object contains a zero value."
            ::= { charPortEntry 8 }

        charPortInFlowType OBJECT-TYPE
            SYNTAX INTEGER { none(1), xonXoff(2), hardware(3),
                             ctsRts(4), dsrDtr(5) }
            ACCESS read-write
            STATUS mandatory
            DESCRIPTION
                "The port's type of input flow control.  'none'
                indicates no flow control at this level or below.
```

```
                'xonXoff' indicates software flow control by
                recognizing XON and XOFF characters.  'hardware'
                indicates flow control delegated to the lower level,
                for example a parallel port.

                'ctsRts' and 'dsrDtr' are specific to RS-232-like
                ports.  Although not architecturally pure, they are
                included here for simplicity's sake."
            ::= { charPortEntry 9 }

        charPortOutFlowType OBJECT-TYPE
            SYNTAX INTEGER { none(1), xonXoff(2), hardware(3),
                             ctsRts(4), dsrDtr(5) }
            ACCESS read-write
            STATUS mandatory
            DESCRIPTION
                "The port's type of output flow control.  'none'
                indicates no flow control at this level or below.
                'xonXoff' indicates software flow control by
                recognizing XON and XOFF characters.  'hardware'
                indicates flow control delegated to the lower level,
                for example a parallel port.

                'ctsRts' and 'dsrDtr' are specific to RS-232-like
                ports.  Although not architecturally pure, they are
                included here for simplicy's sake."
            ::= { charPortEntry 10 }

        charPortInFlowState OBJECT-TYPE
            SYNTAX INTEGER { none(1), unknown(2), stop(3), go(4) }
            ACCESS read-only
            STATUS mandatory
            DESCRIPTION
                "The current operational state of input flow control
                on the port.  'none' indicates not applicable.
                'unknown' indicates this level does not know.
                'stop' indicates flow not allowed.  'go' indicates
                flow allowed."
            ::= { charPortEntry 11 }

        charPortOutFlowState OBJECT-TYPE
            SYNTAX INTEGER { none(1), unknown(2), stop(3), go(4) }
            ACCESS read-only
            STATUS mandatory
            DESCRIPTION
                "The current operational state of output flow
                control on the port.  'none' indicates not
                applicable.  'unknown' indicates this level does not
```

```
                know.  'stop' indicates flow not allowed.  'go'
                indicates flow allowed."
            ::= { charPortEntry 12 }

        charPortInCharacters OBJECT-TYPE
            SYNTAX Counter
            ACCESS read-only
            STATUS mandatory
            DESCRIPTION
                "Total number of characters detected as input from
                the port since system re-initialization and while
                the port operational state was 'up', 'active', or
                'maintenance', including, for example, framing, flow
                control (i.e. XON and XOFF), each occurrence of a
                BREAK condition, locally-processed input, and input
                sent to all sessions."
            ::= { charPortEntry 13 }

        charPortOutCharacters OBJECT-TYPE
            SYNTAX Counter
            ACCESS read-only
            STATUS mandatory
            DESCRIPTION
                "Total number of characters detected as output to
                the port since system re-initialization and while
                the port operational state was 'up', 'active', or
                'maintenance', including, for example, framing, flow
                control (i.e. XON and XOFF), each occurrence of a
                BREAK condition, locally-created output, and output
                received from all sessions."
            ::= { charPortEntry 14 }

        charPortAdminOrigin OBJECT-TYPE
            SYNTAX INTEGER { dynamic(1), network(2), local(3),
                             none(4) }
            ACCESS read-write
            STATUS mandatory
            DESCRIPTION
                "The administratively allowed origin for
                establishing session on the port.  'dynamic' allows
                'network' or 'local' session establishment. 'none'
                disallows session establishment."
            ::= { charPortEntry 15 }

        charPortSessionMaximum OBJECT-TYPE
            SYNTAX INTEGER
            ACCESS read-write
            STATUS mandatory
```

```
            DESCRIPTION
                "The maximum number of concurrent sessions allowed
                on the port.  A value of -1 indicates no maximum.
                Setting the maximum to less than the current number
                of sessions has unspecified results."
            ::= { charPortEntry 16 }

        charPortSessionNumber OBJECT-TYPE
            SYNTAX Gauge
            ACCESS read-only
            STATUS mandatory
            DESCRIPTION
                "The number of open sessions on the port that are in
                the connecting, connected, or disconnecting state."
            ::= { charPortEntry 17 }

        charPortSessionIndex OBJECT-TYPE
            SYNTAX INTEGER
            ACCESS read-only
            STATUS mandatory
            DESCRIPTION
                "The value of charSessIndex for the port's first or
                only active session.  If the port has no active
                session, the agent returns the value zero."
            ::= { charPortEntry 18 }


        -- the Character Session table

        charSessTable OBJECT-TYPE
            SYNTAX SEQUENCE OF CharSessEntry
            ACCESS not-accessible
            STATUS mandatory
            DESCRIPTION
                "A list of port session entries."
            ::= { char 3 }

        charSessEntry OBJECT-TYPE
            SYNTAX CharSessEntry
            ACCESS not-accessible
            STATUS mandatory
            DESCRIPTION
                "Status and parameter values for a character port
                session."
            INDEX { charSessPortIndex, charSessIndex }
            ::= { charSessTable 1 }
```

```
        CharSessEntry ::=
            SEQUENCE {
                charSessPortIndex
                    INTEGER,
                charSessIndex
                    INTEGER,
                charSessKill
                    INTEGER,
                charSessState
                    INTEGER,
                charSessProtocol
                    AutonomousType,
                charSessOperOrigin
                    INTEGER,
                charSessInCharacters
                    Counter,
                charSessOutCharacters
                    Counter,
                charSessConnectionId
                    InstancePointer,
                charSessStartTime
                    TimeTicks
            }

        charSessPortIndex OBJECT-TYPE
            SYNTAX INTEGER
            ACCESS read-only
            STATUS mandatory
            DESCRIPTION
                "The value of charPortIndex for the port to which
                this session belongs."
            ::= { charSessEntry 1 }

        charSessIndex OBJECT-TYPE
            SYNTAX INTEGER
            ACCESS read-only
            STATUS mandatory
            DESCRIPTION
                "The session index in the context of the port, a
                non-zero positive integer.  Session indexes within a
                port need not be sequential.  Session indexes may be
                reused for different ports.  For example, port 1 and
                port 3 may both have a session 2 at the same time.
                Session indexes may have any valid integer value,
                with any meaning convenient to the agent
                implementation."
            ::= { charSessEntry 2 }
```

```
        charSessKill OBJECT-TYPE
            SYNTAX INTEGER { ready(1), execute(2) }
            ACCESS read-write
            STATUS mandatory
            DESCRIPTION
                "A control to terminate the session.  In response to
                a get-request or get-next-request, the agent always
                returns 'ready' as the value.  Setting the value to
                'execute' causes termination."
            ::= { charSessEntry 3 }

        charSessState OBJECT-TYPE
            SYNTAX INTEGER { connecting(1), connected(2),
                             disconnecting(3) }
            ACCESS read-only
            STATUS mandatory
            DESCRIPTION
                "The current operational state of the session,
                disregarding flow control.  'connected' indicates
                that character data could flow on the network side
                of session.  'connecting' indicates moving from
                nonexistent toward 'connected'.  'disconnecting'
                indicates moving from 'connected' or 'connecting' to
                nonexistent."
            ::= { charSessEntry 4 }

        charSessProtocol OBJECT-TYPE
            SYNTAX AutonomousType
            ACCESS read-only
            STATUS mandatory
            DESCRIPTION
                "The network protocol over which the session is
                running.  Other OBJECT IDENTIFIER values may be
                defined elsewhere, in association with specific
                protocols.  However, this document assigns those of
                known interest as of this writing."
            ::= { charSessEntry 5 }

        wellKnownProtocols OBJECT IDENTIFIER ::= { char 4 }

        protocolOther  OBJECT IDENTIFIER ::= {wellKnownProtocols 1}
        protocolTelnet OBJECT IDENTIFIER ::= {wellKnownProtocols 2}
        protocolRlogin OBJECT IDENTIFIER ::= {wellKnownProtocols 3}
        protocolLat    OBJECT IDENTIFIER ::= {wellKnownProtocols 4}
        protocolX29    OBJECT IDENTIFIER ::= {wellKnownProtocols 5}
        protocolVtp    OBJECT IDENTIFIER ::= {wellKnownProtocols 6}
```

```
            charSessOperOrigin OBJECT-TYPE
                SYNTAX INTEGER { unknown(1), network(2), local(3) }
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                    "The session's source of establishment."
                ::= { charSessEntry 6 }

            charSessInCharacters OBJECT-TYPE
                SYNTAX Counter
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                    "This session's subset of charPortInCharacters."
                ::= { charSessEntry 7 }

            charSessOutCharacters OBJECT-TYPE
                SYNTAX Counter
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                    "This session's subset of charPortOutCharacters."
                ::= { charSessEntry 8 }

            charSessConnectionId OBJECT-TYPE
                SYNTAX InstancePointer
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                    "A reference to additional local MIB information.
                    This should be the highest available related MIB,
                    corresponding to charSessProtocol, such as Telnet.
                    For example, the value for a TCP connection (in the
                    absence of a Telnet MIB) is the object identifier of
                    tcpConnState.  If an agent is not configured to have
                    such values, the agent returns the object
                    identifier:

                        nullConnectionId OBJECT IDENTIFIER ::= { 0 0 }
                    "
                ::= { charSessEntry 9 }

            charSessStartTime OBJECT-TYPE
                SYNTAX TimeTicks
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                    "The value of sysUpTime in MIB-2 when the session
```

```
                    entered connecting state."
                ::= { charSessEntry 10 }

            END
```

## 6. Acknowledgements

Based on several private MIBs, this document was produced by the Character MIB Working Group:

Anne Ambler, Spider
Charles Bazaar, Emulex
Christopher Bucci, Datability
Anthony Chung, Hughes LAN Systems
George Conant, Xyplex
John Cook, Chipcom
James Davin, MIT-LCS
Shawn Gallagher, DEC
Tom Grant, Xylogics
Frank Huang, Emulex
David Jordan, Emulex
Satish Joshi, SynOptics
Frank Kastenholz, Clearpoint
Ken Key, University of Tennessee
Jim Kinder, Fibercom
Rajeev Kochhar, 3Com
John LoVerso, Xylogics
Keith McCloghrie, Hughes LAN Systems
Donald Merritt, BRL
David Perkins, 3Com
Jim Reinstedler, Ungerman-Bass
Marshall Rose, PSI
Ron Strich, SSDS
Dean Throop, DG
Bill Townsend, Xylogics
Jesse Walker, DEC
David Waitzman, BBN
Bill Westfield, cisco

## 7. References

[1] Cerf, V., "IAB Recommendations for the Development of Internet Network Management Standards", RFC 1052, NRI, April 1988.

[2] Cerf, V., "Report of the Second Ad Hoc Network Management Review Group", RFC 1109, NRI, August 1989.

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

[4] 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.

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

[6] McCloghrie K., and M. Rose, Editors, "Management Information Base for Network Management of TCP/IP-based internets", RFC 1213, Performance Systems International, March 1991.

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

[8] 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.

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

[10] Rose, M., Editor, "A Convention for Defining Traps for use with the SNMP", RFC 1215, Performance Systems International, March 1991.

## 8. Security Considerations

Security issues are not discussed in this memo.

## 9. Author's Address

Bob Stewart
Xyplex, Inc.
330 Codman Hill Road
Boxborough, MA 01719

Phone: (508) 264-9900
EMail: rlstewart@eng.xyplex.com