

Network Working Group
Request for Comments: 1666
Obsoletes: 1665
Category: Standards Track

Z. Kielczewski
Eicon Technology Corporation
D. Kostick
Bell Communications Research
K. Shih
Novell
Editors
August 1994

Definitions of Managed Objects for SNA NAUs using SMIV2

Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Table of Contents

| | |
|--|----|
| 1. Introduction | 2 |
| 2. The SNMPv2 Network Management Framework | 2 |
| 2.1 Object Definitions | 2 |
| 3. Overview | 3 |
| 3.1 Applying MIB II to managing SNA NAUs | 4 |
| 3.2 SNANAU MIB Structure | 4 |
| 3.2.1 snaNode group | 5 |
| 3.2.2 snaLu group | 6 |
| 3.2.3 snaMgtTools group | 7 |
| 3.2.4 Conformance statement | 7 |
| 3.3 SNANAU MIB special feature | 7 |
| 3.3.1 Row Creation mechanism | 8 |
| 3.3.2 State Diagrams | 8 |
| 4. Object Definitions | 9 |
| 5. Acknowledgments | 67 |
| 6. References | 67 |
| 7. Security Considerations | 68 |
| 8. Authors' Addresses | 68 |

1. Introduction

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it defines objects for managing the configuration, monitoring and control of Physical Units (PUs) and Logical Units (LUs) in an SNA environment. PUs and LUs are two types of Network Addressable Units (NAUs) in the logical structure of an SNA network. NAUs are the origination or destination points for SNA data streams. This memo identifies managed objects for PU Type 1.0, 2.0 and Type 2.1 and LU Type 0, 1, 2, 3, 4, 7. The generic objects defined here can also be used to manage LU 6.2 and any LU-LU session. The SNA terms and overall architecture are documented in [1].

2. The SNMPv2 Network Management Framework

The SNMPv2 Network Management Framework consists of four major components. They are:

- o RFC 1442 [2] which defines the SMI, the mechanisms used for describing and naming objects for the purpose of management.
- o STD 17, RFC 1213 [3] defines MIB-II, the core set of managed objects for the Internet suite of protocols.
- o RFC 1445 [4] which defines the administrative and other architectural aspects of the framework.
- o RFC 1448 [5] which defines the protocol used for network access to managed objects.

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

2.1. Object Definitions

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) defined in the SMI (RFC 1442 [2]). In particular, each object type is named by an OBJECT IDENTIFIER, an administratively assigned name. 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 descriptor, to refer to the object type.

3. Overview

This document identifies the proposed set of objects for managing the configuration, monitoring and control of Physical Units (PUs) and Logical Units (LUs) in an SNA environment. In this document, the name "Node" is used to describe SNA Node Type 1.0, 2.0 and Type 2.1 and the name "LU" is used to describe Logical Unit of Type 0, 1, 2, 3, 4, 7 and 6.2. Note however that only objects common to all PU and LU types are covered here and LU 6.2 specific objects are not included in this MIB module.

Highlights of the management functions supported by the SNANAU MIB module include the following:

- o Creation/deletion of Nodes and LUs via the RowStatus objects in the snaNodeAdminTable and in the snaLuAdminTable.
- o Creation/deletion of table entries associating Node instances with link instances via the RowStatus object in the snaNodeLinkAdminTable
- o Activation/Deactivation of Nodes via the AdminState object in the snaNodeAdminTable
- o Deactivation of sessions via the AdminState object in the snaLuSessnTable
- o Monitoring and modification of parameters related to Nodes, LUs, and Node/link associations
- o Monitoring of session operational parameters
- o PU2.0 operational statistics
- o Session operational statistics
- o RTM statistics
- o Traps for:
 - + Node state change
 - + Node activation failure
 - + LU state change
 - + LU session BIND failure

This MIB module does not support:

- o creation of links,
- o activation or deactivation of LUs, nor
- o activation of sessions.

3.1. Applying MIB II to managing SNA NAUs

This section identifies how MIB II objects, specifically the MIB II system group will be used in SNMP-based management of SNA NAUs. The MIB II system group applies to the SNMP Agent. The following object is from the MIB II system group:

sysUpTime: clock in the SNMP Agent/proxy-Agent; expressed in TimeTicks (1/100s of a seconds).

This MIB module uses the TimeStamp TEXTUAL-CONVENTION which is defined in the SNMPv2 Textual Conventions (RFC 1443 [6]) as "the value of MIB II's sysUpTime object when a specific occurrence happens." The specific occurrences related to SNA NAU management are defined in this MIB module.

3.2. SNANAU MIB Structure

The SNANAU MIB module contains three groups of objects:

- o snaNode - objects related to Node configuration, monitoring and control.
- o snaLu - objects related to LU definition, monitoring and control.
- o snaMgtTools - objects related to specific management tools well known in SNA environment.

These groups are described below in more detail.

The objects related to PUs and LUs are organized into two types of tables: the Admin and Oper tables.

The "Admin" table contains parameters which are used by a Management Station to affect the operation of the SNA service. Some parameters are used to initialize and configure the SNA service at the next startup, while others can take effect immediately. A Management Station can dynamically define SNA resources (PUs, LUs) by creating new entries in the Admin table. It uses a special object, AdminState,

to control the desired state of a defined PU or LU Session resource. Note that this MIB does not allow the manipulation of an LU's operational state.

The "Oper" table is an extension (augment) of the corresponding Admin table. It contains objects which correspond to the values of parameters currently used by the SNA system.

3.2.1. snaNode group

The snaNode group consists of the following tables:

1) snaNodeAdminTable - This table contains objects which describe the configuration parameters of an SNA Node. Link-specific configuration objects are contained in a separate MIB module (e.g., the SNA DLC MIB module) corresponding to link type. Entries in this table can be created, modified and deleted by either an Agent or a Management Station. The snaNodeAdminRowStatus object describes the status of an entry and is used to change the status of that entry.

The snaNodeAdminState object describes the desired operational state of a Node and is used to change the operational state of a Node.

How an Agent or a Management Station obtains the initial value of each object at creation time is an implementation specific issue not addressed in this memo.

For each entry in the snaNodeAdminTable, there is a corresponding entry in the snaNodeOperTable. While the objects in this table describe the desired or configured operational values of the SNA Node, the actual runtime values are contained in snaNodeOperTable.

2) snaNodeOperTable - Each row contains runtime and operational state variables for a Node. It is an extension of snaNodeAdminTable and as such uses the same index. The rows in this table are created by an Agent as soon as the entry in the Admin Table become 'active'. The entries in this table cannot be modified by a Management Station.

3) snaPu20StatsTable - Each row contains statistics variables (counters) for a PU 2.0. The entries in this table are indexed by snaNodeAdminIndex. The rows in this table are created by an Agent as soon as the corresponding entry in the snaNodeAdminTable becomes 'active'.

4) `snaNodeLinkAdminTable` - This table contains all references to link-specific tables. If a Node is configured with multiple links, then it will have multiple entries in this table. The entries in this table can be generated initially, after startup of SNA service, by the Agent which uses information from Node configuration file. Subsequent modifications of parameters, creation of new Node link entries and deletion of entries is possible. The modifications to this table can be saved in the Node configuration file for the next startup (i.e., restart or next initialization) of SNA service, but the mechanism for this function is not defined in this memo. Each entry contains the configuration information that associates a Node instance to one link instance. The entries are indexed by `snaNodeAdminIndex` and `snaNodeLinkAdminIndex`.

5) `snaNodeLinkOperTable` - This table contains all references to link-specific tables for operational parameters. If the Node is configured for multiple links, then it will have multiple entries in this table. This table augments the `snaNodeLinkAdminTable`.

6) `snaNodeTraps` - Two traps are defined for Nodes. The `snaNodeStateChangeTrap` indicates that the operational state of a Node has changed. The `snaNodeActFailTrap` indicates the failure of ACTPU received from host.

3.2.2. `snaLu` group

The `snaLu` group consists of the following tables:

1) `snaLuAdminTable` - Table containing LU configuration information. The rows in this table can be created and deleted by a Management Station. Only objects which are common to all types of LUs are included in this table. The entries are indexed by Node and LU indices.

2) `snaLuOperTable` - Table containing dynamic runtime information and control variables relating to LUs. Only objects which are common to all types of LUs are included in this table. This table augments the `snaLuAdminTable`.

3) `snaLuSessnTable` - This is a table containing objects which describe the operational state of LU-LU sessions. Only objects which are common to all types of LU-LU sessions are included in this table. When a session's `snaLuSessnOperState` value changes to entry in the session table is created by the Agent. When the `snaLuSessionOperState` value changes to will be removed from the session table by the Agent. Entries are indexed by Node, local LU, remote LU and session indices.

4) `snaLuSessnStatsTable` - Table containing dynamic statistics information relating to LU-LU sessions. The entries in this table augment the entries in the `snaLuSessnTable` and cannot be created by a Management Station.

5) `snaLuTraps` - Two traps are defined for LUs. The `snaLuStateChangeTrap` indicates that the operational state of an LU has changed. The `snaLuSessnBindFailTrap` indicates the failure of a BIND request.

3.2.3. `snaMgtTools` group

This is an optional group. The `snaMgtTools` group consists of the following table:

1) `snaLuRtmTable` - Each row contains Response Time Monitor (RTM) variables for an LU. The table is indexed by Node and LU indices. Entries correspond to LU 2 entries in the `snaLuAdminTable`. A Management Station can read collection of RTM statistics for a given LU.

3.2.4. Conformance statement

Compliance of the SNMPv2 management entity to the SNANAU MIB is defined in terms of following conformance units called groups.

Unconditionally mandatory groups: `snaNodeGroup`, `snaLuGroup`, `snaSessionGroup`.

Conditionally mandatory groups: `snaPu20Group` - mandatory only for those entities which implement PU type 2.0. The `snaMgtToolsRtmGroup` - mandatory only for those entities which implement LU type 2 and RTM.

Refinement of requirements for objects access: an Agent which does not implement row creation for `snaNodeAdminTable`, `snaNodeLinkAdminTable` and `snaLuAdminTable` must at least support object modification requests (i.e., read-write access instead of read-create).

3.3. SNANAU MIB special feature

This section describes the mechanism used for row creation in the Admin tables and also presents critical state transitions for PUs, LUs and Sessions.

3.3.1. Row Creation mechanism

The row creation mechanism for the Admin tables in this MIB module is based on the use of the RowStatus object. Restriction of some operations for specific tables are described in each table. In particular, before accepting the 'destroy' value for an entry, an Agent has to verify the operational state of the corresponding entry in the Oper table.

3.3.2. State Diagrams

The following state diagram models the state transitions for Nodes. When a row is created by a Management Station, an Agent creates the Oper table entry for that Node with the OperState equal to 'inactive'. An Agent cannot accept any operations for that Node until the RowStatus is set to 'active'.

| OperState -> | inactive | active | waiting | stopping |
|--------------|-------------|---------------|-------------|-------------|
| -----I----- | -----I----- | -----I----- | -----I----- | -----I----- |
| AdminState: | I | I | I | I |
| active | I active | I active | I waiting | I no |
| | I | I | I | I |
| inactive | I inactive | I stopping | I inactive | I stopping |
| | | I or inactive | I | |

The following state diagram models state transitions for Sessions. When a session goes to the 'unbound' state [1], the corresponding entry will be removed from the Session table by the Agent.

| OperState -> | unbound | pendingBind | bound | pendingUnbind |
|--------------|-------------|-------------|-------------|---------------|
| -----I----- | -----I----- | -----I----- | -----I----- | -----I----- |
| AdminState: | I | I | I | I |
| bound | I no | I no | I no | I no |
| | I | I | I | I |
| unbound | I unbound | I unbound | I unbound | I unbound |

4. Object Definitions

SNA-NAU-MIB DEFINITIONS ::= BEGIN

-- This MIB module contains objects necessary
-- for management of the following SNA devices: PU types 1.0, 2.0, 2.1
-- and LU types 0, 1, 2, 3, 4, 7. It also contains generic objects
-- which can be used to manage LU 6.2.

-- Naming conventions in this document:
-- The following names are used in object descriptors according to
-- SNA conventions.
-- The name 'PU' or 'Node' is used to describe Node type 1.0, 2.0 or
-- 2.1.
-- The name 'LU' is used to describe Logical Unit of type 0,1,2,3,
-- 4,7 or 6.2.

IMPORTS

DisplayString, RowStatus, TimeStamp, InstancePointer
FROM SNMPv2-TC

Counter32, Gauge32, Integer32,
OBJECT-TYPE, MODULE-IDENTITY, NOTIFICATION-TYPE
FROM SNMPv2-SMI

MODULE-COMPLIANCE, OBJECT-GROUP
FROM SNMPv2-CONF;

snanauMIB MODULE-IDENTITY

LAST-UPDATED "9405120900Z"
ORGANIZATION "IETF SNA NAU MIB Working Group"
CONTACT-INFO

" Zbigniew Kielczewski
Eicon Technology Inc.
2196 32nd Avenue
Lachine, Que H8T 3H7
Canada
Tel: 1 514 631 2592
E-mail: zbig@eicon.qc.ca

Deirdre Kostick
Bellcore
331 Newman Springs Road
Red Bank, NJ 07701
Tel: 1 908 758 2642

E-mail: dck2@mail.bellcore.com

Kitty Shih (editor)
 Novell
 890 Ross Drive
 Sunnyvale, CA 94089
 Tel: 1 408 747 4305
 E-mail: kmshih@novell.com"

DESCRIPTION

"This is the MIB module for objects used to
 manage SNA devices."

```
::= { mib-2 34 }
```

```
-- The SNANAU MIB module contains an objects part and a conformance part.
-- Objects are organized into the following groups:
-- (1)snaNode group,
-- (2)snaLU group,
-- (3)snaMgtTools group.
```

```
snanauObjects OBJECT IDENTIFIER ::= { snanauMIB 1 }
```

```
    snaNode      OBJECT IDENTIFIER ::= { snanauObjects 1 }
    snaLu        OBJECT IDENTIFIER ::= { snanauObjects 2 }
    snaMgtTools  OBJECT IDENTIFIER ::= { snanauObjects 3 }
```

```
-- *****
-- snaNode group
--
-- It contains Managed Objects related to any type of Node and
-- some specific objects for Node Type 2.0.
-- *****
```

```
-- *****
-- The following table contains generic Node configuration
-- parameters.
-- *****
```

snaNodeAdminTable OBJECT-TYPE

SYNTAX SEQUENCE OF SnaNodeAdminEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"This table contains objects which describe the
 configuration parameters for an SNA Node. Link
 specific configuration objects are contained in
 a separate MIB module (e.g., SNA DLC MIB)

corresponding to the link type.
 The table `snaNodeAdminLinkTable` contains objects which identify the relationship between node instances and link instances.

The entries (i.e., rows) in this table can be created by either an Agent or a Management Station. The Management Station can do this through setting the appropriate value in the `snaNodeAdminRowStatus`.

The `snaNodeAdminRowStatus` object describes the status of an entry and is used to change the status of an entry. The entry is deleted by an Agent based on the value of the `snaNodeAdminRowStatus`.

The `snaNodeAdminState` object describes the desired operational state of a Node and is used to change the operational state of a Node. For example, such information may be obtained from a configuration file.

How an Agent or a Management Station obtains the initial value of each object at creation time is an implementation specific issue.

For each entry in this table, there is a corresponding entry in the `snaNodeOperTable`.

While the objects in this table describe the desired or configured operational values of the SNA Node, the actual runtime values are contained in `snaNodeOperTable`."

```
::= { snaNode 1 }
```

`snaNodeAdminEntry` OBJECT-TYPE

```
SYNTAX      SnaNodeAdminEntry
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
```

"An entry contains the configuration parameters for one SNA Node instance. The objects in the entry have read-create access.

An entry can be created, modified or deleted. The object `snaNodeAdminRowStatus` is used (i.e., set) to create or delete a row entry."

```
INDEX { snaNodeAdminIndex }
```

```
::= { snaNodeAdminTable 1 }
```

```
SnaNodeAdminEntry ::= SEQUENCE {
    snaNodeAdminIndex
```

```

        Integer32,
snaNodeAdminName
        DisplayString,
snaNodeAdminType
        INTEGER,
snaNodeAdminXidFormat
        INTEGER,
snaNodeAdminBlockNum
        DisplayString,
snaNodeAdminIdNum
        DisplayString,
snaNodeAdminEnablingMethod
        INTEGER,
snaNodeAdminLuTermDefault
        INTEGER,
snaNodeAdminMaxLu
        Integer32,
snaNodeAdminHostDescription
        DisplayString,
snaNodeAdminStopMethod
        INTEGER,
snaNodeAdminState
        INTEGER,
snaNodeAdminRowStatus
        RowStatus
    }

```

snaNodeAdminIndex OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"Index used to uniquely identify each Node instance.
If an Agent creates the entry, then it will assign
this number otherwise a Management Station
generates a random number when it reserves the
entry for creation."

::= { snaNodeAdminEntry 1 }

snaNodeAdminName OBJECT-TYPE

SYNTAX DisplayString (SIZE(0..17))

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The value indicates the desired name of the
Node for use during Node activation.
In Type 2.1 networks, this is a fully-qualified name,
meaning that the Node name is preceded by the NetId (if

present) with a period as the delimiter.

A write operation to this object will not change the operational value reflected in snaNodeOperName until the Node has been re-activated (e.g., after the next initialization of the SNA services)."

::= { snaNodeAdminEntry 2 }

snaNodeAdminType OBJECT-TYPE

SYNTAX INTEGER {
 other(1),
 pu10(2),
 pu20(3),
 t21len(4),
 endNode(5),
 networkNode(6)
}

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The value indicates the type of SNA Node.

A write operation to this object will not change the operational value reflected in snaNodeOperType until the Node has been re-activated (e.g., after the next initialization of the SNA services)."

::= { snaNodeAdminEntry 3 }

snaNodeAdminXidFormat OBJECT-TYPE

SYNTAX INTEGER {
 format0(1),
 format1(2),
 format3(3)
}

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The value indicates the type of XID format used for this Node. Note that there is no format type 2.

A write operation to this object will not change the operational value reflected in snaNodeOperAdminXidFormat until the Node has been re-activated (e.g., after the next initialization of the SNA services)."

::= { snaNodeAdminEntry 4 }

snaNodeAdminBlockNum OBJECT-TYPE

SYNTAX DisplayString (SIZE(3))

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The value indicates the block number for this Node instance. It is the first 3 hexadecimal digits of the SNA Node id.

A write operation to this object will not change the operational value reflected in snaNodeOperBlockNum until the Node has been re-activated (e.g., after the next initialization of the SNA services)."

::= { snaNodeAdminEntry 5 }

snaNodeAdminIdNum OBJECT-TYPE

SYNTAX DisplayString (SIZE(5))

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The value indicates the ID number for this Node instance. This is the last 5 hexadecimal digits of the SNA Node id.

A write operation to this object will not change the operational value reflected in snaNodeOperIdNum until the Node has been re-activated (e.g., after the next initialization of the SNA services)."

::= { snaNodeAdminEntry 6 }

snaNodeAdminEnablingMethod OBJECT-TYPE

SYNTAX INTEGER {
 other (1),
 startup (2),
 demand (3),
 onlyMS (4)
}

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The value indicates how the Node should be activated for the first time.

The values have the following meanings:

other (1) - may be used for proprietary methods
 not listed in this enumeration,

startup (2) - at SNA services' initialization time
 (this is the default),
 demand (3) - only when LU is requested by application,
 or
 onlyMS (4) - by a Management Station only.

A write operation to this object may immediately change the operational value reflected in `snaNodeOperEnablingMethod` depending on the Agent implementation. If the Agent implementation accepts immediate changes, then the behavior of the Node changes immediately and not only after the next system startup of the SNA services. An immediate change may only apply when the current value 'demand (3)' is changed to 'onlyMS (4)' and vice versa."

::= { `snaNodeAdminEntry` 7 }

`snaNodeAdminLuTermDefault` OBJECT-TYPE

```
SYNTAX  INTEGER {
    unbind (1),
    termself (2),
    rshutd (3),
    poweroff(4)
}
```

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The value indicates the desired default method used to deactivate LUs for this Node
 For LU6.2s, 'unbind(1)' is the only valid value.

`unbind(1)` - terminate the LU-LU session by sending an SNA UNBIND request.

`termself(2)` - terminate the LU-LU session by sending an SNA TERM-SELF (Terminate Self) request on the SSCP-LU session. The SSCP will inform the remote session LU partner to send an UNBIND request to terminate the session.

`rshutd(3)` - terminate the LU-LU session by sending an SNA RSHUTD (Request ShutDown) request to the remote session LU partner. The remote LU will then send an UNBIND request to terminate the session.

`poweroff(4)` - terminate the LU-LU session by sending either an SNA LUSTAT (LU Status) request on the LU-LU session or an SNA NOTIFY request on the SSCP-LU session indicating that the LU has

been powered off. Sending both is also acceptable. The result should be that the remote session LU partner will send an UNBIND to terminate the session.

The default behavior indicated by the value of this object may be overridden for an LU instance. The override is performed by setting the snaLuAdminTerm object instance in the snaLuAdminTable to the desired value.

A write operation to this object may immediately change the operational value reflected in snaNodeOperLuTermDefault depending on the Agent implementation."

::= { snaNodeAdminEntry 8 }

snaNodeAdminMaxLu OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The maximum number of LUs that may be activated for this Node. For PU2.1, this object refers to the number of dependent LUs.

A write operation to this object will not change the operational value reflected in snaNodeOperMaxLu until the Node has been re-activated (e.g., after the next initialization of the SNA services)."

::= { snaNodeAdminEntry 9 }

snaNodeAdminHostDescription OBJECT-TYPE

SYNTAX DisplayString (SIZE(0..128))

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The value identifies the remote host associated with this Node.

Since SSCP Id's may not be unique across hosts, the host description is required to uniquely identify the SSCP. This object is only applicable to PU2.0 type Nodes. If the remote host is unknown, then the value is the null string.

A write operation to this object may immediately

change the operational value reflected
in snaNodeOperHostDescription depending
on the Agent implementation."
 ::= { snaNodeAdminEntry 10 }

snaNodeAdminStopMethod OBJECT-TYPE

SYNTAX INTEGER {
 other (1),
 normal (2),
 immed (3),
 force (4)
}

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The value indicates the desired method to be used
by the Agent to stop a Node (i.e., change the Node's
operational state to inactive(1)).

The values have the following meaning:

other (1) - used for proprietary
 methods not listed in this enumeration.
normal(2) - deactivate only when there is no more
 activity on this Node (i.e., all data flows
 have been completed and all sessions
 have been terminated).
immed(3) - deactivate immediately regardless of
 current activities on this Node. Wait for
 deactivation responses (from remote Node)
 before changing the Node state to inactive.
force(4) - deactivate immediately regardless of
 current activities on this Node. Do not wait
 for deactivation responses (from remote Node)
 before changing the Node state to inactive.

A write operation to this object may immediately
change the operational value reflected
in snaNodeOperStopMethod depending
on the Agent implementation."
 ::= { snaNodeAdminEntry 11 }

snaNodeAdminState OBJECT-TYPE

SYNTAX INTEGER {
 inactive (1),
 active (2)
}

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The value indicates the desired operational state of the SNA Node. This object is used by the Management Station to activate or deactivate the Node.

If the current value in snaNodeOperState is 'active (2)', then setting this object to 'inactive (1)' will initiate the Node shutdown process using the method indicated by snaNodeOperStopMethod.

If the current value in snaNodeOperState is 'inactive (1)', then setting this object to 'active (2)' will initiate the Node's activation.

A Management Station can always set this object to 'active (2)' irrespective of the value in the snaOperEnablingMethod."

::= { snaNodeAdminEntry 12 }

snaNodeAdminRowStatus OBJECT-TYPE

SYNTAX RowStatus

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This object is used by a Management Station to create or delete the row entry in the snaNodeAdminTable following the RowStatus textual convention.

Upon successful creation of the row, an Agent automatically creates a corresponding entry in the snaNodeOperTable with snaNodeOperState equal to 'inactive (1)'.

Row deletion can be Management Station or Agent initiated:

- (a) The Management Station can set the value to 'destroy (6)' only when the value of snaNodeOperState of this Node instance is 'inactive (1)'. The Agent will then delete the rows corresponding to this Node instance from the snaNodeAdminTable and the snaNodeOperTable.
- (b) The Agent detects that a row is in the 'notReady (3)' state for greater than a

default period of 5 minutes.

(c) All rows with the snaNodeAdminRowStatus object's value of 'notReady (3)' will be removed upon the next initialization of the SNA services."

::= { snaNodeAdminEntry 13 }

```
-- *****
-- The following object is updated when there is a change to
-- the value of any object in the snaNodeAdminTable.
-- *****
```

snaNodeAdminTableLastChange OBJECT-TYPE

SYNTAX TimeStamp

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The value indicates the timestamp (e.g., the Agent's sysUpTime value) of the last change made to any object in the snaNodeAdminTable, including row deletions/additions (e.g., changes to snaNodeAdminRowStatus values).

This object can be used to reduce frequent retrievals of the snaNodeAdminTable by a Management Station. It is expected that a Management Station will periodically poll this object and compare its current value with the previous one. A difference indicates that some Node configuration information has been changed. Only then will the Management Station retrieve the entire table."

::= { snaNode 2 }

```
-- *****
-- The following table contains Node operational parameters.
-- *****
```

snaNodeOperTable OBJECT-TYPE

SYNTAX SEQUENCE OF SnaNodeOperEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"This table contains the dynamic parameters which have read-only access. These objects reflect the actual status of the Node. The entries in this table cannot be created or modified by a Management Station.

This table augments the snaNodeAdminTable."
 ::= { snaNode 3 }

snaNodeOperEntry OBJECT-TYPE

 SYNTAX SnaNodeOperEntry

 MAX-ACCESS not-accessible

 STATUS current

 DESCRIPTION

 "The entry contains parameters which describe the
 state of one Node. The entries are created by the
 Agent. They have read-only access."

 AUGMENTS { snaNodeAdminEntry }

 ::= { snaNodeOperTable 1 }

SnaNodeOperEntry ::= SEQUENCE {

 snaNodeOperName

 DisplayString,

 snaNodeOperType

 INTEGER,

 snaNodeOperXidFormat

 INTEGER,

 snaNodeOperBlockNum

 DisplayString,

 snaNodeOperIdNum

 DisplayString,

 snaNodeOperEnablingMethod

 INTEGER,

 snaNodeOperLuTermDefault

 INTEGER,

 snaNodeOperMaxLu

 Integer32,

 snaNodeOperHostDescription

 DisplayString,

 snaNodeOperStopMethod

 INTEGER,

 snaNodeOperState

 INTEGER,

 snaNodeOperHostSscpId

 OCTET STRING,

 snaNodeOperStartTime

 TimeStamp,

 snaNodeOperLastStateChange

 TimeStamp,

 snaNodeOperActFailures

 Counter32,

 snaNodeOperActFailureReason

 INTEGER

}

snaNodeOperName OBJECT-TYPE

SYNTAX DisplayString (SIZE(0..17))

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The value identifies the current name of the Node.

In Type 2.1 networks, this

is a fully-qualified name, meaning that the Node name

is preceded by the NetId (if present) with a period

as the delimiter."

::= { snaNodeOperEntry 1 }

snaNodeOperType OBJECT-TYPE

SYNTAX INTEGER {
 other(1),
 pu10(2),
 pu20(3),
 t21LEN(4),
 endNode(5),
 networkNode(6)
}

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The value identifies the current type of the Node."

::= { snaNodeOperEntry 2 }

snaNodeOperXidFormat OBJECT-TYPE

SYNTAX INTEGER {
 format0 (1),
 format1 (2),
 format3 (3)
}

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The value identifies the type of XID format currently
used for this Node.

Note that there is no format type 2."

::= { snaNodeOperEntry 3 }

snaNodeOperBlockNum OBJECT-TYPE

SYNTAX DisplayString (SIZE(3))

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The value identifies the block number for this Node
instance. It is the first 3 hexadecimal digits

of the SNA Node id."
 ::= { snaNodeOperEntry 4 }

snaNodeOperIdNum OBJECT-TYPE

SYNTAX DisplayString (SIZE(5))

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The value identifies the ID number for this Node instance. This is the last 5 hexadecimal digits of the SNA Node id."

::= { snaNodeOperEntry 5 }

snaNodeOperEnablingMethod OBJECT-TYPE

SYNTAX INTEGER {

other (1),

startup (2),

demand (3),

onlyMS (4)

}

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The value indicates how the Node is activated for the first time.

The values have the following meanings:

other (1) - not at boot time, LU activation or by a Management Station;

startup (2) - at SNA services' initialization time (this is the default),

demand (3) - only when LU is requested by application,

onlyMS (4) - by a network Management Station only."

::= { snaNodeOperEntry 6 }

snaNodeOperLuTermDefault OBJECT-TYPE

SYNTAX INTEGER {

unbind (1),

termself (2),

rshutd (3),

poweroff (4)

}

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The value identifies the default method used to deactivate LUs for this Node.

For LU6.2s, 'unbind(1)' is the only valid value.

```

unbind(1) - terminate the LU-LU session by sending
            an SNA UNBIND request.
termself(2) - terminate the LU-LU session by sending
              an SNA TERM-SELF (Terminate Self) request on
              the SSCP-LU session. The SSCP will inform the
              remote session LU partner to send an UNBIND
              request to terminate the session.
rshutd(3) - terminate the LU-LU session by sending
            an SNA RSHUTD (Request ShutDown) request to
            the remote session LU partner. The remote LU
            will then send an UNBIND request to terminate
            the session.
poweroff(4) - terminate the LU-LU session by sending
              either an SNA LUSTAT (LU Status) request on
              the LU-LU session or an SNA NOTIFY request on
              the SSCP-LU session indicating that the LU has
              been powered off. Sending both is also
              acceptable. The result should be that the
              remote session LU partner will send an UNBIND
              to terminate the session.

```

This object describes the default behavior for this Node; however, it is possible that for a specific LU the behavior indicated by the snaLuOperTerm object is different."

```
::= { snaNodeOperEntry 7 }
```

snaNodeOperMaxLu OBJECT-TYPE

```

SYNTAX  Integer32
MAX-ACCESS  read-only
STATUS  current
DESCRIPTION

```

"This value identifies the current, maximum number of LUs that are activated for this Node. For PU2.1, this object refers to the number of dependent LUs."

```
::= { snaNodeOperEntry 8 }
```

snaNodeOperHostDescription OBJECT-TYPE

```

SYNTAX  DisplayString (SIZE(0..128))
MAX-ACCESS  read-only
STATUS  current
DESCRIPTION

```

"This value identifies the remote host currently associated with this Node. Since SSCP Id's may not be unique across hosts, the host description

is required to uniquely identify the SSCP."
 ::= { snaNodeOperEntry 9 }

snaNodeOperStopMethod OBJECT-TYPE

SYNTAX INTEGER {
 other (1),
 normal (2),
 immed (3),
 force (4)
}

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This value identifies the current Node shutdown method to be used by the Agent to stop the Node. When the Agent changes the Node's state to 'inactive (1)', the Agent must use the shutdown method indicated by this object.

The values have the following meaning:

other (1) - proprietary method not listed in this enumeration

normal(2) - deactivate only when there is no more activity on this Node (i.e., all data flows have been completed and all sessions have been terminated).

immed(3) - deactivate immediately regardless of current activities on this Node. Wait for deactivation responses (from remote Node) before changing the Node state to inactive.

force(4) - deactivate immediately regardless of current activities on this Node. Do not wait for deactivation responses (from remote Node) before changing the Node state to inactive.

Note that a write operation to snaNodeAdminOperStopMethod may immediately change the value of snaNodeOperStopMethod depending on the Agent implementation."

::= { snaNodeOperEntry 10 }

snaNodeOperState OBJECT-TYPE

SYNTAX INTEGER {
 inactive (1),
 active (2),
 waiting (3),
 stopping (4)


```

    }
MAX-ACCESS read-only
STATUS current
DESCRIPTION
    "The current state of the Node.
    The values have the following meanings:
        inactive (1), a row representing the Node has
        been created in the AdminTable
        and, the Node is ready for activation -or-
        an active Node has been stopped -or-
        a waiting Node has returned to the inactive
        state.
        waiting (3), a request to have the Node activated
        has been issued, and the Node is pending
        activation.
        active (2), the Node is ready and operating.
        stopping (4), the request to stop the Node has
        been issued while the StopMethod normal
        or immediate is used."
 ::= { snaNodeOperEntry 11 }

```

```

snaNodeOperHostSscpId OBJECT-TYPE
    SYNTAX OCTET STRING (SIZE(0..6))
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "This value identifies the current SSCP Id
        associated with the Node. This object is only
        applicable to PU 2.0s. If the Node
        is not a PU 2.0 type, then this object contains a
        zero length string."
 ::= { snaNodeOperEntry 12 }

```

```

snaNodeOperStartTime OBJECT-TYPE
    SYNTAX TimeStamp
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The timestamp (e.g, the Agent's sysUpTime value)
        at the Node activation."
 ::= { snaNodeOperEntry 13 }

```

```

snaNodeOperLastStateChange OBJECT-TYPE
    SYNTAX TimeStamp
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The timestamp (e.g., the Agent's sysUpTime value)

```

at the last state change of the Node."
 ::= { snaNodeOperEntry 14 }

snaNodeOperActFailures OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This value identifies the number of failed Node activation attempts."

::= { snaNodeOperEntry 15 }

snaNodeOperActFailureReason OBJECT-TYPE

SYNTAX INTEGER {
 other (1),
 linkFailure (2),
 noResources (3),
 badConfiguration (4),
 internalError (5)
 }

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The value indicates the reason for the activation failure. The value 'other (1)' indicates a reason not listed in the enumeration. This object will be sent in the trap snaNodeActFailTrap."

::= { snaNodeOperEntry 16 }

```
-- *****
-- The following object is updated when there is a change to
-- the value of snaNodeOperState in any row or a row is
-- added/deleted from the snaNodeOperTable via the snaNodeAdminTable.
-- *****
```

snaNodeOperTableLastChange OBJECT-TYPE

SYNTAX TimeStamp

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The timestamp (e.g., the Agent's sysUpTime value) at the last change made to any object in the snaNodeOperTable, including row deletions/additions made as a result of changes to the snaNodeAdminRowStatus object.

This object can be used to reduce frequent

retrievals of the snaNodeOperTable by a Management Station. It is expected that a Management Station will periodically poll this object and compare its current value with the previous one. A difference indicates that some Node operational information has been changed. Only then will the Management Station retrieve the entire table."

::= { snaNode 4 }

```
-- *****
-- The following table contains PU 2.0 statistics dynamic parameters.
-- *****
```

snaPu20StatsTable OBJECT-TYPE

SYNTAX SEQUENCE OF SnaPu20StatsEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"This table contains the dynamic parameters which have read-only access. The entries in this table correspond to PU 2.0 entries in the snaNodeOperTable and cannot be created by a Management Station."

::= { snaNode 5 }

snaPu20StatsEntry OBJECT-TYPE

SYNTAX SnaPu20StatsEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"The entry contains parameters which describe the statistics for one PU 2.0. They have read-only access.

The counters represent traffic for all kinds of sessions: LU-LU, SSCP-PU, SSCP-LU.

Each Node of PU Type 2.0 from the snaNodeAdminTable has one entry in this table and the index used here has the same value as snaNodeAdminIndex of that PU. The entry is created by the Agent."

INDEX { snaNodeAdminIndex }

::= { snaPu20StatsTable 1 }

SnaPu20StatsEntry ::= SEQUENCE {

snaPu20StatsSentBytes

Counter32,

snaPu20StatsReceivedBytes

Counter32,

```
snaPu20StatsSentPius
    Counter32,
snaPu20StatsReceivedPius
    Counter32,
snaPu20StatsSentNegativeResps
    Counter32,
snaPu20StatsReceivedNegativeResps
    Counter32,
snaPu20StatsActLus
    Gauge32,
snaPu20StatsInActLus
    Gauge32,
snaPu20StatsBindLus
    Gauge32
}
```

```
snaPu20StatsSentBytes OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The number of bytes sent by this Node."
    ::= { snaPu20StatsEntry 1 }
```

```
snaPu20StatsReceivedBytes OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The number of bytes received by this Node."
    ::= { snaPu20StatsEntry 2 }
```

```
snaPu20StatsSentPius OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The number of PIUs sent by this Node."
    ::= { snaPu20StatsEntry 3 }
```

```
snaPu20StatsReceivedPius OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The number of PIUs received by this Node."
    ::= { snaPu20StatsEntry 4 }
```

```
snaPu20StatsSentNegativeResps OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The number of negative responses sent
         by this Node."
    ::= { snaPu20StatsEntry 5 }

snaPu20StatsReceivedNegativeResps OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The number of negative responses received
         by this Node."
    ::= { snaPu20StatsEntry 6 }

snaPu20StatsActLus OBJECT-TYPE
    SYNTAX Gauge32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The number of LUs on this PU which have
         received and responded to ACTLU from the host."
    ::= { snaPu20StatsEntry 7 }

snaPu20StatsInActLus OBJECT-TYPE
    SYNTAX Gauge32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The number of LUs on this PU which have
         not received an ACTLU from the host.  This is
         possible if the number of configured LUs exceeds
         that on the host."
    ::= { snaPu20StatsEntry 8 }

snaPu20StatsBindLus OBJECT-TYPE
    SYNTAX Gauge32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The number of LUs on this PU which have
         received and acknowledged a BIND request from the
         host."
    ::= { snaPu20StatsEntry 9 }
```

```
-- *****
-- The following table contains the association between Nodes and
-- link identifiers.
-- It is used for configuration purposes.
-- *****
```

snaNodeLinkAdminTable OBJECT-TYPE

SYNTAX SEQUENCE OF SnaNodeLinkAdminEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"This table contains the references to link specific tables. If a Node is configured for multiple links, then the Node will have multiple entries in this table. The entries in this table can be generated initially, after initialization of SNA service, by the Agent which uses information from Node configuration file. Subsequent modifications of parameters, creation of new Nodes link entries and deletion of entries is possible. The modification to this table can be saved in the Node configuration file for the next initialization of SNA service, but the mechanism for this function is not defined here."

::= { snaNode 6 }

snaNodeLinkAdminEntry OBJECT-TYPE

SYNTAX SnaNodeLinkAdminEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"Entry contains the configuration information that associates a Node instance to one link instance. The objects in the entry have read-create access. Entry can be created, modified or deleted. The object snaNodeLinkAdminRowStatus is used (set) to create or delete an entry. The object snaNodeLinkAdminSpecific can be set later, after the entry has been created."

INDEX { snaNodeAdminIndex,
snaNodeLinkAdminIndex }

::= { snaNodeLinkAdminTable 1 }

SnaNodeLinkAdminEntry ::= SEQUENCE {
snaNodeLinkAdminIndex
Integer32,

```
snaNodeLinkAdminSpecific
    InstancePointer,
snaNodeLinkAdminMaxPiu
    Integer32,
snaNodeLinkAdminRowStatus
    RowStatus
}
```

snaNodeLinkAdminIndex OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"This value is used to index the instances of objects.
If an Agent creates the entry, then it will assign
this number otherwise a Management Station
generates a random number when it reserves the
entry for creation."

::= { snaNodeLinkAdminEntry 1 }

snaNodeLinkAdminSpecific OBJECT-TYPE

SYNTAX InstancePointer

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This value points to the row in the table
containing information on the link instance.
(e.g., the sdlcLSAdminTable of
the SNA DLC MIB module)."

::= { snaNodeLinkAdminEntry 2 }

snaNodeLinkAdminMaxPiu OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This value identifies the maximum number of octets
that can be exchanged by this Node in one
Path Information Unit (PIU)."

::= { snaNodeLinkAdminEntry 3 }

snaNodeLinkAdminRowStatus OBJECT-TYPE

SYNTAX RowStatus

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This object is used by a Management Station to
create or delete the row entry in the

snaNodeLinkAdminTable.

To activate a row, a Management Station sets the value to 'active (1)' or 'notReady (3)'. Upon successful creation of the row, the Agent automatically creates a corresponding entry in the snaNodeLinkOperTable.

Row deletion can be Management Station or Agent initiated:

- (a) The Management Station can set the value to 'destroy (6)' only when the value of snaNodeLinkOperState of this Link instance is 'inactive (1)'. The Agent will then delete the row corresponding to this Link instance from snaNodeLinkOperTable and from snaNodeLinkAdminTable.
- (b) The Agent detects that a row is in the 'notReady (3)' state for greater than a default period of 5 minutes.
- (c) The Agent will not include a row with RowStatus='notReady (3)', after SNA system re-initialization (e.g., reboot)."

```
::= { snaNodeLinkAdminEntry 4 }
```

```
-- *****
-- The following object is updated when there is a change to
-- the value of any object in the snaNodeLinkAdminTable.
-- *****
```

snaNodeLinkAdminTableLastChange OBJECT-TYPE

SYNTAX TimeStamp

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The timestamp (e.g., the Agent's sysUpTime value) at the last change made to any object in the snaNodeLinkAdminTable, including row deletions/additions (i.e., changes to the snaNodeLinkAdminRowStatus object).

This object can be used to reduce frequent retrievals of the snaNodeLinkAdminTable by a Management Station. It is expected that a Management Station will periodically poll this object and compare its current value with the previous one. A difference indicates that some Node operational information has been changed. Only then will the

Management Station retrieve the entire table."
 ::= { snaNode 7 }

```
-- *****
-- The following table contains the association between
-- Nodes and link identifiers.
-- It provides the current status.
-- *****
```

snaNodeLinkOperTable OBJECT-TYPE

SYNTAX SEQUENCE OF SnaNodeLinkOperEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"This table contains all references to link specific tables for operational parameters. If a Node is configured for multiple links, then the Node will have multiple entries in this table. This table augments the snaNodeLinkAdminTable."

::= { snaNode 8 }

snaNodeLinkOperEntry OBJECT-TYPE

SYNTAX SnaNodeLinkOperEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"Entry contains all current parameters for one Node link. The objects in the entry have read-only access."

AUGMENTS { snaNodeLinkAdminEntry }

::= { snaNodeLinkOperTable 1 }

```
SnaNodeLinkOperEntry ::= SEQUENCE {
    snaNodeLinkOperSpecific
        InstancePointer,
    snaNodeLinkOperMaxPiu
        Integer32
}
```

snaNodeLinkOperSpecific OBJECT-TYPE

SYNTAX InstancePointer

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This value points to the row in the table containing information on the link instance."

```

        (e.g., the sdlcLSOperTable of
        the SNA DLC MIB module)."
 ::= { snaNodeLinkOperEntry 1 }

```

```
snaNodeLinkOperMaxPiu OBJECT-TYPE
```

```
SYNTAX Integer32
```

```
MAX-ACCESS read-only
```

```
STATUS current
```

```
DESCRIPTION
```

```

        "Maximum number of octets that can
        be exchanged by this Node in one Path
        Information Unit (PIU)."
```

```
 ::= { snaNodeLinkOperEntry 2 }
```

```

-- *****
-- The following object is updated when a row is added/deleted
-- from the snaNodeLinkOperTable.
-- *****

```

```
snaNodeLinkOperTableLastChange OBJECT-TYPE
```

```
SYNTAX TimeStamp
```

```
MAX-ACCESS read-only
```

```
STATUS current
```

```
DESCRIPTION
```

```

        "The timestamp of the last
        change made to any object in the snaNodeLinkOperTable,
        including row deletions/additions.
```

```

        This object can be used to reduce frequent
        retrievals of the snaNodeLinkOperTable by a
        Management Station. It is expected that a
        Management Station will periodically poll this
        object and compare its current value with the
        previous one.
```

```

        A difference indicates that some Node operational
        information has been changed. Only then will the
        Management Station retrieve the entire table."
```

```
 ::= { snaNode 9 }
```

```

-- *****
-- Traps
-- *****

snaNodeTraps OBJECT IDENTIFIER ::= { snaNode 10 }

snaNodeStateChangeTrap NOTIFICATION-TYPE
    OBJECTS { snaNodeOperName,
               snaNodeOperState }
    STATUS current
    DESCRIPTION
        "This trap indicates that the operational state
        (i.e., value of the snaNodeOperState object) of a Node
        has changed. The following variables are returned:
        snaNodeOperName - current name of the Node,
        with the instance identifying the Node; and,
        snaNodeOperState - current state after
        the change."
    ::= { snaNodeTraps 1 }

snaNodeActFailTrap NOTIFICATION-TYPE
    OBJECTS { snaNodeOperName,
               snaNodeOperState,
               snaNodeOperActFailureReason }
    STATUS current
    DESCRIPTION
        "This trap indicates a Node activation failure.
        The value of snaNodeOperState indicates the current
        state after the activation attempt.
        The value of snaNodeOperActFailureReason indicates
        the failure reason."
    ::= { snaNodeTraps 2 }

-- *****
-- snaLu group
--
-- It contains Managed Objects related to LUs in general and some
-- specific for LUs of type 0, 1, 2, 3.
-- *****

-- *****
-- The following table contains LU configuration parameters.
-- *****

```

```
snaLuAdminTable OBJECT-TYPE
    SYNTAX SEQUENCE OF SnaLuAdminEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "This table contains LU configuration information.
        The rows in this table can be created and deleted
        by a Management Station.
        Only objects which are common to all types of LUs
        are included in this table."
    ::= { snaLu 1 }
```

```
snaLuAdminEntry OBJECT-TYPE
    SYNTAX SnaLuAdminEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "Contains configuration variables for an LU."
    INDEX { snaNodeAdminIndex, snaLuAdminLuIndex }
    ::= { snaLuAdminTable 1 }
```

```
SnaLuAdminEntry ::= SEQUENCE {
    snaLuAdminLuIndex
        Integer32,
    snaLuAdminName
        DisplayString,
    snaLuAdminSnaName
        DisplayString,
    snaLuAdminType
        INTEGER,
    snaLuAdminDepType
        INTEGER,
    snaLuAdminLocalAddress
        OCTET STRING,
    snaLuAdminDisplayModel
        INTEGER,
    snaLuAdminTerm
        INTEGER,
    snaLuAdminRowStatus
        RowStatus
}
```

```
snaLuAdminLuIndex OBJECT-TYPE
    SYNTAX Integer32
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "This value identifies the unique index for an
```

LU instance within a Node."
 ::= { snaLuAdminEntry 1 }

snaLuAdminName OBJECT-TYPE

SYNTAX DisplayString (SIZE(0..48))

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This value identifies the user configurable name for this LU. If a name is not assigned to the LU, then this object contains a zero length string.

A write operation to this object will not change the operational value reflected in snaLuOperName until the Node has been re-activated (e.g., after the next initialization of the SNA services)."

::= { snaLuAdminEntry 2 }

snaLuAdminSnaName OBJECT-TYPE

SYNTAX DisplayString (SIZE(1..17))

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This value identifies the SNA LU name used in exchange of SNA data.

A write operation to this object will not change the operational value reflected in snaLuOperSnaName until the Node has been re-activated (e.g., after the next initialization of the SNA services)."

::= { snaLuAdminEntry 3 }

snaLuAdminType OBJECT-TYPE

SYNTAX INTEGER {
 other(1),
 lu0(2),
 lu1(3),
 lu2(4),
 lu3(5),
 lu4(6),
 lu62(7),
 lu7(8)
}

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This value identifies the LU type.

A write operation to this object will not change the operational value reflected in snaLuOperAdminType until the Node has been re-activated (e.g., after the next initialization of the SNA services)."

::= { snaLuAdminEntry 4 }

snaLuAdminDepType OBJECT-TYPE

SYNTAX INTEGER {
 dependent(1),
 independent(2)
}

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This value identifies whether the LU is dependent or independent.

A write operation to this object will not change the operational value reflected in snaLuOperDepType until the Node has been re-activated (e.g., after the next initialization of the SNA services)."

::= { snaLuAdminEntry 5 }

snaLuAdminLocalAddress OBJECT-TYPE

SYNTAX OCTET STRING (SIZE(1))

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The local address for this LU is a byte with a value ranging from 0 to 254. For dependent LUs, this value ranges from 1 to 254 and for independent LUs this value is always 0.

A write operation to this object will not change the operational value reflected in snaLuOperLocalAddress until the Node has been re-activated (e.g., after the next initialization of the SNA services)."

::= { snaLuAdminEntry 6 }

snaLuAdminDisplayModel OBJECT-TYPE

SYNTAX INTEGER {
 invalid(1),
 model2A(2),
 model2B(3),

```

        model3A(4),
        model3B(5),
        model4A(6),
        model4B(7),
        model5A(8),
        model5B(9),
        dynamic(10)
    }
MAX-ACCESS    read-create
STATUS        current
DESCRIPTION
    "The value of this object identifies the model type
    and screen size of the terminal connected to the host.
    This is only valid for LU Type 2. The values have
    the following meaning:

    model2A(2) - Model 2 (24 rows x 80 cols) with base
                  attributes
    model2B(3) - Model 2 (24 rows x 80 cols) with
                  extended attributes
    model3A(4) - Model 3 (32 rows x 80 cols) with base
                  attributes
    model3B(5) - Model 3 (32 rows x 80 cols) with extended
                  attributes
    model4A(6) - Model 4 (43 rows x 80 cols) with base
                  attributes
    model4B(7) - Model 4 (43 rows x 80 cols) with extended
                  attributes
    model5A(8) - Model 5 (27 rows x 132 cols) with base
                  attributes
    model5B(9) - Model 5 (27 rows x 132 cols) with
                  extended attributes
    dynamic(10) - Screen size determine with BIND and Read
                  Partition Query.

    In case this LU is not Type 2, then this object
    should contain the invalid(1) value."
 ::= { snaLuAdminEntry 7 }

```

```

snaLuAdminTerm OBJECT-TYPE
    SYNTAX  INTEGER {
        unbind (1),
        termself (2),
        rshutd (3),
        poweroff (4)
    }
    MAX-ACCESS    read-create
    STATUS        current

```

DESCRIPTION

"This value identifies the desired method for deactivation of this LU. This value overrides the default method (snaNodeOperLuTermDefault) for this Node. For LU 6.2, only the value 'unbind (1)' applies.

unbind(1) - terminate the LU-LU session by sending an SNA UNBIND request.

termself(2) - terminate the LU-LU session by sending an SNA TERM-SELF (Terminate Self) request on the SSCP-LU session. The SSCP will inform the remote session LU partner to send an UNBIND request to terminate the session.

rshutd(3) - terminate the LU-LU session by sending an SNA RSHUTD (Request ShutDown) request to the remote session LU partner. The remote LU will then send an UNBIND request to terminate the session.

poweroff(4) - terminate the LU-LU session by sending either an SNA LUSTAT (LU Status) request on the LU-LU session or an SNA NOTIFY request on the SSCP-LU session indicating that the LU has been powered off. Sending both is also acceptable. The result should be that the remote session LU partner will send an UNBIND to terminate the session.

A write operation to this object may immediately change the operational value reflected in snaLuOperTerm depending on the Agent implementation."

::= { snaLuAdminEntry 8 }

snaLuAdminRowStatus OBJECT-TYPE

SYNTAX RowStatus

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This object is used by a Management Station to create or delete the row entry in the snaLuAdminTable.

To activate a row, the Management Station sets the value to 'active (1)' or 'notReady (3)'.

Upon successful creation of the row, the Agent automatically creates a corresponding entry in the snaLuOperTable with snaLuOperState equal to 'inactive (1)'.

Row deletion can be Management Station or Agent initiated:

- (a) The Management Station can set the value to 'destroy (6)' only when the value of snaLuOperState of this LU instance is 'inactive (1)'. The Agent will then delete the row corresponding to this LU instance from snaLuAdminTable and from snaLuOperTable.
- (b) The Agent detects that a row is in the 'notReady (3)' state for greater than a default period of 5 minutes.
- (c) The Agent will not create a row with RowStatus equal to 'notReady (3)', after SNA system re-initialization (e.g., reboot)."

```
::= { snaLuAdminEntry 9 }
```

```
-- *****
-- The following table contains LU state dynamic parameters.
-- *****
```

snaLuOperTable OBJECT-TYPE

SYNTAX SEQUENCE OF SnaLuOperEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"This table contains dynamic runtime information and control variables relating to LUs. Only objects which are common to all types of LUs are included in this table. This table augments the snaLuAdminTable."

```
::= { snaLu 2 }
```

snaLuOperEntry OBJECT-TYPE

SYNTAX SnaLuOperEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"Contains objects reflecting current information for an LU.

Each entry is created by the Agent. All entries have read-only access."

AUGMENTS { snaLuAdminEntry }

```
::= { snaLuOperTable 1 }
```

SnaLuOperEntry ::= SEQUENCE {

snaLuOperName

DisplayString,

```

    snaLuOperSnaName
        DisplayString,
    snaLuOperType
        INTEGER,
    snaLuOperDepType
        INTEGER,
    snaLuOperLocalAddress
        OCTET STRING,
    snaLuOperDisplayModel
        INTEGER,
    snaLuOperTerm
        INTEGER,
    snaLuOperState
        INTEGER,
    snaLuOperSessnCount
        Gauge32
}

```

```

snaLuOperName OBJECT-TYPE
    SYNTAX  DisplayString (SIZE(0..48))
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "User configurable name for this LU. If a name
         is not assigned, then this object contains a
         zero length string."
    ::= { snaLuOperEntry 1 }

```

```

snaLuOperSnaName OBJECT-TYPE
    SYNTAX  DisplayString (SIZE(1..17))
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "The value identifies the current SNA LU name."
    ::= { snaLuOperEntry 2 }

```

```

snaLuOperType OBJECT-TYPE
    SYNTAX  INTEGER {
        other(1),
        lu0(2),
        lu1(3),
        lu2(4),
        lu3(5),
        lu4(6),
        lu62(7),
        lu7(8)
    }
    MAX-ACCESS  read-only

```

STATUS current
DESCRIPTION
 "The value identifies the current LU type."
 ::= { snaLuOperEntry 3 }

snaLuOperDepType OBJECT-TYPE
 SYNTAX INTEGER {
 dependent(1),
 independent(2)
 }
 MAX-ACCESS read-only
 STATUS current
 DESCRIPTION
 "The value identifies whether the LU is currently
 dependent or independent."

 A write operation to this object will
 not change the operational value reflected
 in snaLuOperDepType until the Node has
 been re-activated (e.g., after the next
 initialization of the SNA services)."
 ::= { snaLuOperEntry 4 }

snaLuOperLocalAddress OBJECT-TYPE
 SYNTAX OCTET STRING (SIZE(1))
 MAX-ACCESS read-only
 STATUS current
 DESCRIPTION
 "The local address for this LU is a byte with a value
 ranging from 0 to 254. For dependent LUs, this value
 ranges from 1 to 254; for independent LUs this value
 is always 0."

 A write operation to this object will
 not change the operational value reflected
 in snaLuOperLocalAddress until the Node has
 been re-activated (e.g., after the next
 initialization of the SNA services)."
 ::= { snaLuOperEntry 5 }

snaLuOperDisplayModel OBJECT-TYPE
 SYNTAX INTEGER {
 invalid(1),
 model2A(2),
 model2B(3),
 model3A(4),
 model3B(5),
 model4A(6),
 }

```

        model4B(7),
        model5A(8),
        model5B(9),
        dynamic(10)
    }
MAX-ACCESS    read-only
STATUS        current
DESCRIPTION
    "The screen model type of the terminal connected to
    the host. If this LU is not Type 2, then this
    object should contain the 'invalid(1)' value."
 ::= { snaLuOperEntry 6 }

```

snaLuOperTerm OBJECT-TYPE

```

SYNTAX        INTEGER {
    unbind (1),
    termself (2),
    rshutd (3),
    poweroff (4)
}
MAX-ACCESS    read-only
STATUS        current
DESCRIPTION

```

"The value identifies the current method for deactivation of this LU. This value overrides the default method (snaNodeOperLuTermDefault) for this Node. For LU 6.2, only the value 'unbind (1)' applies.

```

unbind(1) -   terminate the LU-LU session by sending
               an SNA UNBIND request.
termself(2) - terminate the LU-LU session by sending
               an SNA TERM-SELF (Terminate Self) request on
               the SSCP-LU session. The SSCP will inform the
               remote session LU partner to send an UNBIND
               request to terminate the session.
rshutd(3) -   terminate the LU-LU session by sending
               an SNA RSHUTD (Request ShutDown) request to
               the remote session LU partner. The remote LU
               will then send an UNBIND request to terminate
               the session.
poweroff(4) - terminate the LU-LU session by sending
               either an SNA LUSTAT (LU Status) request on
               the LU-LU session or an SNA NOTIFY request on
               the SSCP-LU session indicating that the LU has
               been powered off. Sending both is also
               acceptable. The result should be that the
               remote session LU partner will send an UNBIND

```

to terminate the session."
 ::= { snaLuOperEntry 7 }

snaLuOperState OBJECT-TYPE

SYNTAX INTEGER {
 inactive (1),
 active (2)
 }

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The value identifies the current operational state of this LU.

It has different meanings for dependent and independent LUs.

For dependent LUs the values indicate the following:

inactive (1) - LU didn't receive ACTLU, or
 it received DACTLU, or received ACTLU and sent
 negative response.

active (2) - LU received ACTLU and acknowledged
 positively.

For independent LUs the values indicate the following:

active (2) - the LU is defined and is able to send
 and receive BIND.

inactive (1) - the LU has a session count equal
 to 0."

::= { snaLuOperEntry 8 }

snaLuOperSessnCount OBJECT-TYPE

SYNTAX Gauge32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of currently active LU-LU sessions of this LU.

For the independent LU, if this object has value 0, it indicates that LU is inactive."

::= { snaLuOperEntry 9 }

```
-- *****
-- The following table contains LU session status parameters.
-- *****
```

snaLuSessnTable OBJECT-TYPE

SYNTAX SEQUENCE OF SnaLuSessnEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"This is a table containing objects which describe the operational state of LU sessions. Only objects which are common to all types of LU sessions are included in this table.

When a session's snaLuSessnOperState value changes to 'pendingBind (2)', then the corresponding entry in the session table is created by the Agent.

When the session's snaLuSessnOperState value changes to 'unbound (1)', then the session will be removed from the session table by the Agent."

::= { snaLu 3 }

snaLuSessnEntry OBJECT-TYPE

SYNTAX SnaLuSessnEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"An entry contains dynamic parameters for an LU-LU session.

The indices identify the Node, local LU, and remote LU for this session."

INDEX { snaNodeAdminIndex,
snaLuAdminLuIndex,
snaLuSessnRluIndex,
snaLuSessnIndex }

::= { snaLuSessnTable 1 }

SnaLuSessnEntry ::= SEQUENCE {

snaLuSessnRluIndex

Integer32,

snaLuSessnIndex

Integer32,

snaLuSessnLocalApplName

DisplayString,

snaLuSessnRemoteLuName

DisplayString,

snaLuSessnMaxSndRuSize

INTEGER,

snaLuSessnMaxRcvRuSize

INTEGER,

snaLuSessnSndPacingSize

INTEGER,

snaLuSessnRcvPacingSize

INTEGER,

```

snaLuSessnActiveTime
    TimeStamp,
snaLuSessnAdminState
    INTEGER,
snaLuSessnOperState
    INTEGER,
snaLuSessnSenseData
    OCTET STRING,
snaLuSessnTerminationRu
    INTEGER,
snaLuSessnUnbindType
    OCTET STRING,
snaLuSessnLinkIndex
    Integer32
}

```

snaLuSessnRluIndex OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This value may be used to identify information about the session partner LU in a table of information about remote LUs. Such a table is not defined in this document. If a table of remote LU information is not implemented, or if the table is implemented but it does not contain information about the partner LU for a particular session (as for dependent LU-LU sessions) then this object will have a value of zero."

::= { snaLuSessnEntry 1 }

snaLuSessnIndex OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This value identifies the unique index of the session. It is recommended that an Agent should not reuse the index of a deactivated session for a significant period of time (e.g., one week)."

::= { snaLuSessnEntry 2 }

snaLuSessnLocalApplName OBJECT-TYPE

SYNTAX DisplayString (SIZE(0..48))

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The name of the local application using this LU."

If the local application is unknown, then this object contains a zero length string."
 ::= { snaLuSessnEntry 3 }

snaLuSessnRemoteLuName OBJECT-TYPE

SYNTAX DisplayString (SIZE(0..17))

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"For dependent LUs which are indicated by the snaLuOperDepType object containing the value 'dependent (1)', this object contains the Primary LU (PLU) name. For independent LUs, this object contains the fully-qualified remote LU name of this 6.2 session. A fully qualified name is an SNA NAU entity name preceded by the NetId and a period as the delimiter."

::= { snaLuSessnEntry 4 }

snaLuSessnMaxSndRuSize OBJECT-TYPE

SYNTAX INTEGER (1..8192)

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The maximum RU size used on this session for sending RUs."

::= { snaLuSessnEntry 5 }

snaLuSessnMaxRcvRuSize OBJECT-TYPE

SYNTAX INTEGER (1..8192)

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The maximum RU size used on this session for receiving RUs."

::= { snaLuSessnEntry 6 }

snaLuSessnSndPacingSize OBJECT-TYPE

SYNTAX INTEGER (1..63)

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The size of the send pacing window on this session."

::= { snaLuSessnEntry 7 }

snaLuSessnRcvPacingSize OBJECT-TYPE

SYNTAX INTEGER (1..63)

MAX-ACCESS read-only


```

STATUS    current
DESCRIPTION
    "The size of the receive pacing window on this
    session."
 ::= { snaLuSessnEntry 8 }

```

```

snaLuSessnActiveTime OBJECT-TYPE
    SYNTAX    TimeStamp
    MAX-ACCESS read-only
    STATUS    current
    DESCRIPTION
        "The timestamp (e.g., the Agent's sysUpTime value)
        when this session becomes active."
    ::= { snaLuSessnEntry 9 }

```

```

snaLuSessnAdminState OBJECT-TYPE
    SYNTAX    INTEGER {
        unbound (1),
        bound (3)
    }
    MAX-ACCESS read-write
    STATUS    current
    DESCRIPTION
        "The value indicates the desired operational state of
        the session. This object is used to
        change the operational state of the session.
        A Management Station can only change the operational
        state of the session to 'unbound (1)'."

```

Session deactivation:

If a session is in the operational state 'bound (3)' then setting the value of this object to 'unbound (1)' will initiate the session shutdown.

If a session is in the operational state 'pendingBind (2)' then setting the value of this object to 'unbound (1)' will initiate the session shutdown.

If a session is in the operational state 'pendingUnbind (4)' for an abnormally long period of time (e.g., three minutes) then setting the value of this object to 'unbound (1)' will change the session operational state to 'unbound (1)'.

Note: for dependent LUs, deactivating the session is the same as deactivating the LU."

```

 ::= { snaLuSessnEntry 10 }

```

snaLuSessnOperState OBJECT-TYPE

```
SYNTAX  INTEGER {
    unbound (1),
    pendingBind (2),
    bound (3),
    pendingUnbind (4)
}
```

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The value indicates the current operational state of the session.

'unbound (1)' - session has been unbound;
in this state it will be removed from the
session table by the Agent.

'pendingBind (2)' - this state has different
meanings for dependent and independent LUs;
for dependent LU - waiting for BIND from
the host, for independent LU - waiting for
BIND response. When a session enters this
state, the corresponding entry in the
session table is created by the Agent.

'bound (3)' - session has been successfully bound.

'pendingUnbind (4)' - session enters this state
when an UNBIND is sent and before the
rsp(UNBIND) is received."

::= { snaLuSessnEntry 11 }

snaLuSessnSenseData OBJECT-TYPE

```
SYNTAX  OCTET STRING (SIZE(0..8))
```

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The value identifies the sense code when there is
a BIND failure. It is taken from the negative BIND
response or UNBIND request.

This is displayed as 8 hexadecimal digits."

::= { snaLuSessnEntry 12 }

snaLuSessnTerminationRu OBJECT-TYPE

```
SYNTAX  INTEGER {
    other (1),
    bindFailure (2),
    unbind (3)
}
```

```

    }
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The value identifies the SNA RU that terminated the
        session.
        If the session is not in the unbound state, this object
        has a value of 'other (1)'."
    ::= { snaLuSessnEntry 13 }

```

```

snaLuSessnUnbindType OBJECT-TYPE
    SYNTAX OCTET STRING (SIZE(0..1))
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "If the session is in the unbound state, and it was
        terminated by an UNBIND, then this object contains
        the UNBIND type value (byte 1 of the UNBIND RU);
        otherwise the string is null."
    ::= { snaLuSessnEntry 14 }

```

```

snaLuSessnLinkIndex OBJECT-TYPE
    SYNTAX Integer32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "This value identifies the link over which the session
        passes. It is an index into snaNodeLinkAdminTable.
        If the index value is not known, the value of this
        object shall be zero."
    ::= { snaLuSessnEntry 15 }

```

```

-- *****
-- The following table contains LU sessions statistics dynamic
-- parameters.
-- *****

```

```

snaLuSessnStatsTable OBJECT-TYPE
    SYNTAX SEQUENCE OF SnaLuSessnStatsEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "This table contains dynamic statistics information
        relating to LU sessions.
        The entries in this table augment the entries in
        the snaLuSessnTable and cannot be created by

```

```

        a Management Station."
 ::= { snaLu 4 }

```

snaLuSessnStatsEntry OBJECT-TYPE

SYNTAX SnaLuSessnStatsEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"Contains statistics information for an LU session.

Each entry is created by the Agent.

Objects in this table have read-only access.

Each session from snaLuSessnTable

has one entry in this table."

AUGMENTS { snaLuSessnEntry }

::= { snaLuSessnStatsTable 1 }

SnaLuSessnStatsEntry ::= SEQUENCE {

snaLuSessnStatsSentBytes

Counter32,

snaLuSessnStatsReceivedBytes

Counter32,

snaLuSessnStatsSentRus

Counter32,

snaLuSessnStatsReceivedRus

Counter32,

snaLuSessnStatsSentNegativeResps

Counter32,

snaLuSessnStatsReceivedNegativeResps

Counter32

}

snaLuSessnStatsSentBytes OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of bytes sent by the local LU."

::= { snaLuSessnStatsEntry 1 }

snaLuSessnStatsReceivedBytes OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of bytes received by the local LU."

::= { snaLuSessnStatsEntry 2 }

snaLuSessnStatsSentRus OBJECT-TYPE

```

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
    "The number of RUs sent by the local LU."
 ::= { snaLuSessnStatsEntry 3 }

```

```

snaLuSessnStatsReceivedRus OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The number of RUs received by the local LU."
    ::= { snaLuSessnStatsEntry 4 }

```

```

snaLuSessnStatsSentNegativeResps OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The number of negative responses sent by the
         local LU."
    ::= { snaLuSessnStatsEntry 5 }

```

```

snaLuSessnStatsReceivedNegativeResps OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The number of negative responses received by the
         local LU."
    ::= { snaLuSessnStatsEntry 6 }

```

```

-- *****
-- Traps
-- *****

```

```

snaLuTraps OBJECT IDENTIFIER ::= { snaLu 5 }

```

```

snaLuStateChangeTrap NOTIFICATION-TYPE
    OBJECTS { snaLuOperName,
               snaLuOperSnaName,
               snaLuOperState }
    STATUS current
    DESCRIPTION
        "This trap indicates that the operational state
         (i.e., snaLuOperState value) of the LU has changed.

```

The value of snaLuOperName indicates the name of the LU.

The value of snaLuOperSnaName indicates the SNA name of LU.

The value of snaLuOperState indicates the current state after change."

::= { snaLuTraps 1 }

snaLuSessnBindFailTrap NOTIFICATION-TYPE

OBJECTS { snaLuSessnLocalApplName,
snaLuSessnRemoteLuName,
snaLuSessnOperState,
snaLuSessnSenseData }

STATUS current

DESCRIPTION

"This trap indicates the failure of a BIND.

The value of snaLuSessnLocalApplName indicates the local application name.

The value of snaLuSessnPartnerName indicates the partner name.

The value of snaLuSessnOperState indicates the current state after change.

The value of snaLuSessnBindFailureReason indicates the failure reason.

The Agent should not generate more than 1 trap of this type per minute to minimize the level of management traffic on the network."

::= { snaLuTraps 2 }

```
-- *****
-- snaMgtTools group
--
-- Currently this group contains only one table.
-- *****
```

```
-- *****
-- The following table contains Response Time Monitoring (RTM)
-- configuration information and statistics for LU Type 2s.
-- RTM supports the capability to measure and report end-user
-- response times for dependent LUs. When the RTM state of an LU
-- is 'on', response times for each LU transaction are monitored.
-- A set of ranges is defined (e.g., Range 1 includes the number of
-- transactions with response times less than 1 second) using the
-- "boundary" definitions (e.g., boundary #2 is defined as 3 seconds).
-- A set of counters (one per range) identifies
-- the number of transactions within each response time range.
-- *****
```

snaLuRtmTable OBJECT-TYPE

SYNTAX SEQUENCE OF SnaLuRtmEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"This table contains Response Time Monitoring (RTM) information relating to an LU (Type 2). Each entry corresponds to an LU 2 entry in snaLuAdminTable."

::= { snaMgtTools 1 }

snaLuRtmEntry OBJECT-TYPE

SYNTAX SnaLuRtmEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"Contains RTM information for an LU (Type 2). Each entry is created by the Agent."

INDEX { snaLuRtmPuIndex, snaLuRtmLuIndex }

::= { snaLuRtmTable 1 }

SnaLuRtmEntry ::= SEQUENCE {

snaLuRtmPuIndex

Integer32,

snaLuRtmLuIndex

Integer32,

snaLuRtmState

INTEGER,

snaLuRtmStateTime

TimeStamp,

snaLuRtmDef

INTEGER,

snaLuRtmBoundary1

Integer32,

snaLuRtmBoundary2

Integer32,

snaLuRtmBoundary3

Integer32,

snaLuRtmBoundary4

Integer32,

snaLuRtmCounter1

Counter32,

snaLuRtmCounter2

Counter32,

snaLuRtmCounter3

Counter32,

snaLuRtmCounter4

Counter32,

```

snaLuRtmOverFlows
    Counter32,
snaLuRtmObjPercent
    Integer32,
snaLuRtmObjRange
    INTEGER,
snaLuRtmNumTrans
    Integer32,
snaLuRtmLastRspTime
    Integer32,
snaLuRtmAvgRspTime
    Integer32
}

```

```

snaLuRtmPuIndex OBJECT-TYPE
    SYNTAX Integer32
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "The value identifies the PU 2.0 with which this LU is
        associated."
    ::= { snaLuRtmEntry 1 }

```

```

snaLuRtmLuIndex OBJECT-TYPE
    SYNTAX Integer32
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "The value uniquely identifies an LU in a PU 2.0."
    ::= { snaLuRtmEntry 2 }

```

```

snaLuRtmState OBJECT-TYPE
    SYNTAX INTEGER {
        off(1),
        on(2)
    }
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The value indicates the current RTM state of an LU."
    ::= { snaLuRtmEntry 3 }

```

```

snaLuRtmStateTime OBJECT-TYPE
    SYNTAX TimeStamp
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The timestamp (e.g., the Agent's sysUpTime value)

```


when this session's RTM state (e.g., snaLuRtmState)
changes value."
 ::= { snaLuRtmEntry 4 }

snaLuRtmDef OBJECT-TYPE

SYNTAX INTEGER {
firstChar(1),
kb(2),
cdeb(3)
}

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The value indicates the mode of measurement for this
RTM request. The values have following meaning:
firstChar(1) - time to first character on screen
kb(2) - time to keyboard usable by operator
cdeb(3) - time to Change Direction/End Bracket."

::= { snaLuRtmEntry 5 }

snaLuRtmBoundary1 OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object contains the value of the first boundary
in units of 1/10th of a second."

::= { snaLuRtmEntry 6 }

snaLuRtmBoundary2 OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object contains the value of the second boundary
in units of 1/10th of a second."

::= { snaLuRtmEntry 7 }

snaLuRtmBoundary3 OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object contains the value of the third boundary
in units of 1/10th of a second."

::= { snaLuRtmEntry 8 }

snaLuRtmBoundary4 OBJECT-TYPE

```
SYNTAX Integer32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
    "This object contains the value of the fourth boundary
    in units of 1/10th of a second."
 ::= { snaLuRtmEntry 9 }
```

```
snaLuRtmCounter1 OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "This value indicates the number of transactions which
        fall in the range specified by the first boundary."
    ::= { snaLuRtmEntry 10 }
```

```
snaLuRtmCounter2 OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "This value indicates the number of transactions which
        fall in the range specified by the second boundary."
    ::= { snaLuRtmEntry 11 }
```

```
snaLuRtmCounter3 OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "This value indicates the number of transactions which
        fall in the range specified by the third boundary."
    ::= { snaLuRtmEntry 12 }
```

```
snaLuRtmCounter4 OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "This value indicates the number of transactions which
        fall in the range specified by the fourth boundary."
    ::= { snaLuRtmEntry 13 }
```

```
snaLuRtmOverFlows OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
    STATUS current
```

DESCRIPTION

"This value indicates the number of transactions which exceed the highest range specified by the boundaries."

::= { snaLuRtmEntry 14 }

snaLuRtmObjPercent OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This value indicates the desired percentage of transactions which should be under a designated boundary range indicated by snaLuRtmObjRange."

::= { snaLuRtmEntry 15 }

snaLuRtmObjRange OBJECT-TYPE

SYNTAX INTEGER {
 other(1),
 range1(2),
 range2(3),
 range3(4),
 range4(5),
 range5(6)
}

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This value indicates the designated boundary range to which the snaLuRtmObject refers.

The values have the following meanings:

 other(1) - not specified
 range1(2) - less than boundary 1
 range2(3) - between boundary 1 and 2
 range3(4) - between boundary 2 and 3
 range4(5) - between boundary 3 and 4
 range5(6) - greater than boundary 4."

::= { snaLuRtmEntry 16 }

snaLuRtmNumTrans OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This value indicates the total number of transactions executed since the RTM monitoring began (i.e., snaLuRtmState changed to 'on(2)') for this LU."

::= { snaLuRtmEntry 17 }

snaLuRtmLastRspTime OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This value indicates the response time for the last transaction in units of 1/10th of a second."

::= { snaLuRtmEntry 18 }

snaLuRtmAvgRspTime OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This value indicates the average response time for all transactions in units of 1/10th of a second."

::= { snaLuRtmEntry 19 }

-- *****

-- Conformance information

-- *****

snanauConformance OBJECT IDENTIFIER ::= { snanauMIB 2 }

snanauCompliances OBJECT IDENTIFIER ::= { snanauConformance 1 }

snanauGroups OBJECT IDENTIFIER ::= { snanauConformance 2 }

-- Compliance statements

snanauCompliance MODULE-COMPLIANCE

STATUS current

DESCRIPTION

"The compliance statement for the SNMPv2 entities which implement the snanau MIB."

MODULE -- this module

-- Unconditionally mandatory groups

MANDATORY-GROUPS { snaNodeGroup,
snaLuGroup,
snaSessionGroup }

-- Conditionally mandatory groups

GROUP snaPu20Group

DESCRIPTION

"The snaPu20Group is mandatory only for those entities which implement PU type 2.0"

GROUP snaMgtToolsRtmGroup

DESCRIPTION

"The snaMgtToolsGroup is mandatory only for those entities which implement LU type 2 and RTM."

-- Refinement of requirements for objects access.
-- The Agent which does not implement row creation for
-- snaNodeAdminTable, snaNodeLinkAdminTable and
-- snaLuAdminTable must at least accept
-- objects modification (read-write access instead of
-- read-create).

OBJECT snaNodeAdminName
MIN-ACCESS read-write
DESCRIPTION
 "An Agent is required to implement read-write
 access to this object."

OBJECT snaNodeAdminType
MIN-ACCESS read-write
DESCRIPTION
 "An Agent is required to implement read-write
 access to this object."

OBJECT snaNodeAdminXidFormat
MIN-ACCESS read-write
DESCRIPTION
 "An Agent is required to implement read-write
 access to this object."

OBJECT snaNodeAdminBlockNum
MIN-ACCESS read-write
DESCRIPTION
 "An Agent is required to implement read-write
 access to this object."

OBJECT snaNodeAdminIdNum
MIN-ACCESS read-write
DESCRIPTION
 "An Agent is required to implement read-write
 access to this object."

OBJECT snaNodeAdminEnablingMethod
MIN-ACCESS read-write
DESCRIPTION
 "An Agent is required to implement read-write
 access to this object."

OBJECT snaNodeAdminLuTermDefault

MIN-ACCESS read-write
DESCRIPTION
 "An Agent is required to implement read-write
 access to this object."

OBJECT snaNodeAdminMaxLu
MIN-ACCESS read-write
DESCRIPTION
 "An Agent is required to implement read-write
 access to this object."

OBJECT snaNodeAdminHostDescription
MIN-ACCESS read-write
DESCRIPTION
 "An Agent is required to implement read-write
 access to this object."

OBJECT snaNodeAdminStopMethod
MIN-ACCESS read-write
DESCRIPTION
 "An Agent is required to implement read-write
 access to this object."

OBJECT snaNodeAdminState
MIN-ACCESS read-write
DESCRIPTION
 "An Agent is required to implement read-write
 access to this object."

OBJECT snaNodeLinkAdminSpecific
MIN-ACCESS read-write
DESCRIPTION
 "An Agent is required to implement read-write
 access to this object."

OBJECT snaNodeLinkAdminMaxPiu
MIN-ACCESS read-write
DESCRIPTION
 "An Agent is required to implement read-write
 access to this object."

OBJECT snaLuAdminName
MIN-ACCESS read-write
DESCRIPTION
 "An Agent is required to implement read-write
 access to this object."

OBJECT snaLuAdminSnaName
MIN-ACCESS read-write

DESCRIPTION

"An Agent is required to implement read-write access to this object."

OBJECT snaLuAdminType

MIN-ACCESS read-write

DESCRIPTION

"An Agent is required to implement read-write access to this object."

OBJECT snaLuAdminDepType

MIN-ACCESS read-write

DESCRIPTION

"An Agent is required to implement read-write access to this object."

OBJECT snaLuAdminLocalAddress

MIN-ACCESS read-write

DESCRIPTION

"An Agent is required to implement read-write access to this object."

OBJECT snaLuAdminDisplayModel

MIN-ACCESS read-write

DESCRIPTION

"An Agent is required to implement read-write access to this object."

OBJECT snaLuAdminTerm

MIN-ACCESS read-write

DESCRIPTION

"An Agent is required to implement read-write access to this object."

::= {snanauCompliances 1 }

-- Units of conformance

snaNodeGroup OBJECT-GROUP

OBJECTS { snaNodeAdminName,
snaNodeAdminType,
snaNodeAdminXidFormat,
snaNodeAdminBlockNum,
snaNodeAdminIdNum,
snaNodeAdminEnablingMethod,
snaNodeAdminLuTermDefault,
snaNodeAdminMaxLu,

```

snaNodeAdminHostDescription,
snaNodeAdminStopMethod,
snaNodeAdminState,
snaNodeAdminRowStatus,
snaNodeAdminTableLastChange,
snaNodeOperName,
snaNodeOperType,
snaNodeOperXidFormat,
snaNodeOperBlockNum,
snaNodeOperIdNum,
snaNodeOperEnablingMethod,
snaNodeOperLuTermDefault,
snaNodeOperMaxLu,
snaNodeOperHostDescription,
snaNodeOperStopMethod,
snaNodeOperState,
snaNodeOperHostSscpId,
snaNodeOperStartTime,
snaNodeOperLastStateChange,
snaNodeOperActFailures,
snaNodeOperActFailureReason,
snaNodeOperTableLastChange,
snaNodeLinkAdminSpecific,
snaNodeLinkAdminMaxPiu,
snaNodeLinkAdminRowStatus,
snaNodeLinkAdminTableLastChange,
snaNodeLinkOperSpecific,
snaNodeLinkOperMaxPiu,
snaNodeLinkOperTableLastChange }
STATUS current
DESCRIPTION
    "A collection of objects providing the
    instrumentation of SNA nodes."
 ::= { snanauGroups 1 }

```

```

snaLuGroup OBJECT-GROUP
    OBJECTS { snaLuAdminName,
               snaLuAdminSnaName,
               snaLuAdminType,
               snaLuAdminDepType,
               snaLuAdminLocalAddress,
               snaLuAdminDisplayModel,
               snaLuAdminTerm,
               snaLuAdminRowStatus,
               snaLuOperName,
               snaLuOperSnaName,
               snaLuOperType,
               snaLuOperDepType,

```



```

        snaLuOperLocalAddress,
        snaLuOperDisplayModel,
        snaLuOperTerm,
        snaLuOperState,
        snaLuOperSessnCount }
STATUS   current
DESCRIPTION
    "A collection of objects providing the
    instrumentation of SNA LUs."
 ::= { snanauGroups 2 }

```

```

snaSessionGroup  OBJECT-GROUP
OBJECTS   { snaLuSessnRluIndex,
            snaLuSessnIndex,
            snaLuSessnLocalApplName,
            snaLuSessnRemoteLuName,
            snaLuSessnMaxSndRuSize,
            snaLuSessnMaxRcvRuSize,
            snaLuSessnSndPacingSize,
            snaLuSessnRcvPacingSize,
            snaLuSessnActiveTime,
            snaLuSessnAdminState,
            snaLuSessnOperState,
            snaLuSessnSenseData,
            snaLuSessnTerminationRu,
            snaLuSessnUnbindType,
            snaLuSessnLinkIndex,
            snaLuSessnStatsSentBytes,
            snaLuSessnStatsReceivedBytes,
            snaLuSessnStatsSentRus,
            snaLuSessnStatsReceivedRus,
            snaLuSessnStatsSentNegativeResps,
            snaLuSessnStatsReceivedNegativeResps }
STATUS   current
DESCRIPTION
    "A collection of objects providing the
    instrumentation of SNA sessions."
 ::= { snanauGroups 3 }

```

```

snaPu20Group  OBJECT-GROUP
OBJECTS   { snaPu20StatsSentBytes,
            snaPu20StatsReceivedBytes,
            snaPu20StatsSentPius,
            snaPu20StatsReceivedPius,
            snaPu20StatsSentNegativeResps,
            snaPu20StatsReceivedNegativeResps,
            snaPu20StatsActLus,
            snaPu20StatsInActLus,

```

```
        snaPu20StatsBindLus }
STATUS   current
DESCRIPTION
        "A collection of objects providing the
        instrumentation of PU 2.0."
 ::= { snanauGroups 4 }

snaMgtToolsRtmGroup OBJECT-GROUP
OBJECTS { snaLuRtmState,
          snaLuRtmStateTime,
          snaLuRtmDef,
          snaLuRtmBoundary1,
          snaLuRtmBoundary2,
          snaLuRtmBoundary3,
          snaLuRtmBoundary4,
          snaLuRtmCounter1,
          snaLuRtmCounter2,
          snaLuRtmCounter3,
          snaLuRtmCounter4,
          snaLuRtmOverFlows,
          snaLuRtmObjPercent,
          snaLuRtmObjRange,
          snaLuRtmNumTrans,
          snaLuRtmLastRspTime,
          snaLuRtmAvgRspTime }
STATUS   current
DESCRIPTION
        "A collection of objects providing the
        instrumentation of RTM for SNA LU 2.0."
 ::= { snanauGroups 5 }

-- end of conformance statement

END
```

5. Acknowledgments

The following people greatly contributed to the work on this MIB document: Michael Allen, Robin Cheng, Bill Kwan. Special thanks goes to Dave Perkins for his assistance in reviewing this MIB proposal.

6. References

- [1] IBM, Systems Network Architecture Technical Overview, GC 30-3073-3, March, 1991.
- [2] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Structure of Management Information for version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1442, SNMP Research, Inc., Hughes LAN Systems, Dover Beach Consulting, Inc., Carnegie Mellon University, April 1993.
- [3] McCloghrie, K., and M. Rose, "Management Information Base for Network Management of TCP/IP-based internets - MIB-II", STD 17, RFC 1213, Hughes LAN Systems, Performance Systems International, March 1991.
- [4] Galvin, J., and K. McCloghrie, "Administrative Model for version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1445, Trusted Information Systems, Hughes LAN Systems, April 1993.
- [5] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Protocol Operations for version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1448, SNMP Research, Inc., Hughes LAN Systems, Dover Beach Consulting, Inc., Carnegie Mellon University, April 1993.
- [6] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Textual Conventions for version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1443, SNMP Research, Inc., Hughes LAN Systems, Dover Beach Consulting, Inc., Carnegie Mellon University, April 1993.

7. Security Considerations

Security issues are not discussed in this memo.

8. Authors' Addresses

Zbigniew Kielczewski
Eicon Technology Corporation
2196 32nd Avenue
Montreal, Quebec, Canada H8T 3H7

Phone: 1 514 631 2592
EMail: zbig@eicon.qc.ca

Deirdre Kostick
Bellcore
331 Newman Springs Road
Red Bank, NJ 07701

Phone: 1 908 758 2642
EMail: dck2@mail.bellcore.com

Kitty Shih
Novell
890 Ross Drive
Sunnyvale, CA 94089

Phone: 1 408 747 4305
EMail: kmshih@novell.com

