ISO/CCITT and Internet Management Coexistence (IIMC):

Translation of ISO/CCITT GDMO MIBs to Internet MIBs

(IIMCOMIBTRANS)

February, 1994

Owen Newnan (Editor)

U S WEST Advanced Technologies
4001 Discovery Drive, Suite 190
Boulder, Colorado 80303
onewnan@advtech.uswest.com

Status of this Memo

This document provides information to the network and systems management community. This document is intended as a contribution to ongoing work in the area of multi-protocol management coexistence and interworking. This document is part of a package; see also [IIMCSEC] [IIMCMIB-II] [IIMCPROXY] and [IIMCIMIBTRANS]. Distribution of this document is unlimited. Comments should be sent to the Network Management Forum IIMC working group (iimc@thumper.bellcore.com).

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

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

Please check the 1id-abstracts.txt listing contained in the internet-drafts Shadow Directories on ds.internic.net, nic.nordu.net, ftp.nisc.sri.com, munnari.oz.au to learn the current status of any Internet Draft.

Abstract

This document provides heuristic procedures that translate managed object specifications from ISO/CCITT Guidelines for Definition of Managed Object (GDMO) templates to Internet MIB macros. Currently, some market segments demand SNMP for customer network management, while others require the ISO/CCITT Common Management Information Protocol (CMIP). The approach defined in this document accommodates both, protecting investment already made in management information specifications and minimizing cost to implement a second protocol where the market requires. This translation is designed to: lose as little functionality as possible in translation; minimize need for human involvement during translation; minimize the cost of dual protocol and proxy-based implementations; and support generic network models which span many computer platforms and network elements. This document is intended to encourage standardization of such an approach and promote consistent usage of MIB definitions, independent of protocol.

Table of Contents

1. INTRODUCTION 1

1.1 PROBLEM STATEMENT 1

1.2 OVERVIEW OF IIMC 1

1.3 MIB TRANSLATION PROCEDURES 2

1.4 NATIVE MANAGEMENT MODEL 3

1.5 PROXY MANAGEMENT MODEL 4

1.6 SCOPE OF THIS DOCUMENT 4

1.7 TERMS AND CONVENTIONS 6

2. SNMPV1 TRANSLATION PROCEDURES 9

2.1 RELATIONSHIP TO RFC 1212 9

2.2 MAPPING OF MANAGED OBJECT CLASSES AND ATTRIBUTES 9

2.2.1 Framework 9

2.2.2 Registration of Modules and Tables 10

2.2.3 Structure of Class Tables 11

2.2.4 Structure of Side Tables 11

2.2.5 Enumeration of Attributes 12

2.2.6 Discussion 13

2.2.6.1 Registration 13

2.2.6.2 Rationale for Tables 15

2.2.6.3 Rationale for the ClassInstancePointer 15

2.2.6.4 Relationship Management 16

2.2.6.5 Attributes of top 16

2.2.6.6 ATTRIBUTE GROUPs 17

2.3 MAPPING TO THE SYNTAX CLAUSE 17

2.3.1 Procedure for Simple Input Syntax 17
2.3.1.1 Mapping 17
2.3.1.2 Discussion 19
2.3.2 General Procedure for Complex Input Syntax 19
2.3.2.1 Mapping 19
2.3.2.2 Discussion 20
2.3.3 Procedures for ANY DEFINED BY Syntax 21

2.4 GENERATION OF INTERNET MIB LABELS 23
2.4.1 Notation 23
2.4.2 Abbreviation Procedure 24
2.4.3 Mapping Procedure 24
2.4.4 Discussion 27

2.5 MAPPING TO THE ACCESS CLAUSE 27

2.6 MAPPING TO THE STATUS CLAUSE 27

2.7 MAPPING TO THE DESCRIPTION CLAUSE 28

2.8 MAPPING TO THE REFERENCE CLAUSE 29

2.9 MAPPING TO THE INDEX CLAUSE 31
2.9.1 Procedure 31
2.9.2 Discussion 31

2.10 MAPPING TO THE DEFVAL CLAUSE 32

2.11 TRANSLATION OF ACTIONS 32
2.11.1 Procedure 32
2.11.2 Discussion 33

2.12 TRANSLATION OF NOTIFICATIONS 33
2.12.1 Procedure 33
2.12.2 Discussion 35

2.13 TRANSLATION OF DELETE OPERATIONS 36

2.14 TRANSLATION OF CREATE OPERATIONS 36
2.14.1 Overview 36
2.14.2 Status of notInService 36
2.14.3 Status of notReady 37
2.14.4 Row Creation 37
2.14.5 ANY DEFINED BY Syntax 37
2.14.6 Discussion 38

3. CONSTRAINTS ON SNMPV1 PROTOCOL USAGE 39

3.1 APPROACH 39

3.2 DISCUSSION 40

4. SNMPV2 TRANSLATION PROCEDURES 41

4.1 GENERAL APPROACH 41

4.2 OBJECT DEFINITIONS 41
4.2.1 Selected Procedures from RFC 1452 41
4.2.2 Additional Procedures 42
4.2.3 Discussion 43

4.3 TRAP DEFINITIONS 43
4.3.1 RFC 1452 Procedure 43
4.3.2 Additional Procedures 44

5. CONSTRAINTS ON SNMPV2 PROTOCOL USAGE 45

5.1 PROCEDURE 45

5.2 DISCUSSION 46

6. STATEMENTS OF COMPLIANCE AND CONFORMANCE 47

6.1 MAPPING OF THE OBJECT-GROUP MACRO 47

6.2 MAPPING OF THE MODULE-COMPLIANCE MACRO 48
6.2.1 General Procedure 48
6.2.2 Translation Compliance Proforma 49
6.2.2.1 Proforma Overview 49
6.2.2.2 Proforma Text 50

6.3 MAPPING OF THE AGENT-CAPABILITIES MACRO 51

6.4 MAPPING OF THE MANAGEMENT OBJECT CONFORMANCE STATEMENT (MOCS) 51

7. COMPLIANCE AND CONFORMANCE 53

7.1 COMPLIANCE 53
7.1.1 All Translated MIB Modules 53
7.1.2 SNMPv1 Translated MIB Modules 53
7.1.3 SNMPv2 Translated MIB Modules 54

7.2 CONFORMANCE 54

ANNEX A (INFORMATIVE): DEFINITION OF MANAGEMENT INFORMATION (DMI) EXAMPLE A-1

A.1 INPUT ISO/CCITT MANAGEMENT INFORMATION BASE A-1

A.2 OUTPUT SNMPV1 MANAGEMENT INFORMATION BASE A-11

A.3 OUTPUT SNMPV2 MANAGEMENT INFORMATION BASE A-32

A.4 OUTPUT TEXTUAL-CONVENTIONS MACROS A-53

A.5 OUTPUT MODULE-COMPLIANCE MACRO A-54

A.6 OUTPUT AGENT-CAPABILITIES MACRO A-57

A.7 APPLICABILITY OF MOCS A-57

ANNEX B (NORMATIVE): SUPPORTING MODULES B-1

B.1. SNMPV1 SUPPORT OBJECTS B-1

B.2. SNMPV2 SUPPORT OBJECTS B-7

B.3. TEXTUAL CONVENTIONS B-12

B.4. MODULE-COMPLIANCE STATEMENT FOR SUPPORT OBJECTS B-17

ANNEX C: GLOSSARY C-1

ANNEX D: REFERENCES D-1

List of Figures

FIGURE 1. MIB TRANSLATION 2

FIGURE 2. NATIVE MANAGEMENT 3

FIGURE 3. PROXY MANAGEMENT 4

FIGURE 4. SAMPLE INPUT REGISTRATION TREE (PARTIAL) 13

FIGURE 5. SAMPLE OUTPUT REGISTRATION TREE (PARTIAL) 14

List of Tables

TABLE 1. TEXTUAL CONVENTIONS 7

TABLE 2. SIMPLE INPUT SYNTAX MAPPING 18

TABLE 3. VARIABLE LABEL GENERATION 23

TABLE 4. CLASS TABLE LABELS 25

TABLE 5. SIDE TABLE LABELS 26

TABLE 6. ANY TABLE LABELS 27

TABLE 7. ACTION SYNTAX MAPPING 32

TABLE 8. SNMPV1 ERROR MAPPING 39

TABLE 9. SNMPV2 ERROR MAPPING 45

REVISION HISTORY

Issue 1.0, October 1993

This is the first issue of this document. The internet draft <draft-newnan-isomib-internet-03>, dated February, 1994, is identical in content to Issue 1.0, October 1993. It has been reformatted for posting as an internet draft.

1. INTRODUCTION

This section provides an overview of ISO/CCITT and Internet Management Coexistence (IIMC) activities, insight into the problem being addressed by IIMC, and a brief introduction to the strategy adopted by IIMC: use of translated MIBs in either a proxy or native implementation. The section concludes by describing the scope of this document, and terms and conventions used by this document.

1.1 PROBLEM STATEMENT

The need for enterprise network management has been addressed by development of network management standards within various communities, most notably the ISO/CCITT and Internet communities.

- The ISO/CCITT community developed the Common Management Information Protocol (CMIP) [5], and related SMI documents [6,7,8].

- The Internet community developed the Simple Network Management Protocol (SNMP) [11], and its successor, SNMPv2 [RFC1448]. The Internet SMI is defined in [10] and [16].

These standards share a nearly common management model, but diverge due to differing management philosophies. Although functionally similar, the Internet and ISO/CCITT protocols and SMIs differ in terms of their complexity and specific operations. Business requirements for end-to-end enterprise management include the need to integrate the management of many different devices, potentially owned or administered by many independent organizations. This requires components to be accessed by ISO/CCITT management, Internet management, and proprietary management mechanisms in a manner which presents a unified view of the network, despite protocol and SMI differences.

For example, many telecommunications and computer vendors, represented by organizations such as the Network Management Forum (NMF), and the U.S. government, as specified in the Government Network Management Profile (GNMP) Version 1.0 [25], have based their enterprise management model on the ISO/CCITT management model. These organizations are particularly interested in integrated management of devices that use the Internet management. This interest is primarily due to the widespread commercial implementation and use of such devices, especially devices that use the Internet TCP/IP protocol suite.

1.2 OVERVIEW OF IIMC

The ISO/CCITT and Internet Management Coexistence (IIMC) package includes the following documents.

IIMCIMIBTRANS	Translation of Internet MIBs to ISO/CCITT GDMO MIBs [23]
IIMCOMIBTRANS	Translation of ISO/CCITT GDMO MIBs to Internet MIBs
IIMCMIB-II	Translation of Internet MIB-II (RFC 1213) to ISO/CCITT GDMO MIB [20]

IIMCPROXY ISO/CCITT to Internet Management Proxy [21]

IIMCSEC ISO/CCITT to Internet Management Security [22]

These documents together comprise a package aimed at integrating ISO/CCITT-based and Internet-based management systems.

IIMC specifications address the problem that end-to-end management requires an integrated, unified view of the managed network, despite differences in management protocol and information structure. Integrated management can be facilitated by the development of "proxy" mechanisms which translate between functionally equivalent service, protocol, and SMI differences to create this unified view. MIB translation procedures can be used to support proxy management, as well as to take advantage of existing MIB definition and avoid duplication of effort. In this way, commercial investment in both ISO/CCITT and Internet-based management technologies can be preserved through deployment of common methods and tools which support integration.

This overall strategy was outlined in a joint publication developed by the NM Forum and X/Open entitled "ISO/CCITT and Internet Management: Coexistence and Interworking Strategy" [24]. The documents included in the IIMC package are the next level of detailed specifications which implement several of the methodologies identified in the strategy. Additional specifications may be defined in the future.

1.3 MIB TRANSLATION PROCEDURES

The foundation of IIMC is provided by a pair of Management Information Base (MIB) translation procedures.

- IIMCIMIBTRANS [23] specifies translation procedures for converting MIBs from Internet MIB macro format into ISO/CCITT GDMO template format.

- IIMCOMIBTRANS specifies translation procedures for converting MIBs from ISO/CCITT GDMO template format into Internet MIB macro format.

The IIMC approach is to specify direct translation procedures which yield a pair of functionally-equivalent MIBs, as shown in Figure 1.

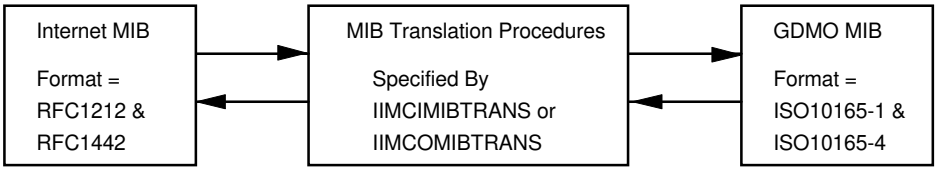


Figure 1. MIB translation.

MIBs translated by these procedures may be used to take advantage of existing MIB definitions when business needs require deployment in a different management environment. Translated MIBs may also be used to provide uniformity when multiple management environments are supported by a single system (e.g., dual stack managers). Finally, IIMC MIB translation procedures may be used to support service emulation by a proxy.

1.4 NATIVE MANAGEMENT MODEL

The basic model for ISO/CCITT and Internet management is illustrated in the following diagram.

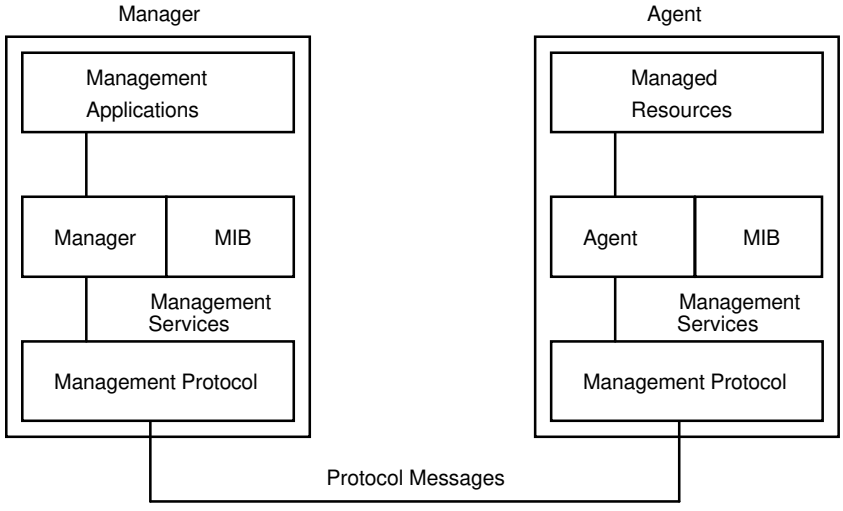


Figure 2. Native management.

Within IIMC documents, this model is referred to as the "native" management model. MIBs translated using IIMC procedures can be used by "native" agent implementations. For example, an ISO/CCITT agent can make visible TCP/IP managed resources using the translated GDMO version of the Internet MIB-II [13] specified by [20]. Dual-stack managers or agents may also be implemented which support both the original MIB and the translated MIB generated using IIMC-specified procedures.

1.5 PROXY MANAGEMENT MODEL

The basic model for ISO/CCITT to Internet proxy management is illustrated in the following diagram. This proxy is specified by this document. A similar approach could also be taken to specify an Internet to ISO/CCITT proxy, although no such IIMC document is currently specified.

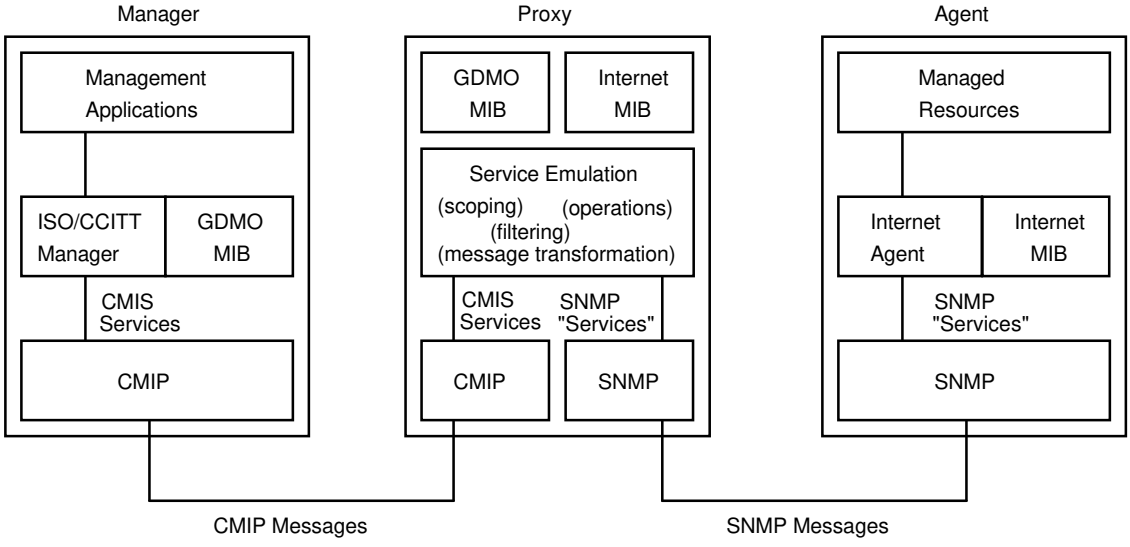


Figure 3. Proxy management.

This ISO/CCITT to Internet proxy provides emulation of CMIS services by mapping to the corresponding SNMP message(s) necessary to carry out the service request. The service emulation allows management of Internet objects by an ISO/CCITT manager. The left hand side of the proxy behaves like an ISO/CCITT agent, communicating with the ISO/CCITT manager using CMIP protocols. The right hand side of the proxy behaves like an Internet manager, communicating with the Internet agent using SNMP protocols.

The proxy relies on the existence of a pair of directly-related MIB definitions, where the Internet MIB has been translated into ISO/CCITT GDMO using the procedures specified in IIMCIMIBTRANS. The proxy uses these MIB definitions and rules to provide run-time translation of management information carried in service requests and responses.

The proxy is designed with a specified interface between the proxy and the underlying protocol stacks, and so deals primarily in terms of CMIS services and SNMP "services". The proxy emulates services such as CMIS scoping and filtering, processing of CMIP operations, and forwarding/logging of CMIS notifications by performing a mapping process which must be tailored for each protocol (for example, SNMPv1 and SNMPv2 are variants of the same protocol mapping process).

1.6 SCOPE OF THIS DOCUMENT

This document (IIMCOMIBTRANS) specifies a heuristic method to translate from ISO/CCITT GDMO MIB specifications to Internet SNMPv1 and SNMPv2 MIB macro specifications. The method is intended to meet six objectives.

- Lose as little functionality as possible in translation.

- Minimize need for human involvement in translation.
- Minimize cost to implement multi-stack (e.g., CMIP, SNMPv1 and SNMPv2) and proxy-based applications.
- Support an end-to-end view (e.g., Generic Network Model) of distributed information resources that span many computing platforms and network elements.
- Support translation to both SNMPv1 and SNMPv2 while minimizing differences between the two mappings. An important corollary is to generate objects and traps/notifications that are syntactically and semantically equivalent and can therefore be registered the same.
- Within constraints of the other objectives, generate MIBs that are as consistent with customary Internet usage as possible. In particular, these should be compatible with common SNMP parsers.

While entirely mechanized translation from an ISO/CCITT GDMO MIB to an Internet MIB is not always possible, the intent is to mechanize the process as much as possible and supply reasonable defaults that may be tempered by human judgment.

This specification provides core methodology such that a GDMO template specification can be translated to Internet MIB macro notation. It thus supports native mode implementations. Proxy implementations are beyond the scope of this document. However, in the longer term, MIBs translated using this method could be used in conjunction with a proxy architecture to enable interworking between ISO/CCITT managers and Internet agents.

The body of this document is composed of the following sections.

- Introduction
- SNMPv1 Translation Procedures
- Constraints on SNMPv1 Protocol Usage
- SNMPv2 Translation Procedures
- Constraints on SNMPv2 Protocol Usage
- Statements of Compliance and Conformance
- Compliance and Conformance

Sample inputs and translated output (informative) appear in annexes. Annex B also provides some supporting Internet macro object specifications to assist in MIB translation and implementation.

Two outstanding issues have been identified that are outside the scope of the present version of this document, but may be addressed in future versions:

- definition of a Proxy for translated ISO/CCITT GDMO MIBs, and
- definition of "Pragmas" to document human overrides to algorithmic translation in a systematic and machine readable form. Among other things, this might facilitate generation of proxies.

This specification assumes existence of appropriate mechanisms and procedures for registry of translated objects. What those procedures might be and where such objects should be registered, however, are beyond the scope of this document.

1.7 TERMS AND CONVENTIONS

This document assumes the reader is familiar with the concepts and vocabularies of Internet and ISO/CCITT management. In cases where there might be confusion between the two, words such as "ISO/CCITT", "GDMO" and "Internet" are inserted to avoid ambiguity.

The term "SNMP" is used throughout this document generically to indicate both the SNMPv1 and SNMPv2 frameworks, unless a distinction needs to be made. The terms SNMPv1 and SNMPv2 are used to refer to the respective frameworks specifically.

Formal language constructs from the various MIB notations (ATTRIBUTE, OBJECT-TYPE, SYNTAX, etc.) are used verbatim in this text. Plurals of such keywords ending in upper case are expressed by appending "s" or "es" (e.g., NOTIFICATIONs) while those ending in lower case are suffixed with "'s" (e.g., TruthValue's).

The terms "class" and "attribute" when expressed in lower case are generic, referring either to ISO/CCITT MANAGED OBJECT CLASSes and ATTRIBUTEs (respectively) or their translated Internet MIB counterparts.

The term "arc" means a single level of branching within an Abstract Syntax Notation One (ASN.1) registration tree.

The terms "enumerate" and "explode" are used synonymously to describe the process of translating ATTRIBUTEs and their values to OBJECT-TYPE macros.

The term "scalar" when applied to a conceptual row means a value that can be expressed as single column.

A "registry family" is defined to be a set of ASN.1 OBJECT IDENTIFIER arcs and nodes sharing a common immediate parent.

Identifiers of Internet MIB macro constructs and GDMO templates are referred to as "labels".

The following notation is used to specify mapping rules.

- Symbols enclosed in quotes are literals.
- Symbols enclosed in angle brackets ("<>") are variables that have different string values depending on instance or context. Variables may be embedded within quoted string literals. In this case, the value for the variable is substituted within the string to arrive at the mapped output.
- Double upended bars ("||") signify the concatenation operator. Concatenate strings without intervening punctuation or white space to arrive at the resulting label.
- Asterisks ("*") indicate that multiple instances of a variable may apply, in which case their values are concatenated to arrive at the output.

Internet conventions are observed throughout the text. Customary SNMP types and textual conventions [17] are produced in the translated output whenever there is a straightforward mapping. New textual conventions may also be introduced in the act of translation. Considering the need to minimize differences between SNMPv1 and SNMPv2 outputs, the SNMPv1 outputs are designed to anticipate SNMPv2 usage.

This specification also introduces SNMPv2 textual conventions to support common constructs appearing in translated MIBs. The following table briefly introduces these conventions — developed further in the body of the text and formally defined (using macro notation) in Annex B.3.

Pointer	A generalized convention that can point to any SNMP MIB object — or to none.
ActionTrigger	Initiates simulation of a translated ACTION.
ActionStatus	Reflects status of a simulated ACTION.
AnyDefinedBy	Facilitates translation of the ASN.1 ANY DEFINED BY construct.
BitString	Equivalent to the ASN.1 BIT STRING type.
ChoiceOfIntOrOID	Simplifies translated syntax of an ASN.1 construct that is a choice of INTEGER or OBJECT IDENTIFIER.
ClassInstancePointer	Points to a instance of an object group; or none.
Complex	Indicates that translated syntax maps to multiple objects — but not multiple rows.
MultiComplex	Indicates that translated syntax maps to multiple objects that can be multiply occurring.
GraphicString	Maps the ASN.1 construct to SNMP.
Parent	Points to the immediately superior object with respect to ISO/CCITT management containment relationships.
Recursion	Points to syntax that is nested at the next lower level of recursion.
RecursivelyNested	Indicates whether particular syntax is recursively nested.
TableIndex	Provides a compact and potentially arbitrary (sub)index for a translated table.

Table 1. Textual conventions.

An SNMPv2 textual convention may imply compliance and/or conformance. The term "observe" when used below means either or both as applicable.

For purposes of brevity, the label of a textual convention may be referenced as follows.

- A noun, meaning a conceptual column observing that convention or specific entry of that column, depending on the context. Thus "a ClassInstancePointer" might imply "an instance of an object observing the ClassInstancePointer textual convention".

- An adjective, referring to a type of conceptual column, e.g., "a TruthValue type".

To align SNMPv1 and SNMPv2 mappings — and facilitate common discussion — this specification requires SNMPv1 translations to observe the same textual conventions as SNMPv2, manifest in IMPORTS statements and the values of SYNTAX clauses. However, some SNMPv1 compilers can not support the TEXTUAL-CONVENTION macro. In such cases, equivalent ASN.1 type references can be introduced to allow compilation (e.g., insert the type definition Pointer ::= OBJECT IDENTIFIER instead of importing the Pointer TEXTUAL-CONVENTION macro).

2. SNMPV1 TRANSLATION PROCEDURES

2.1 RELATIONSHIP TO RFC 1212

While *translation* per se has not previously been widely investigated, RFC 1212 [12] does provide advice for adopting MIB objects from ISO/CCITT GDMO to Internet MIB macros. RFC 1212 advises a minimalistic approach to MIB specification, discouraging carryover of the complexities often found in ISO/CCITT GDMO specifications. Thus, it does not so much tell how to translate a MIB as provide advice for borrowing elements of a ISO/CCITT GDMO specification and constructing a MIB more in keeping with Internet philosophy.

The translation procedure provided here seeks instead to provide as faithful a translation as possible, in order to support the objectives identified in Section 1.

2.2 MAPPING OF MANAGED OBJECT CLASSES AND ATTRIBUTES

This subsection includes:

- Framework,
- Registration of Modules and Tables,
- Structure of Class Tables,
- Structure of Side Tables,
- Enumeration of Attribute Values, and
- Discussion.

2.2.1 Framework

The framework that follows establishes the general approach to translation. It also introduces associated vocabulary used throughout the remainder of the text.

Every MANAGED OBJECT CLASS input maps to a corresponding object group, the "class group". Each class group consists of one or more Internet MIB tables that are logically linked to each other, as follows.

- A "class table" corresponds to the class as a whole. It contains objects representing ATTRIBUTEs and additional objects that reflect status and containment relationships.
- Zero or more "side tables" accommodate complex (compound or multiply occurring) ATTRIBUTE values. Subsection 2.2.4 provides instruction on generation of side tables.

For each managed object mapped, there is a single entry in the corresponding class table, the "class entry". Associated with this may be zero or more "side entries" that accommodate complex ATTRIBUTE syntaxes. A "class instance" is a class entry together with any associated side entries.

A ClassInstancePointer is an OBJECT IDENTIFIER that discriminates (points to) a class instance or to an instance of a native (non-translated) SNMP object group. This textual convention is specified in Annex B.3.

The ClassInstancePointer is used in lieu of the OSI Distinguished Name (DN) and similar constructs, such as the [7]managedObjectInstance ATTRIBUTE. It is analogous to the DN in that it both names a particular instance of a managed object class and implies the type of the object pointed to. It is more compact but less mnemonic than the DN.

2.2.2 Registration of Modules and Tables

Note that examples appearing in this section and throughout this text reflect the sample input and outputs of Annex A. The reader may also wish to refer to section 2.2.6.1 below, which provides further discussion and illustration of registration principles.

Begin translation of a registry family of MANAGED OBJECT CLASS specifications by allocating a node under which the translated modules will be registered, the "output module subtree". Also assign a brief naming prefix to assist in labeling the contents of a corresponding ASN.1 module to be generated. This must be alphanumeric, beginning with a lower case letter.

Not all objects or notifications of an input registry family need be translated together. If they are, the result is referred to as full translation. If not, that is partial translation.

For a full translation, register the resulting ASN.1 module as:

"{ <output module subtree> 1 }"

and name the resulting module:

"MIB" || <naming prefix> || "SNMPv1", e.g., MIBsmi2SNMPv1.

For a partial translation, register the resulting ASN.1 module as:

"{ <output module subtree> 1 <least translated arc number> }"

and name the resulting module:

"MIB" || <naming prefix> || "SNMPv1Increment" || <least translated arc number>

where the least translated arc number is the lowest relative arc number corresponding to any object translated. If no object is translated, the <least translated arc number> must be assigned manually to avoid possible collision..

For each MANAGED OBJECT CLASS in the input registry family, define a corresponding Internet MIB object group — the class group. First, allocate a node under which the translated class groups will be registered, the "output class subtree". Beneath this output class subtree, assign a node to each class group, keeping the same relative numbering under the output class subtree as assigned to the equivalent MANAGED OBJECT CLASSes in the input class subtree. Next, assign arcs to the various tables within each class group.

- Assign the class table arc 1 beneath the class group arc:

  "{ <output class subtree> <class group> 1 }".

- Assign arcs for side tables in ascending order beneath the class group arc, beginning with arc number two:

"{ <output class subtree> <class group> 2 }", etc..

Note: Since all ATTRIBUTEs expand into class tables or side tables, generated class groups never contain scalar values.

For each table in the class group, define a table entry object and syntax consistent with RFC 1212 usage for table entries (naming of objects and entries is addressed below).

2.2.3 Structure of Class Tables

For the entries of a class table, allocate the following conceptual columns and associated arcs sequentially.

Reserve arc 0 beneath the class entry arc per the Internet Structure of Management Information (SMI).

Assign arc 1 to the "class entry index":

"{ <output class subtree> <class group> <class table> 1 }".

This is of textual convention TableIndex, i.e., provides the unique index for the entry but may otherwise be an arbitrary integer value without particular meaning. This object is not-accessible.

Note: The value of a ClassInstancePointer can be determined by concatenating:

- *The OBJECT IDENTIFIER of the class entry,*
- *The value 1 (arc number for the class entry index), and*
- *The subidentifier of that particular entry.*

Here, "concatenating" means arriving at a simple combined sequence of arc values, without repeat counts or nesting. The ClassInstancePointer is used in the translated MIB in lieu of the OSI Distinguished Name's and ObjectInstance's that represent relationships between managed objects. As discussed in Section 2.2.6.3, no direct translation of Distinguished Names is provided.

Assign one conceptual column for each ATTRIBUTE of the translated class. Thus, assign arcs 2 through <n+1> sequentially to the <n> conceptual columns of the class table corresponding to the <n> translated ATTRIBUTEs.

Note: Instructions for mapping ATTRIBUTE syntax to conceptual columns of the class and side tables and assigning ACCESS levels appear in subsection 2.3.

Assign arc <n+2> to the parent object, a ClassInstancePointer that points to the superior (parent) class entry. This has read-only ACCESS.

If the object could possibly be created or deleted by a management station, assign the last arc to a RowStatus object. This value has ACCESS of read-write and conforms to a subset of the SNMPv2 RowStatus textual convention as documented below. This convention allows creation and deletion of class instances.

2.2.4 Structure of Side Tables

Assign arcs of side tables sequentially as follows.

Reserve arc 0 beneath the class entry arc per the Internet Structure of Management Information (SMI).

If the side table can have multiple entries per entry in the immediately superior table (which may be a class table or another side table) reserve arc 1 for the TableIndex. This is used to qualify the index(es) of superior table(s) to uniquely identify a particular entry of the table. It is not-accessible.

Assign the next <n> arcs to the <n> conceptual columns that correspond to the ATTRIBUTE's complex value.

If the syntax being translated for this entry is recursive, i.e., may contain itself directly or indirectly, assign one arc for each of the following objects.

- An object of textual convention Recursion, used to point to nested recursive syntax (discussed below in Mapping to the SYNTAX clause). Its ACCESS is read-write if the management station is allowed to update the syntax represented by the row, otherwise, access is read-only.
- An object of textual convention RecursionFlag, used to indicate whether a conceptual row is pointed to by a Recursion pointer. ACCESS is read-only.

If the object could possibly be created by a management station, assign to the final conceptual column a RowStatus object. This has ACCESS of read-write and conforms to the subset RowStatus convention.

2.2.5 Enumeration of Attributes

All else being equal, enumerate PACKAGEs and their contents based upon a left-to-right scanning of the input ISO/CCITT GDMO specification. ATTRIBUTEs and their syntaxes may be enumerated multiple times when translating a MIB module, once for every template in which they are referenced. Thus if an ATTRIBUTE is included in the PACKAGEs of two MANAGED OBJECT CLASSes, it will be enumerated twice in the output — once for each class group. Reuse of a pre-existing ATTRIBUTE translation is encouraged, and can be highlighted by using the REINSTANTIATES scannable REFERENCE clause described in Section 2.8.

Note: Use of partial "proforma" translations might facilitate reinstantiation and make translation more efficient; such methods are currently beyond the scope of this specification.

Enumerate ATTRIBUTEs and their syntaxes in the same sequence as declared in a PACKAGE. These translate to OBJECT-TYPEs that are assigned successive arcs in the class table.

Enumerate all components of superclasses before those of subclasses. Thus where a package is refined by a subclass, first enumerate all components of the superclass, preserving the sequence of the superclass PACKAGEs. Explode ATTRIBUTEs of a subclass later, enumerating them in order of specification within the subclass.

In the case of multiple inheritance, enumerate contents of superclasses left-to-right and depth first, i.e., enumerate all components attributable to one immediate superclass (and subclasses) before proceeding to the next.

2.2.6 Discussion

2.2.6.1 Registration

Figure 4 illustrates the typical registration structure of an OSI MIB specification. Registration arcs for documents (modules), classes, notifications, etc. that correspond to the specification are organized in a common registry family. The root of this family is the "input subtree" — the node labeled part2 in the example.

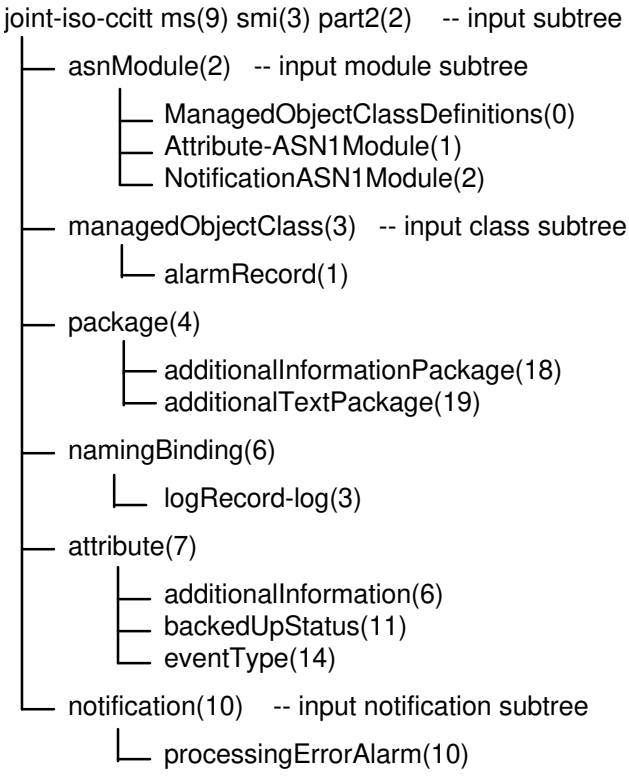


Figure 4. Sample input registration tree (partial)**.**

Assign equivalent structure to the output, rooted in the "output subtree". The arc smi2Translated, for example, is (the root of) the output subtree shown in Figure 5.

Number output arcs the same as equivalent input arcs in assigning registration for the output module, class and notification subtrees. If the input subtrees do not have this structure, still organize the output subtree in this way to simplify registration.

Note: Assignment of naming prefixes and output subtrees are required activities, however, procedures for these are outside the scope of this document.

Assign these arcs labels that are similar but not identical to their analogs within the input subtree. In the Annex A example translation, input subtree labels "smi part2" resulted in the output subtree label "smi2Translated".

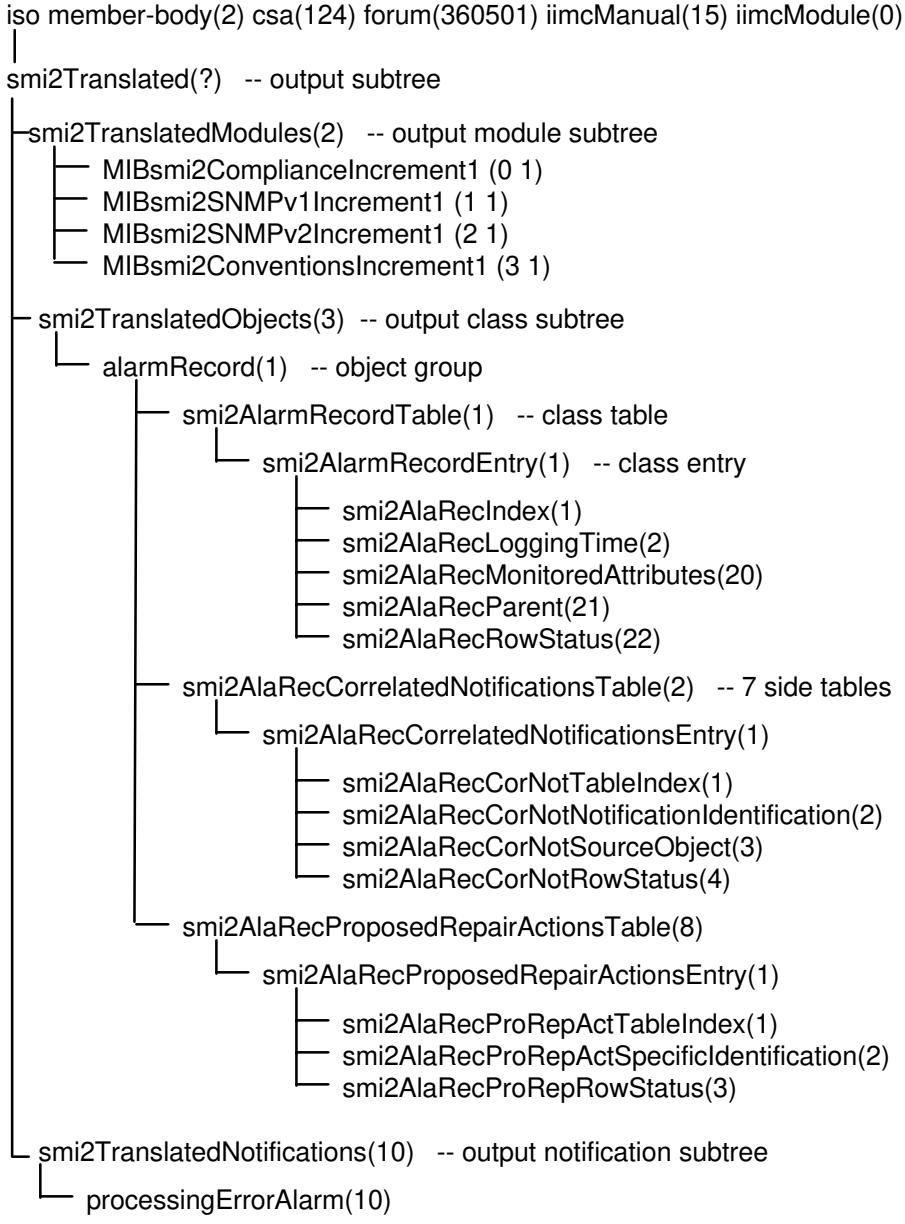


Figure 5. Sample output registration tree (partial)**.**

The structure of an output module subtree is as follows for a full translation (not all arcs are present in all cases):

Arc 0: Compliance Statement.
Arc 1: SNMPv1 Translation.
Arc 2: SNMPv2 Translation.
Arc 3: Translated Textual Conventions.

However, incremental translation requires a mapping convention to assure unique registration of the translated modules. This requires an additional level of qualification for a partial translation, hence the use of the least translated arc number.

Thus, since the Annex A example reflects an incremental translation and alarmRecord is the lowest numbered arc of any object translated (1), the translated SNMPv1 module is registered as { smi2TranslatedModules 1 1 }. Similar "increment" qualifiers apply to the other modules as well.

2.2.6.2 Rationale for Tables

RFC 1212 recommends defining a table for every object group that can be multiply occurring within an agent. It would be very unusual for a MANAGED OBJECT CLASS not to have potentially multiple occurrences, especially given a generic network model that spans systems. Therefore, all object classes map to tables. The tabular approach has additional advantages.

- It supports default values for new object instances.
- It is easy to mechanize, since there is no need to recognize special cases that are not multiply occurring.

2.2.6.3 Rationale for the ClassInstancePointer

The notion of a ClassInstancePointer is introduced since direct translation of Distinguished Name appears impractical for multiple reasons.

- A possible ambiguity arises since NAME BINDINGs allow different object instances of the same MANAGED OBJECT CLASS to exist under parent objects of different classes. Likewise, different classes can have the same syntax for their distinguished attribute(s). Thus, a simple sequence of attribute values is not sufficient to uniquely distinguish an object instance.
- Furthermore, NAME BINDINGs allow different instances of the same class to be subordinate to different types of parent. They even allow instances of a class to be contained recursively within others of the same class. Thus, in directly translating the DN one cannot assume a fixed sequence of parameters as required for an INDEX clause (DNs for different instances of the same object class may have different numbers of RDNs).

Such problems could in theory be resolved by fully translating the Distinguished Name and incorporating Attribute Type's as well as Attribute Value's into the subidentifier OID (in which case the INDEX clause would only need to specify one index, an OBJECT IDENTIFIER). While that is in theory possible, it would result in exceedingly verbose subidentifiers (on the order of 70 octets rather than 7).

This is of serious concern due to PDU length restrictions of SNMP. RFC 1212 proposes a "rule of twenty," i.e., no more than twenty attributes per operation. That guideline was designed for

relatively compact subidentifiers. When using an RDN for an INDEX, this would more likely amount to a "rule of three," which is why comprehensive translation of the ISO/CCITT DN to an INDEX appears impractical.

The result of this approach is that Distinguished Names are not translated at all. Similar functionality (naming of objects) is instead provided by the ClassInstancePointer's that identify table entries corresponding to managed objects.

Each class or side table has within it at most one TableIndex, a non-accessible indexing object of type INTEGER. This value may have no significance other than to discriminate between entries within that table. While a class table has exactly one index (its own TableIndex), a side table shares the index of its class table and any superior side tables, qualifying these as necessary with its own indexing object. Assignment of TableIndexes is discussed in subsection 2.14 (Translation of Create Operations).

The class of a ClassInstancePointer is implicit in the prefix corresponding to the class table. A single mapped ClassInstancePointer conveys the information of both.

2.2.6.4 Relationship Management

The ClassInstancePointer is the basis for all translation of relationship management, including containment. For the ISO/CCITT Structure of Management Information (SMI), object naming documents the containment relationships. A subset of such functionality is mapped via the Parent pointer, which allows the immediately superior object to be found, its superiors, and so forth.

The optional children object group (see Annex B) provides a subset of scoping functionality. That facilitates discovery of the entries immediately subordinate to a class entry.

A ClassInstancePointer may reference a native SNMP object (one that was not translated). That allows such objects to be incorporated in generic network models which may span many systems, allowing common notifications, logging, trouble exchange, etc.

A native object is ordinarily treated as an immediate subordinate of the internet system (as regards containment relations). The exception is a table sharing one or more indexes with a superior table. In that case, the subordinate table is consider to be contained within the superior one when interpreting (OSI) containment relationships.

2.2.6.5 Attributes of top

For the following reasons, most attributes of [7] top are not directly translated.

- Most of these notions would be unfamiliar to the Internet user. Thus their presence would add questionable value.
- Multi-valued attributes would require additional side tables for all object class groups, which would be cumbersome.
- Managed object class is implicit in the class prefix and thus does not require a special attribute.
- An allomorphs attribute is supported in ISO/CCITT to allow dynamic recognition of allomorphs which are supported by an instance. Since concise MIB notation does not support allomorphs, there is no reason to list them.

- Presence or absence of conditional PACKAGEs can be detected by using a Get-Next Request, then determining whether the conceptual rows returned are as expected. No special attribute is needed.

In ISO/CCITT systems management, ambiguity may (in rare cases) arise about what NAME BINDING to use when creating an object. For example, two alternative NAME BINDINGs might associate the same subordinate and superior classes, but with different semantics. CMIS addresses this by allowing the NAME BINDING identifier to be specified using M-CREATE. The omibtransSptNameBindingTable (Annex B) provides for this contingency. Since this is an unusual situation, it is handled by an auxiliary table rather than by defining a special object for every class group.

2.2.6.6 ATTRIBUTE GROUPs

This construct provides an abbreviated form whereby CMIP can refer to a grouping of ATTRIBUTEs collectively. That does not map to the SNMP. Translated objects corresponding to ATTRIBUTEs must be fetched explicitly (individually).

2.3 MAPPING TO THE SYNTAX CLAUSE

Syntactical mappings for simple (scalar) and complex (multi-valued) syntax are discussed separately below, as is the special procedure for translating ANY DEFINED BY syntax. The procedures for complex syntax also addresses the syntax and semantics of optionality.

2.3.1 Procedure for Simple Input Syntax

2.3.1.1 Mapping

Following are the rules for translating non-constructed (scalar) syntax.

Always assign a translated SNMPv1 object the same (actual) syntax as for SNMPv2 usage. SNMPv2 usage includes standard SNMPv2 types [16] and textual conventions [17]. It also includes additional textual conventions specified in Annex B.3. The following table summarizes the textual conventions and SNMPv2 types that are anticipated by SNMPv1 translations. Any textual conventions used shall be imported into the translated module.

ISO/CCITT Construct	SNMPv2	SNMPv1
BIT STRING	BitString Convention	BitString Convention
BOOLEAN type	TruthValue Convention	TruthValue Convention
CHOICE of INTEGER or OBJECT IDENTIFIER	ChoiceOfIntOrOID Convention	ChoiceOfIntOrOID Convention
counter ATTRIBUTE	Counter32 & Counter64 Types	INTEGER
ENUMERATED type	INTEGER with named values	INTEGER with named values
gauge ATTRIBUTE	Gauge32 Type	INTEGER
GeneralizedTime & UTCTime	DateAndTime Convention	DateAndTime Convention
GraphicString	GraphicString convention	GraphicString convention
IA5String, NumericString, PrintableString, VisibleString	DisplayString Convention	DisplayString Convention
INTEGER Type	INTEGER with typing values, else Integer32 or UInteger32	INTEGER
ObjectInstance or DN	ClassInstancePointer Convention	ClassInstancePointer Convention
REAL; or choice of REAL or INTEGER	Integer32 or UInteger32	INTEGER

Table 2. Simple input syntax mapping.

For SNMPv2, use Counter64 only if Counter32 cannot accommodate all permissible values. Use UInteger32 where negative choices are not permitted; otherwise use Integer32. For SNMPv1, INTEGER must always be used instead of Counter64/32, Integer 32, or UInteger32 types.

When mapping an OSI time construct having less granularity to the DateAndTime convention, pad the unmatched values (least significant) of the convention with zero.

Map a REAL to an INTEGER by selecting an appropriate unit of measurement, i.e., an INTEGER raised to a power of ten. For SNMPv1, document the scaling factor in the DESCRIPTION clause; for SNMPv2, provide this information in the UNITS clause. Map a CHOICE of REAL or INTEGER to INTEGER, simplifying the syntax.

Also simplify the syntax by mapping CHOICE of INTEGER or OBJECT IDENTIFIER to the ChoiceOfIntOrOID textual convention. That convention is specified in Annex B.3.

Transform ENUMERATED types to named INTEGER types, mapping the value (0) to (32767). Document in the DESCRIPTION clause that only the named values may be used.

Treat a constructed type that contains no more than one scalar (e.g., various forms of string specialization) as if it were the contained scalar.

For ASN.1 string types that are subsets of ASCII (PrintableString, NumericString, IA5String, VisibleString, etc.) use DisplayString if at all possible (which does imply a length restriction). Implementations shall comply with the NVT ASCII subset where practical. Map other string constructs (e.g., GraphicString) to OCTET STRING.

As necessary, define TEXTUAL-CONVENTIONs for scalar attribute values that will be used in multiple contexts (e.g., appear in more than one context in a translated MIB module). Import and reference existing conventions whenever that is an option.

When defining new conventions, populate their DESCRIPTION clauses with the same text that would otherwise appear in the DESCRIPTION clause of an OBJECT-TYPE macro (discussed below). Package these in a separate "conventions module". Import such conventions into the translated SNMPv1 and SNMPv2 modules as appropriate. For a full translation, label the conventions module:

"MIB" || <naming-prefix> || "Conventions".

and register it as:
{ <output module subtree> 3 }.

For an incremental translation, the same module label and object identifier qualifier rules apply as described previously in Section 2.2.6.

2.3.1.2 Discussion

Internet SMI precludes a named INTEGER value of zero. Some compilers will not accept it. 32767 is a number that practically any machine architecture can support, large enough so it should not conflict with any enumerated actually specified. If collisions occur, human judgment must be used to revise named INTEGER values such that no collisions appear in the translated MIB.

Allowing optional conceptual columns within rows is not consistent with the philosophy of RFC 1212, but neither are the MIBs the procedure seeks to translate. However, optional columns can be accommodated by SNMP using the Get-Next request. In that case protocol returns inconsistent object ID prefixes, rather than an error condition, for any non-present objects.

These rules will work for most but not all cases. Human judgment must be applied where they do not, e.g., an OSI counter uses a modulus not supported by SNMP, a real value ranges so widely a single scaling factor can not be applied, etc.

2.3.2 General Procedure for Complex Input Syntax

2.3.2.1 Mapping

Map all complex syntax — that which can not be mapped to a single columnar entry — to side tables. Apply the following rules recursively as necessary to create the side tables.

- For a given level of nesting, when encountering any of the following — CHOICE, SEQUENCE (OF), or SET (OF) — create a new side table; and in the current table, provide a Complex or MultiComplex column that points to it. If the mapped syntax is singly occurring (CHOICE, SEQUENCE, SET), the side table pointer is of type Complex. If the mapped syntax is multiply occurring (SEQUENCE OF, SET OF), the side table pointer is of type MultiComplex. Given the recursive enumeration process, a side table may itself have side tables.
- Having decomposed complex syntax into scalar elements, apply the same rules as for simple syntax in determining the value for each SYNTAX clause.

- If the syntax mapped by a side table is recursive — can directly or indirectly contain instances of itself — allocate a Recursion pointer and a RecursivelyNested flag before the RowStatus flag. This RecursionPointer when populated points to the next lower level of nested syntax (which may be in another table). The RecursivelyNested flag indicates whether a given row is pointed to by a recursion pointer, or is at the outermost level of nesting.

The following apply for all forms of optional syntax, whether scalar or complex.

- Enumerate objects for all possible values, including optional values for CHOICEs, SETs and SEQUENCES.
- Optional syntax may be nested, e.g., a CHOICE may be OPTIONAL. When that is the case, the rules of presence apply cumulatively, with outermost levels taking precedence over contained levels.

Semantics of optionality differ depending on the syntax translated.

- For CHOICE types, the agent enforces that exactly one option will be populated at any time (at most one if other optionality applies). Successfully setting new column(s) corresponding to the value of a CHOICE automatically causes pre-existing values to be destroyed.
- For SET OF CHOICE types, multiple conceptual rows may be mapped to. However, exactly one choice is populated per row.
- For OPTIONAL types that are scalar (including optional AnyDefinedBy pointers) the omibtransSptDestroyObjectTable shall be implemented. That allows previously populated values that are no longer wanted to be deleted by a management station without destroying and recreating the containing rows.
- Destroy complex types as needed using RowStatus (discussed further below).
- The agent enforces that otherwise mandatory objects within a conditional PACKAGE may be populated only at the time the class instance is created.
- The agent is generally responsible for enforcing consistency with ISO/CCITT usage regarding presence or absence of translated objects.

2.3.2.2 Discussion

This is the messiest part of the translation process, but cannot be avoided while translating complex GDMO MIBs. One way of viewing side tables is that constructed types are put in "third normal form," i.e., broken up into a set of flat tables each of which has a unique key. That is true for all tables except side tables of Complex type, which are extensions of their superior tables and have exactly the same indexes.

The SNMP does not support SET (OF) constructs. Those map to the SEQUENCEs (OF) side table entries, as do all forms of complex syntax.

The Complex and MultiComplex types are formalisms whose invariant values are determined when the MIB is translated. Their purpose is to document that the actual data is elsewhere, i.e., in the side tables named. Since these types are invariant there is no need ever to fetch their values over a network. As discussed below (Mapping of the INDEX Clause) explicit pointers are not required to locate parent entries given side table entries; or vice versa. Such relationships are implicit in the INDEX structure.

2.3.3 Procedures for ANY DEFINED BY Syntax

Translate ANY DEFINED BY syntax using one of the three procedures set forth below. The procedure depends whether:

- The syntax resolves to exactly one choice,
- The syntax references ATTRIBUTE values, or
- Neither of the above cases apply.

In the simplest case, the ANY DEFINED BY value resolves to a specific syntax for a particular object class. For example, this may happen when that syntax is resolved due to refinement of a superior class. In this case, treat the ANY DEFINED BY construct as if a direct reference to the resulting syntax. Document that in the translator's Proforma (see 6.2.2).

If the ANY DEFINED BY construct references an ATTRIBUTE value, map the construct to an object that observes the AnyDefinedBy textual convention. Such a pointer (when populated) points to an entry in a side table holding syntax for the appropriate ATTRIBUTE. The side table will typically be in a different class group from the table holding the AnyDefinedBy pointer. A typical example is the log record that points to a historical value of an ATTRIBUTE.

For instance, a StaChaDefOldAttributeValue AnyDefinedBy pointer might point to a columnar entry of a system class table that represents AdministrativeState. In this case, only the TableIndex, AdministrativeState, and RowStatus columns would be populated in the row that is pointed to. This is because the entry is historical and not associated with any current class entry. Likewise, the entry's TableIndex does not correspond to any current entry. To avoid ambiguity between current and historical information, the Parent pointer of a historical ATTRIBUTE entry in a class table is set to zero (0).

If the ATTRIBUTE value referenced by AnyDefinedBy pointer is in a side table, then the entry pointed to in the side table has no corresponding entry in the parent table. This allows differentiation of historical and current ATTRIBUTE values in the table. Note that an AnyDefinedBy pointer always references a side entry directly, rather than pointing to an object of type Complex or Multi-Complex.

Syntax referenced by an AnyDefinedBy pointer is regarded as an extension of the class group holding the pointer. Thus, the context and enforcement of consistency rules are both determined by the position of the AnyDefinedBy pointer in its class group.

Once a class entry is created, the value of an AnyDefinedBy pointer within it may only be deleted or changed if the new syntax (or absence thereof) is valid. In such cases, the agent automatically destroys any syntax the pointer previously pointed to, deallocating associated row indexes as well.

As always, it is the responsibility of the agent to enforce semantics of the managed object. The manager need not be aware of consistency rules except in cases where it is allowed to populate ANY DEFINED BY constructs.

Occasionally, ANY DEFINED BY syntax neither resolves to a single value nor corresponds to an ATTRIBUTE value. In that case, enumerate the syntax as an ANY DEFINED BY pointer pointing to an "ANY table".

An ANY table is very similar to a side table. It has identical structure except it has one additional indexing object — the InstancePointer — that precedes the TableIndex object. All its other

objects have the same syntax as an equivalent side table. However, since it may be referenced by multiple class groups, an ANY table has different registration, INDEX arguments and labeling conventions from a side table.

The InstancePointer points to the superior table entry holding the associated AnyDefinedBy pointer, or, the superior ANY table entry. The InstancePointer and TableIndex (in that order) provide the two parameters for the INDEX clause.

An ANY table belongs to its own OBJECT-GROUP ("ANY group") rather than a class group. This group contains the ANY table to be referenced by AnyDefinedBy pointers, and possibly, additional ANY tables subordinate to that table (analogous to side tables).

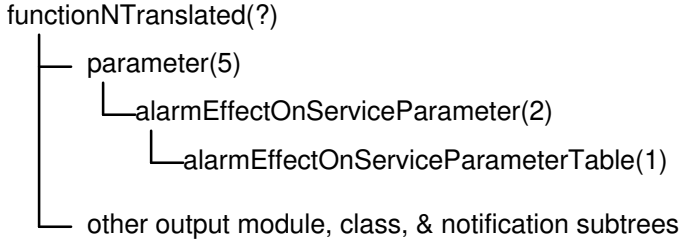
To register ANY tables, assign an output "discriminator subtree" for each input registry family of OBJECT IDENTIFIERs that discriminates ANY DEFINED BY syntax. Reserve an arc in the output tree for each OBJECT IDENTIFIER of the input family, keeping the same relative arc number. The resulting arcs are the ANY group arcs.

Beneath an ANY group arc, assign arc 1 to the (highest level) ANY table holding syntax that corresponds to the associated OBJECT IDENTIFIER in the input registry family. If the ANY table has subordinate tables, assign those sequential arcs beneath the ANY group arc beginning with arc two.

For example, the draft Software Management Function (ISO/IEC JTC1/SC21 N7949) defines an alarmEffectOnServiceParameter to be included in the ManagementExtension parameter of the AdditionalInformation parameter of the AlarmInfo field in an alarmRecord. In this case, the input OID that discriminates the ManagementExtension ANY DEFINED BY is:

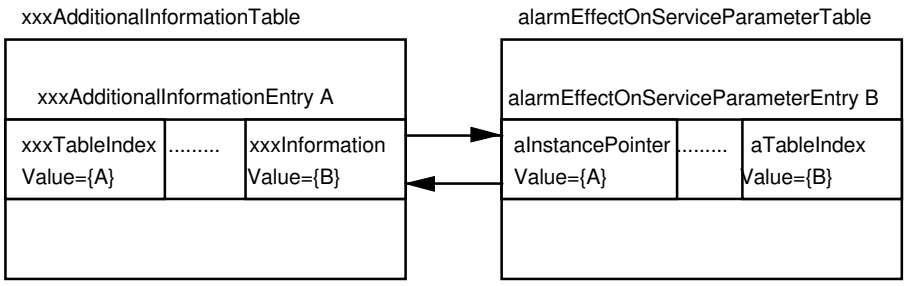
{ joint-iso-ccitt ms(9) function(2) partN(?) parameter(5) 2}

Thus, in a translation of the Software Management Function (functionN), an output "discriminator" subtree would be allocated to this OID:



and the associated syntax AlarmEffectOnServiceParameter (BOOLEAN) would be expanded as a column of the alarmEffectOnServiceParameterTable.

Each row of the AdditionalInformation side table holds an AnyDefinedBy pointer to a ManagementExtension. Suppose one AdditionalInformation side table row has the name (object identifier) A, and its AnyDefinedBy pointer refers to row B of the alarmEffectOnServiceParameterTable. In this case, the InstancePointer variable of row B must contain the object identifier value A (pointing back to the superior entry). The index for row B uniquely distinguishes row B from all other entries in the alarmEffectOnServiceParameterTable, whether associated with the AdditionalInformation table or any other superior table.



The TableIndex variable in row B, like every TableIndex, discriminates this entry in the table from other entries in the same table. The TableIndex of an ANY table also enables creation of an entry by a management station and discrimination between multiple entries in a subordinate ANY table corresponding to a common superior entry. Neither of these capabilities apply to the AlarmEffectOnServiceParameter example. However, all ANY tables have a TableIndex for consistency.

2.4 GENERATION OF INTERNET MIB LABELS

The previous subsection completed the procedure for enumerating OBJECT-TYPE macros. This subsection explains how to label them.

2.4.1 Notation

The following variables are used to generate labels.

<ASN.1 type reference>	Label of a field within an Abstract Syntax Notation One (ASN.1) production, e.g., "additionalText".
<ATTRIBUTE id>	Label of an ATTRIBUTE template, e.g., "probableCause".
<MANAGED OBJECT CLASS id>	Label of a MANAGED OBJECT CLASS template, e.g., "alarmRecord".
<naming prefix>	The string literal assigned to the ASN.1 module to be generated and associated labels, e.g., "smi2."

Table 3. Variable label generation.

The notation “<ASN.1 type reference>*” means the concatenation of one or more <ASN.1 type reference>s together. This form is used for naming side tables, with one <ASN.1 type reference> concatenated for each level of side table subordination. A side table immediately subordinate to the class table will have one <ASN.1 type reference> concatenated in its label; a table immediately subordinate to that one will have two <ASN.1 type reference>s concatenated; and so forth.

2.4.2 Abbreviation Procedure

Concatenation of constituent labels can result in lengthy mappings for resulting labels that require abbreviation.

Abbreviate a label as follows.

- Take up to the first three characters of the label.
- Find the first special (upper case or numeric) character subsequent to those three, if any.
- Append that special character and up to two subsequent characters to the initial characters of the output label.
- In either case, "up to" means unless the end of the (text comprising the) label is reached.

For a <class-label>, spell it out in full to arrive at the concatenated names of the class table, class entry and entry syntax. Use the abbreviated form to create names of individual columns in the class table and subsequent tables.

For an <attribute-label>, spell in out it full to arrive at the concatenated names individual columns in the class table. Use the abbreviated form for subsequent tables of the class group.

For an <ASN.1 type reference>, spell in out it full to arrive at the name of the corresponding column in the highest level table in which it is referenced. Use abbreviated forms for subsequent tables.

2.4.3 Mapping Procedure

Considering the above abbreviation rules, use the following mapping rules to arrive at labels for translated OBJECT-TYPE macros.

Label	Syntax and Example
class table	<naming prefix> || <MANAGED OBJECT CLASS id> || "Table" e.g., smi2AlarmRecordTable
class entry	<naming prefix> || <MANAGED OBJECT CLASS id> || "TableEntry" e.g., smi2AlarmRecordTableEntry
class entry TableIndex	<naming prefix> || <MANAGED OBJECT CLASS id> || "TableIndex" e.g., smi2AlaRecTableIndex
class entry column for attribute value	<naming prefix> || <MANAGED OBJECT CLASS id> || <ATTRIBUTE id> || <ASN.1 type reference> e.g., smi2AlaRecLoggingTime
class entry parent	<naming prefix> || <MANAGED OBJECT CLASS id> || "Parent" e.g., smi2AlaRecParent
class entry RowStatus	<naming prefix> || <MANAGED OBJECT CLASS id> || "RowStatus" e.g., smi2AlaRecRowStatus

Table 4. Class table labels.

Label	Syntax and Example
side table	<naming prefix> || <MANAGED OBJECT CLASS id> || <ATTRIBUTE id>|| <ASN.1 type reference>* || "Table" e.g., smi2AlaRecCorrelatedNotificationsTable
side entry	<naming prefix> ||<MANAGED OBJECT CLASS id> ||<ATTRIBUTE id> || <ASN.1 type reference>* || "TableEntry" e.g., smi2AlaRecCorrelatedNotificationsEntry
side entry TableIndex	<naming prefix> || <MANAGED OBJECT CLASS id> || <ATTRIBUTE id> || <ASN.1 type reference>* || "TableIndex" e.g., smi2AlaRecCorrelatedNotificationsTableIndex
side entry column for attribute value	<naming prefix> || <MANAGED OBJECT CLASS id> || <ATTRIBUTE id> || <ASN.1 type reference>* e.g., smi2AlaRecCorNotNotificationIdentifier
side entry recursion	<naming prefix> || <MANAGED OBJECT CLASS id> || <ATTRIBUTE id> || <ASN.1 type reference>* || "Recursion" e.g., smi2EveForDisDisConNotRecursion
side entry recursively nested flag	<naming prefix> || <MANAGED OBJECT CLASS id> || <ATTRIBUTE id> || <ASN.1 type reference>* || "RecursivelyNested" e.g., smi2EveForDisDisConNotRecursivelyNested
side entry RowStatus	<naming prefix> || <MANAGED OBJECT CLASS id> || <ATTRIBUTE id> || <ASN.1 type reference>* || "RowStatus" e.g., smi2AlaRecCorNotRowStatus

Table 5. Side table labels.

Make the label for the syntax of a table entry the same as that of the table entry, except, promote its first character to upper case. Therefore, for smi2AlarmRecordEntry the corresponding syntax is Smi2AlarmRecordEntry.

Assign labels to objects of an ANY table as follows:

Label	Syntax
ANY table	<naming prefix> || <ASN.1 type reference>* || "Table"
ANY entry	<naming prefix> || <ASN.1 type reference>* || "TableEntry"
ANY entry InstancePointer	<naming prefix> || <ASN.1 type reference>* || "InstancePointer"
ANY entry TableIndex	<naming prefix> || <ASN.1 type reference>* || "TableIndex"
ANY entry column for attributes' values	<naming prefix> || <ASN.1 type reference>*
ANY entry recursion	<naming prefix> || <ASN.1 type reference>* || "Recursion"
ANY entry recursively nested flag	<naming prefix> || <ASN.1 type reference>* || "RecursivelyNested"
ANY entry RowStatus	<naming prefix> || <ASN.1 type reference>* || "RowStatus"

Table 6. ANY table labels.

2.4.4 Discussion

In choosing the naming prefix, try to make it both mnemonic and unique. Often, the label of the OBJECT IDENTIFIER from the input specification provides an appropriate choice, e.g., "smi2".

This approach is somewhat verbose. It does facilitate mechanization, also, ambiguities and collisions should be rare. There is the additional advantage that generated labels can be used for C and C++ language program variables without further manipulation.

Separating constituent ids with hyphens would increase readability and decrease likelihood of ambiguity. Unfortunately, many Internet MIB compilers do not allow this. For this reason, hyphens must be removed from input labels in arriving at output labels.

Having dealt with labeling issues, this discussion returns to the mapping of OBJECT-TYPE clauses.

2.5 MAPPING TO THE ACCESS CLAUSE

This is straightforward. Note that ADD-REMOVE and GET-REPLACE map to "read-write", while GET maps to "read." REPLACE (without GET) maps to write-only. There is no direct mapping to SET-TO-DEFAULT, since SNMP requires values to be explicitly set for existing objects.

2.6 MAPPING TO THE STATUS CLAUSE

Always specify STATUS mandatory.

2.7 MAPPING TO THE DESCRIPTION CLAUSE

For all cases in this subsection, carry over text from associated ISO/CCITT constructs, stripped of enclosing <text-delimiter>s, making sure any embedded double quotation marks are sanitized (i.e., replaced with single-quotes or removed). The variables (encased in angle brackets — "<>") are all as defined for the input OSI constructs.

If an object observes a textual convention, do not restate the DESCRIPTION of that convention.

If an object corresponds to a translated MANAGED OBJECT CLASS, insert the following information in the class table's DESCRIPTION clause.

- State that "All objects of this object group are translated from the ISO/CCITT <class-label> MANAGED OBJECT CLASS according to the NMF IIMCOMIBTRANS specification and conform to behavioral requirements that it specifies."
- Carry over any BEHAVIOUR clauses of the class's mandatory PACKAGEs. Optionally, carry over the BEHAVIOUR clauses of mandatory PACKAGEs for superclasses from which the object is derived.
- State the translated notifications that may be emitted by the class group.

If an object corresponds to part of a translated conditional PACKAGE (i.e., corresponds to an ATTRIBUTE or ACTION), the following rules apply.

- If this is the first translated OBJECT-TYPE corresponding to a particular conditional PACKAGE, state "This object belongs to the <package-label> package, present if <condition-definition>" where <condition-definition> is inserted stripped of enclosing quotation marks.
- If this is a subsequent translated OBJECT-TYPE corresponding to the same conditional PACKAGE, insert the text "This object is also part of the <package-label> package."
- Optionally carry over text from the BEHAVIOUR clauses of conditional PACKAGEs. If that is done, do so only for the first object enumerated for the PACKAGE.

If an object represents an option of a CHOICE, insert text stating "This object corresponds to the <ASN.1 choice label> option of the <ASN.1 option label> choice. Only one such choice may be present in a table. Setting a new value for this choice implicitly deletes any pre-existing value.", where:

- The <ASN.1 option label> is the (innermost) field reference label of the input ASN.1 syntax corresponding to this particular object; and
- The <ASN.1 choice label> is the type reference label of the containing ASN.1 input syntax that specifies the CHOICE keyword.

If an object represents OPTIONAL syntax, state "This object corresponds to the <ASN.1 option label> option, which may or may not be present in a given table entry.", where <ASN.1 option label> is the (innermost) field reference label of the input ASN.1 syntax corresponding to this particular object.

A particular object may have multiple forms of optionality (conditional PACKAGE, CHOICE and/or OPTIONAL syntax) in which case the DESCRIPTION clause will have comments documenting each form.

Also provide in this clause any information that would be included in an SNMPv2 UNITS clause.

Insert additional text in this clause if the mechanically translated result is deemed hard to understand, or there is additional information from normative text that should be carried over.

2.8 MAPPING TO THE REFERENCE CLAUSE

The translation procedure captures information about OSI constructs associated with an OBJECT-TYPE by means of machine scannable fields within the REFERENCE clause. The primary purpose of this information is to allow interpretation of Management Object Conformance Statements (MOCS — see below).

Scannable text shall be present only in the REFERENCE clauses of macros that:

- are object groups;
- are class tables;
- occupy columns within class tables representing translated syntax;
- are in top-level ANY tables (not subordinate to other ANY tables) representing translated syntax; or
- correspond to field names referenced in Management Object Conformance Statements (MOCS — discussed below). In such cases, reference is made only at the highest level (least subordinate) object that represents the field.

The scannable text has the structure:

[REFMACRO <reference-macro-name> <reference-macro-value>]*
[MANAGED OBJECT CLASS <class-label> object-identifier]
[PACKAGE <package-label> [object-identifier]]
[ATTRIBUTE <attribute-label> object-identifier [SET-BY-CREATE]]
[ACTION <action-label> object-identifier [CONFIRMED]]
[FIELD field-name-label]
[REINSTANTIATES <translated object name> object-identifier]
[NOTES <free format text>]

The notation is adopted from GDMO [8] subsection 9.2, "Conventions used in template definitions". Hyphenated terms are as defined in GDMO [8] subsection 9.2, "Notational tools for managed object definition", and correspond to instances of those terms encountered in the input specification. The object-identifiers correspond to the labels that immediately precede them.

The presence or absence of these optional constructs within a particular scannable clause depends on the following:

- The MANAGED OBJECT CLASS clause is specified for the class table object but is not repeated for other objects within the class group.

- The REFMACRO construct may optionally be specified for the object group (see the OBJECT-GROUP macro specified in section 6.1). In this case the string literal <reference-macro-value> is logically substituted for any occurence of the <reference-macro-name> wherever it occurs in the scannable REFERENCE clauses of any template within the object group.

  The <reference-macro-name> must:
  - consist of a single alphanumeric identifier,

- begin with a lower case alpha character, and
- be unique from any other label appearing in the translated output.

The <reference-macro-value> must:

- not contain any of the keywords found in the scannable REFERENCE clause, and
- be delimited by the first keyword encountered following the <reference-macro-name>, or the REFERENCE clause delimiter (''), whichever occurs first.

For example, as illustrated in Annex A, the construct "REFMACRO smi2Doc Rec. X.721 | ISO/IEC 10165-2 : 1992" means that all occurrences of "smi2Doc" should be replaced by the expanded document reference shown above.

- The PACKAGE construct is present only for objects corresponding to syntax translated for PACKAGEs.

- The ACTION construct is present if the object maps to an ACTION (discussed elsewhere in this document). The construct CONFIRMED is present only if the ACTION is confirmed.

- The FIELD construct is present if the object maps to an ACTION or NOTIFICATION field. The field-name-label is the label in the ASN.1 syntax of the action information, event information, or reply information of the action or notification. In rare cases, there may be no field-name-label in the ASN.1 syntax. If this occurs, any available MOCS proforma for the ACTION or NOTIFICATION shall be consulted for the field-name-label. If no MOCS proforma is available during translation, one of the suggestions specified by ISO/IEC 10165-6, Clause H.8, "Action/notification field name labels", shall be applied to generate artificial field-name-labels.

- Object-identifier values shall be present when corresponding OSI constructs are registered. If this construct is absent, the corresponding label is understood to be within the scope of the GDMO specification of the associated MANAGED OBJECT CLASS.

  These must be ASN.1 OBJECT IDENTIFIER values that could validly be referenced in the containing MIB module. However, they may include macro identifiers, in which case substitution takes place before evaluating the result.

- The SET-BY-CREATE construct represents a semantic that does not completely map to SNMP. This construct is present if and only if the corresponding ATTRIBUTE has the GET property but the object's BEHAVIOR clause allows the ATTRIBUTE to be populated using the M-CREATE service. In this special case, the ACCESS clause shall specify read-write. However, such objects may only be set at time of row creation. Attempts to update these objects at other times shall result in protocol violation.

- OBJECT-TYPEs corresponding to ACTIONs (see Section 2.11) reflect that fact with the presence of the ACTION clause and associated PACKAGE and possibly FIELD clauses.

- The REINSTANTIATES construct, if present, references an existing translation of the syntax and semantics applying to the present object. This construct is used where a textual convention is inappropriate, i.e., when reusing Complex or MultiComplex syntax.

Never invoke the REINSTANTIATES construct to reference another object that also uses this construct. Instead, reference the object that was originally translated.

The <translated attribute name> names the column of the table in which the construct was previously translated. The <object-identifier> gives the registration of that column.

Presence of this clause means that syntax, semantics and exercise of human judgment of this translation are identical to those of the object referenced.

- The NOTES construct, if present, precedes free format text that provides additional information about the reference. The free format text, which must be separated by white space from the NOTES keyword, comprises the rest of the REFERENCE clause.

2.9 MAPPING TO THE INDEX CLAUSE

2.9.1 Procedure

For the class table, list its TableIndex as the one object in the INDEX clause. For a side table, list the TableIndex of the class table; followed by TableIndexes of any superior side tables in descending order; followed by the side table's own TableIndex (if any).

2.9.2 Discussion

The table index structure allows one to find subordinate tables' entries that are associated with a superior tables entry, and vice versa. For example, knowing the subidentifier of a class entry, one can use a Get-Next operation referencing the name of a side table appended with that subidentifier concatenated to find the first associated entry (if any) in that side table. Subsequent Get-Next operations will reveal subsequent entries. A similar approach will discover entries of side tables subordinate to other side tables.

Also, indices of superior tables (class tables included) are implicit in the indices of all subordinate tables. Thus there is no need for explicit pointers from class entries to side tables or vice versa. The Complex and MultiComplex types are provided for purposes of documentation only and are not necessary to find related rows within a class group. The AnyDefinedBy pointer is not necessary to find side tables either, but helps determine which side table to examine when there are many alternatives.

There may be a "natural key" for a particular multi-valued syntax, e.g., an address or name. In this case, an artificial index may be inappropriate. Human judgment must weigh whether there is in fact a "natural" key and whether the length of the associated subidentifier would be permissible for purposes of indexing. If the natural key is an integer, treat that as a refinement to TableIndex, documenting that refinement in the object's DESCRIPTION clause.

Note: It is not recommended that natural keys other than integers be used for the INDEX parameter of a class table, as that will result in very long subidentifiers and complicate allocation of indexes for new object creation. Human judgment can be used to supplement class entry indices with side tables that provide secondary indices that support access based on natural keys.

There is no need to actually access OBJECT-TYPES that correspond to table indices — one would have to know them first to read them, and they cannot be changed. Therefore, their ACCESS clauses specify not-accessible.

2.10 MAPPING TO THE DEFVAL CLAUSE

Use the DEFVAL clause to specify values according to the DEFAULT VALUE construct of the associated PACKAGE.

2.11 TRANSLATION OF ACTIONS

2.11.1 Procedure

Translate each action-label <a> encountered in a PACKAGE of a MANAGED OBJECT CLASS as if it were instead a sequence of two or more attribute-label's occurring at the end of an ATTRIBUTEs clause in the same PACKAGE. Do so even if such a clause does not actually exist. Where multiple ACTIONS are specified for a PACKAGE, evaluate them according to their order in the input text, i.e., as if there were several such multi-ATTRIBUTE sets appended to the ATTRIBUTES clause.

Treat each ACTION with action-label <a> as if it were a multi-ATTRIBUTE sequence, its members occurring in the following order with the following labels, ATTRIBUTE syntax and access properties.

Status	Label	Syntax	Access Property
Mandatory	<a> || "ActionTrigger"	ActionTrigger	GET-REPLACE
	<a> || "ActionStatus"	ActionStatus	GET
Conditional	<a> || "ActionInfoArg"	CMIP ActionInfoArg	GET-REPLACE
	<a> || "ActionReply"	CMIP ActionReply	GET

Table 7. Action syntax mapping.

Transcribe the BEHAVIOUR information (if any) to the DESCRIPTION clause. Populate the ACTION construct rather than the ATTRIBUTE construct in the scannable REFERENCE clause. Otherwise, treat the pseudo-ATTRIBUTEs as if actually listed in an ATTRIBUTES construct.

Enumerate the ActionTrigger and ActionStatus components for every mapped ACTION, observing the associated ActionTrigger and ActionStatus textual conventions (per Annex B.3). These jointly assure that a particular ACTION is serialized for a given class entry. Thus, a management station need not "poll" to determine when a (simulated) ACTION has completed. Also, two or more such stations can access a class entry in an orderly manner.

The values ActionStatus can assume are:

-2: inProgress.
-1: noProblemDetected.
Other non-negative: specificError.

If the ACTION being emulated is confirmed, the value noProblemDetected indicates successful completion of the n-1th operation, where the ActionTrigger is of current value n. If the ACTION

being emulated is unconfirmed, this value (noProblemDetected) indicates no emulated ACTION is in progress, or the status of the last action is not known.

Any other value (of ActionStatus) reflects the outcome emulating a confirmed ACTION for which a processingError has been detected and a single, non-negative status code returned. For other forms of processingError (non-integral or negative), human judgment is needed to map the outcome to SNMP.

The values of ActionReply (if present) reflect the outcome of the last action successfully emulated for the class entry.

Values associated with ActionInfoArg are only present if set by a management station. Prior to implementing a SetRequest-PDU for such objects, the agent tests whether intended values are logically consistent. If not, it returns error-status of badValue for SNMPv1 (what would be inconsistentValue for SNMPv2). Should such an attempted update fail, all objects related to ActionInfoArg shall retain prior values (if any).

2.11.2 Discussion

There is subtle interplay between the CMIP protocol, the GDMO ACTION and PARAMETERS templates, and various CMIS errors in construing ISO/CCITT management syntax. The above mapping is succinct but requires the translator to understand the mapping of GDMO templates to CMIP.

The SetRequest-PDU that triggers emulation of the action does not complete until emulation terminates. Meanwhile there is no need for the management station to try to poll for results. CMIS(P) errors that can arise with an action almost all map into the error-status fields returned for SetRequest-PDUs that set up and trigger the emulated action. No special status field is needed to carry these.

The exception is the processingError, which is of syntax ANY DEFINED BY. This (default) mapping deals with the typical case where the exception is reported using non-negative integral syntax. Other cases require human judgment.

2.12 TRANSLATION OF NOTIFICATIONS

2.12.1 Procedure

To translate a notification requires that there also be a corresponding (translated) log record class group. Furthermore, an instance of the log record must be created for each trap message emitted, since variables from the log record are used to populate the VarBind list of the trap. The log record class group may be defined in a pre-existing MIB module. Preferably, it should accompany the translated notification in the same MIB module. When packaging translated notifications and an associated log record group in the same module, insert the translated TRAP-TYPE macro(s) immediately preceding the class table.

For each input subtree of NOTIFICATIONs to be translated, assign a corresponding output notification subtree. In Annex A, the example input notification subtree is smi2Notification and the example output notification subtree is smi2TranslatedNotifications.

Create a TRAP-TYPE macro for each NOTIFICATION. Assign it the label

<naming prefix> || <notification-label>

where <notification-label> is the label for the input NOTIFICATION template. In the example, processingErrorAlarm maps to smi2ProcessingErrorAlarm.

Assign the ENTERPRISE clause the registry of the output notification subtree (e.g., smi2TranslatedNotifications).

List in the VARIABLES clause:

- The ManagedObjectInstance variable of the associated log record; and
- Other objects of that log record that:
  - correspond to ATTRIBUTEs of the NOTIFICATION that are mandatory according to the input specification;
  - can not be multiply occurring for an instance of the log record; and
  - are not Complex or MultiComplex types.

Specify the DESCRIPTION clause, as follows.

- Transcribe the text from the NOTIFICATION's BEHAVIOUR construct (if any), stripped of delimiters.
- Insert text stating "This notification carries information associated with a <log-record> log record. The VarBind list associated with this message may include the following additional information from that log record: <additional-information>." where
  - <log-record> is replaced by the label of the class table of the translated log record, e.g., smi2AlarmRecordTable.
  - <additional-information> is a comma-delimited list of all other accessible objects of the log record class group not already listed in the VARIABLES clause.

In the REFERENCE clause, insert scannable syntax of the form:

NOTIFICATION <notification-label> object-identifier
[FIELD <Notification field name label> MAPS TO
<log record variable> [, <log record variable>]*]*
[NOTES <free form text>]

where the following applies.

- The <notification-label> is the label of the associated NOTIFICATION template (e.g., processingErrorAlarm).
- The object-identifier documents the ASN.1 registry of that template (e.g., {smi2Notification 10}).
- The FIELD construct is specified once for each Notification field name label referenced in the corresponding MOCS. The one or more log record variables name the associated log record objects. The names of these variables must be known in the context of the surrounding MIB module, i.e., imported or defined therein.
- Additional explanatory text (<free form text>) may follow the NOTES keyword.

For the TRAP-TYPE macro value, assign the same relative arc number as that of the translated notification (e.g., arc 10 for processingErrorAlarm).

The log record referenced in the VarBind list may be either:

- Part of a set of translated logging objects, e.g., subordinate to a translated log object that is ultimately contained within root; or
- A "pseudo log record" that does not have a Parent object, and exists only for purposes of reporting notifications.

2.12.2 Discussion

The mapping of the notification to variables of a log record serves multiple purposes.

- It circumvents SNMPv1's message size constraint, since a management station can fetch the missing information if all does not fit in one message.
- It saves inventing additional objects to report values in traps; log records will often be needed anyhow.
- Since log records map to tables, that mapping allows multiply occurring or optional values to be reported. Those are not supported by the VARIABLES clause.

The agent populates the VarBind list first with the values listed in the VARIABLES clause; then with other mandatory variables; then optionally with other values. The mandatory variables are as specified in the Management Object Conformance Statement (MOCS — see section 6). When interpreting a MOCS, the relationship between translated objects and corresponding ATTRIBUTEs is determined by examining the scannable REFERENCE clauses for the columnar objects of the translated log record.

When translating NOTIFICATIONs for previously translated log records, assign a naming prefix the same as when translating MANAGED OBJECT CLASSes. Use the ASN.1 IMPORT statement to import the relevant columns of the log record.

The pseudo log record accommodates the native SNMP environment where ISO/CCITT logging constructs are not applicable.

The (actual or pseudo) log record reported should be persistent enough so a manager requiring further information can retrieve it through "trap directed polling." One minute might be a reasonable minimum for the log record retention period. Whatever policy is used for log record retention should be documented in the AGENT-CAPABILITIES statement (discussed below).

Use caution carrying over NOTIFICATIONs to Internet MIBs, since notifications are not popular with the Internet community. Consider the volume of messages that will be generated without full event forwarding discrimination. Also bear in mind that:

- SNMPv2 does provide some event discrimination capability with its Party MIB; and
- The optional omibtransSptTrapSuppressionTable specified in Annex B.1 provides a crude mechanism to selectively suppress traps, by trap type, for SNMPv1.

2.13 TRANSLATION OF DELETE OPERATIONS

To simplify mechanized translation, RowStatus is provided for all class tables (and side tables). This is simpler than trying to determine which classes support manager-initiated create and delete operations (specified on a per NAME BINDING, not per MANAGED OBJECT CLASS, basis).

Deletion is accomplished by setting RowStatus to "destroy". Successful deletion of a class instance automatically leads to deletion of associated side and child table rows. The operation succeeds or fails as a whole (e.g., if there are dependents and automatic deletion is not supported, then nothing is deleted). Instances of side table entries can be deleted if and only if that is allowed for the associated set valued ATTRIBUTEs or the syntax translated is OPTIONAL.

2.14 TRANSLATION OF CREATE OPERATIONS

2.14.1 Overview

The creation (and deletion) rules for conceptual rows created according to this procedure conform to a subset (refinement) of the SNMPv2 RowStatus textual convention. That subset enforces consistency with OSI containment relationships, services and valid object states of OSI management.

The RowStatus convention defines six possible status values (states) for a conceptual row: active, notInService, notReady, createAndGo, createAndWait and destroy. Three of these are visible to Get operations: active and notInService, which map to existing OSI managed objects, and notReady, which is a preparatory state during creation of a managed object. Statuses of createAndGo and createAndWait are transitory values used by Set operations, discussed further below.

The fundamental principle for this subset of RowStatus usage is that all values of an instance of a class entry remain in a state consistent with OSI usage so long as RowStatus is active or notInService. This consistency constraint applies subsequent to every Set operation, although the operation may partially succeed (best efforts). If a Set operation is attempted that in any way violates OSI usage (NAME BINDINGs, containment, conditional PACKAGEs, etc.) error-status other than noError is returned (see below for further discussion of error-status).

While OSI structural notions are translated to Internet MIB macro format where there is an analog, in many cases there is no equivalent, e.g., NAME BINDINGs and PACKAGEs. No attempt is made to define alternative notation in such cases other than to document the mapping in the scannable REFERENCE clause. The source GDMO specification is the ultimate authority determining valid state of translated objects, their attributes and relationships.

2.14.2 Status of notInService

Status of notInService is only supported if the corresponding OSI resource is governed by an administrativeState ATTRIBUTE that supports a value of "locked". The notInService status is functionally equivalent to locked, while active status equates to unlocked (or created without administrativeState). An instance of a class entry can have its administrativeState locked or

unlocked either by setting the administrativeState column directly or by switching RowStatus between active and notInService.

The transitory administrativeState of "shutting down" is reflected in a RowStatus of active until it completes, at which point RowStatus becomes notInService. In this case, a response to the setting the administrative state variable is not returned until the resource is shut down. The same is true if RowStatus is directly set.

2.14.3 Status of notReady

A conceptual row having RowStatus of notReady is not yet created with respect to OSI usage and consistency checking is not yet applied to columnar objects of that entry. It is impossible for an badValue error-status to arise when manipulating a row with this status, since there is no enforcement of integrity constraints.

However, once a row achieves an active or notInService status, it cannot return to notReady unless first destroyed and recreated.

2.14.4 Row Creation

A prerequisite to create an instance of a class entry is that the conceptual row's parent column points to an existing instance of a superior class entry for which there is a valid NAME BINDING. In the event there is ambiguity about which NAME BINDING to use, the row of the omibtransSptNameBindingTable corresponding to the row to be created must be set using the OBJECT IDENTIFIER of the NAME BINDING to be used. That must be accomplished using the same SetRequest operation that creates the row.

A class instance must be created (first set to active or notInService) in a single operation resulting in a state consistent with OSI usage. This is accomplished using a single Set operation without setup, or by staging information in notReady status and then creating the instance. Creation is ordinarily accomplished by setting class entry RowStatus to createAndGo, resulting in active status. For a class having administrativeState, this can alternatively be done by setting RowStatus to createAndWait, resulting in a new instance in locked state.

In either case, values for conditional PACKAGEs that need to be present must be populated when the instance is activated. Side table instances cannot "be created" before corresponding class entries are created, although they may be staged in notReady status.

The omibtransSptTableIndexTable (see Annex B) provides a convenient mechanism for agent to supply manager with an index to create a new conceptual row. The index returned is unique to the table in question and allows the agent (optionally) to implement index assignment policies (e.g., FIFO or small number) on a per table basis.

Implementation of that table is discretionary. Any valid option specified by the RowStatus textual convention may be used.

2.14.5 ANY DEFINED BY Syntax

The handling of an ATTRIBUTE value is similar to other handling of side tables. However, the entry to be created will hardly ever have a corresponding entry in its class table, since it is logically equivalent to an object (the ANY DEFINED by pointer) which is typically in another object group.

Therefore, allocate "phantom" index values for superior tables such as the class table — in addition to the TableIndex of the side table — when creating such entries. The phantom indexes are prepended to the table's own index to arrive at the subidentifier for the ATTRIBUTE syntax. The agent permits creation of such side table entries so long as their syntax is permissible in the context of the AnyDefinedBy object that points to them.

For ANY tables, assign a TableIndex for the tables using normal procedures. Concatenate that value following the InstancePointer of the superior entry to arrive at the complete index of a particular entry.

The agent enforces that syntax pointed by an AnyDefinedBy pointer is consistent with the rest of the class entry, the same as for a regular side table entry. Deleting an AnyDefinedBy object (where permitted) has the side effect of deleting the entry pointed to as well as subordinate entries, and de-allocating their indexes. Such deletion is not permitted unless the class entry will remain internally consistent. Deletion of an ANY table entry implicitly de-allocates not only the TableIndex for that entry but also the associated phantom indexes.

Attempting to delete an entry pointed to by an AnyDefinedBy pointer by setting RowStatus to destroy is the same as attempting deletion of the pointer itself. If the operation succeeds, the pointer is implicitly deleted.

2.14.6 Discussion

To support creation of class instances complicates matters considerably, since the manager and its users must understand the underlying OSI constructs. This may not always be appropriate for native SNMP implementations, in which case a subset can be implemented that does not support create/delete.

To create an object mapping into the ISO/CCITT world requires that its parent be known, hence the parent table. A child table is also provided to allow general navigation of the MIB tree. The name binding table is necessary under some circumstances to determine the OBJECT IDENTIFIER associated with each parent/child binding.

3. CONSTRAINTS ON SNMPV1 PROTOCOL USAGE

3.1 APPROACH

The following assumptions about use of SNMPv1 protocol are made to facilitate MIB translation.

- The management station shall expect that conditional attributes may not be present on a per conceptual row basis and shall act appropriately, e.g., use Get-Next Request to test for presence.
- The management station shall expect that access actually granted may be less than that stated in the ACCESS clause due to dynamic access controls.
- As discussed above, Set operations on RowStatus and associated objects are constrained to leave the object in a state consistent with OSI usage. Among other things, that means that multiple columns having consistency dependencies must jointly be updated in a single operation when the class entry has active or outOfService status.

Mapping of OSI functionality also requires refinements to use of error-status as documented as follows.

badValue	Customary usage, and may be returned due to a set request that is contrary to the translated object's PERMITTED VALUEs, or would violate integrity of the class entry overall. This is used where SNMPv2 would return inconsistentValue, wrongType, wrongLength, wrongEncoding or wrongValue.
genErr	Customary "none-of-the above" usage, e.g., is returned where SNMPv2 would return genErr, resourceUnavailable, commitFailed or undoFailed.
noError	No constraints.
noSuchName	Customary usage, and • is returned when attempt is made to access an object for which read access is denied due to access control. In that case, the object is "invisible." • is returned in cases where the SNMPv2 would return inconsistentName, noAccess, noCreation, authorizationError, notWritable, noSuchInstance or noSuchObject.
tooBig	No constraints.

Table 8. SNMPv1 error mapping.

3.2 DISCUSSION

Clarification of SNMPv1 error-status usage is necessary to map functionality not supported by that protocol. Thus an off-the-shelf manager should be able to interoperate with an agent that implements a translated MIB but usage of the PDUs will not be entirely conventional.

4. SNMPV2 TRANSLATION PROCEDURES

4.1 GENERAL APPROACH

RFC 1452 [19] provides succinct guidelines for converting (translating) SNMPv1 MIBs to SNMPv2. This section cites applicable rules from that RFC, also specifying additional procedures, as a concise means to express the difference between translated SNMPv1 and SNMPv2 MIBs.

However, this is merely an editorial convenience. In practice a translator is unlikely to translate from GDMO to SNMPv1 and then translate that output to SNMPv2. Indeed, translation to either SNMP framework requires deep knowledge of the original GDMO MIB. Translation is best conceived of as a single input, dual output process.

4.2 OBJECT DEFINITIONS

4.2.1 Selected Procedures from RFC 1452

The following rules, quoted from [19], are relevant to translation of SNMPv2 MIBs. (Rules which have been omitted are not relevant to translated MIBs.)

"... Conversion of a MIB module does not require the deprecation of the objects contained therein. Only if the semantics of an object truly changes should deprecation be performed.

(1) The IMPORTS statement must reference SNMPv2-SMI, instead of RFC1155-SMI and RFC-1212.

(2) The MODULE-IDENTITY macro must be invoked immediately after any IMPORTs or EXPORTs statement...

(4) For any object with an integer-valued SYNTAX clause, in which the corresponding INTEGER does not have a range restriction (i.e., the INTEGER has neither a defined set of named-number enumerations nor an assignment of lower- and upper-bounds on its value), the object must have the value of its SYNTAX clause changed to Integer32...

(6) For any object with a SYNTAX clause value of Counter, the object must have the value of its SYNTAX clause changed to Counter32.

(7) For any object with a SYNTAX clause value of Gauge, the object must have the value of its SYNTAX clause changed to Gauge32.

(8) For all objects, the ACCESS clause must be replaced by a MAX-ACCESS clause. The value of the MAX-ACCESS clause is the same as that of the ACCESS clause unless some other value makes "protocol sense" as the maximal level of access for the object...

(10) For all objects, if the value of the STATUS clause is "mandatory", the value must be replaced with "current".

(11) For all objects, if the value of the STATUS clause is "optional", the value must be replaced with "obsolete".

(12) For any object not containing a DESCRIPTION clause, the object must have a DESCRIPTION clause defined...

Other changes are desirable, but not necessary...

(2) For any object with a string-valued SYNTAX clause, in which the corresponding OCTET STRING does not have a size restriction (i.e., the OCTET STRING has no assignment of lower- and upper-bounds on its length), one might consider defining the bounds for the size of the object..."

4.2.2 Additional Procedures

Translated MIBs must also import objects from SNMPv2-TC and SNMPv2-CONF.
For a full translation, register the resulting ASN.1 module as:

"{ <output module subtree> 2 }"

and name it:

"MIB" || <naming prefix> || "SNMPv2", e.g., MIBsmi2SNMPv2

labeling the MODULE-IDENTITY macro:

"mib" || <naming-prefix> || "SNMPv2".

For a partial translation, register the resulting ASN.1 module as:

"{ <output module subtree> 2 <least translated arc number> }"

and name the resulting module:

"MIB" || <naming prefix> || "SNMPv2Increment" || <least translated arc number>

labeling the MODULE-IDENTITY macro:

"mib" || <naming-prefix> || "SNMPv2Increment" || <least translated arc number>"

where the least translated arc number is the lowest relative arc number corresponding to any object translated. If no object is translated, the <least translated arc number> must be assigned manually to avoid possible collision.

Most values for the MODULE-IDENTITY clauses must be populated manually. However, a mechanized translator should generate skeletal text sufficient to compile. The value of the LAST-UPDATED clause, the macro's Label and OBJECT-IDENTIFIER can be automatically generated. The value of that macro production equals the registry of the generated SNMPv2 ASN.1 module, i.e., { <output module subtree> 2 }.

Populate the UNITS clause where applicable. The text included in SNMPv1 DESCRIPTION clauses that documents such SNMPv2 constructs need not be carried over to SNMPv2 modules.

Generate an INTEGER type if there are named values (map enumerated values into named values) or if subtyping information is available, carrying over the values and subtypes. Otherwise, use Integer32.

Objects with read-write ACCESS for SNMPv1 shall have read-create MAX-ACCESS for SNMPv2.

In cases where entries of a side table are in one-to-one correspondence with those of its superior table (having no TableIndex of its own) use the AUGMENTS clause pointing to the superior table in lieu of an INDEX clause.

In the DESCRIPTION clause, use the term NOTIFICATION-TYPE rather than TRAP-TYPE.

4.2.3 Discussion

While at any point there may not be a known NAME BINDING that would allow a create operation, it is always possible that a future binding will so allow. Unless it is clearly impossible for a class entry ever to be instantiated by a create operation, specify read-create.

Information about UNITS can optionally be picked up in a manual post-processing phase, by examining BEHAVIOR clauses of mapped ATTRIBUTES and PACKAGEs.

Proprietary extensions can be accommodated by referencing a table of a class group using AUGMENTS clause to index an "extension table". This is preferable to translating a proprietary class that is allomorphic to the translated class, as that would result in a class group having entirely different registration for its objects. This also holds for SNMPv1, although the dependency can only be reflected in the indexing structure, since SNMPv1 does not support the AUGMENTS clause.

4.3 TRAP DEFINITIONS

4.3.1 RFC 1452 Procedure

The following rules, quoted from [19], are relevant to translation of SNMPv2 MIBs. (Rules which have been omitted are not relevant to translated MIBs.)

"If a MIB module is changed to conform to the SNMPv2 framework, then each occurrence of the TRAP-TYPE macro must be changed to a corresponding invocation of the NOTIFICATION-TYPE macro.

(1) The IMPORTS statement must not reference RFC-1215.

(2) The ENTERPRISES clause must be removed.

(3) The VARIABLES clause must be renamed to the OBJECTS clause.

(4) The STATUS clause must be added.
Procedure: STATUS is always current.

(5) The value of an invocation of the NOTIFICATION-TYPE macro is an OBJECT IDENTIFIER, not an INTEGER, and must be changed accordingly."

4.3.2 Additional Procedures

The NOTIFICATION-TYPE value is generated by concatenating the value that is provided for SNMPv1 by the ENTERPRISES clause with the INTEGER value assigned to that TRAP-TYPE. For example, a TRAP-TYPE with the value 10 and ENTERPRISES clause of smi2TranslatedNotifications corresponds to an SNMPv2 NOTIFICATION-TYPE of value {smi2TranslatedNotifications 10}.

5. CONSTRAINTS ON SNMPV2 PROTOCOL USAGE

5.1 PROCEDURE

As with SNMPv1, SNMPv2 Set operations on RowStatus and associated objects are constrained to leave the object in a state consistent with OSI usage. Among other things, that means that multiple columns having consistency dependencies must jointly be updated in a single operation when RowStatus has status of active or outOfService.

The following table reflects error-status usage for SNMPv2.

authorizationError	No constraints.
badValue	Does not arise.
commitFailed	No constraints. Note that class instance is still left in consistent state.
genErr	No constraints.
inconsistentName	No constraints.
inconsistentValue	Consistency is enforced across a class instance, and may apply to the presence or absence of row entries, as well as their values.
noAccess	No constraints.
noCreation	Will be returned if there is no NAME BINDING permitting row creation.
noError	No constraints, except that certain conditions are interpreted as errors that would not otherwise be.
noSuchName	Does not arise.
notWritable	No constraints.
readOnly	Does not arise.
resourceUnavailable	No constraints.
tooBig	No constraints.
undoFailed	Does not arise for translated MIBs. The class instance is always left in a consistent state.
wrongEncoding	No constraints.
wrongLength	No constraints.
wrongType	No constraints.
wrongValue	No constraints.

Table 9. SNMPv2 error mapping.

Attributes and objects for which read access is denied due to access control remain invisible. However, that is reflected by returning noSuchInstance in the VarBind rather than error-status of noSuchName.

5.2 DISCUSSION

The added capabilities of the SNMPv2 protocol allow most constraints required for translation to SNMPv1 to be relaxed. Existence of the authorizationError value avoids the contrived use of readOnly to signal access control violation. The codes for authorizationError and inconsistentValue provide straightforward solutions to the major problems encountered with SNMPv1 usage.

6. STATEMENTS OF COMPLIANCE AND CONFORMANCE

This section prescribes how SNMPv2 OBJECT-GROUP, MODULE-COMPLIANCE and AGENT-CAPABILITIES macros [18] shall be used in conjunction with MIB modules generated according to this specification. It also describes how the ISO/CCITT specified Management Object Conformance Statement (MOCS) [9] Proforma shall be used in conjunction with those SNMP macros.

Throughout this section, use of the term "translate" means application of procedures specified herein to create SNMPv1 or SNMPv2 output specifications given input GDMO specification(s).

6.1 MAPPING OF THE OBJECT-GROUP MACRO

Provide one OBJECT-GROUP macro for each MANAGED OBJECT CLASS mapped to SNMPv2. Assign this, the class group, the same ASN.1 label as that of the input MANAGED OBJECT CLASS. In the OBJECTS clause, list all accessible object-types enumerated for that class group. Identify STATUS as current. In the DESCRIPTION clause, name the translated MANAGED OBJECT CLASS, state that it is translated according to the procedures set forth in this specification, and observe that added detail can be found in the class table description. Populate the REFERENCE clause as specified in section 2.8.

Insert the OBJECT-GROUP macro for a class group so it appears immediately preceding the associated class table in the output MIB. If the class table corresponds to a log record, insert the corresponding TRAP-TYPE or NOTIFICATION-TYPE macro(s) following the OBJECT-GROUP macro but preceding the class table definition.

Insert an OBJECT-GROUP macro right before each ANY group. Assign this the same label as the corresponding ASN.1 type reference, except with its first letter demoted to lower case. In the OBJECTS clause, list all accessible object-types within the ANY group. Identify STATUS as current. In the DESCRIPTION clause, name the translated ASN.1 type reference and state that it is translated according to the procedures set forth in this specification. Populate the FIELD construct of the associated scannable REFERENCE clause with the ASN.1 type reference. Also, using the associated REGISTERED AS construct, provide the OBJECT IDENTIFIER that discriminates this particular ANY DEFINED BY option.

Include OBJECT-GROUP macros for SNMPv1 outputs the same as for SNMPv2, except expressed as a series of ASN.1 comments, i.e., prefix to each line of the macro a double hyphen ("--"). Do not insert any other comments between the lines corresponding to the OBJECT-GROUP macro.

Do not provide OBJECT-GROUP macros for stand-alone modules that contain only translated notifications. Document the dependency of such mapped notifications on associated log records using the MODULE-COMPLIANCE macro.

6.2 MAPPING OF THE MODULE-COMPLIANCE MACRO

6.2.1 General Procedure

For every input subtree translated to SNMPv1 and/or SNMPv2, provide one companion SNMPv2 information module containing a MODULE-COMPLIANCE macro.

For a full translation:

- register the Compliance ASN.1 module as:

  "{ <output module subtree> 0 0 }" e.g., { smi2TranslatedModules 0 0 }

  naming it:

  "MIB" || <naming prefix> || "Compliance", e.g., MIBsmi2Compliance;

- register the MODULE-IDENTITY macro as:

  "{ <output module subtree> 0 0 }" e.g., { smi2TranslatedModules 0 0 }

  naming it:

  "mib" || <naming-prefix> || "Compliance" e.g., mibsmi2Compliance; and

- register the MODULE-COMPLIANCE macro as:

  "{ <output module subtree> 0 1 }" e.g., { smi2TranslatedModules 0 1 }

  naming it:

  "mib" || <naming-prefix> || "ComplianceStatement"
  e.g., mibsmi2ComplianceStatement.

For a partial translation, append the qualifiers "Increment" and <least translated arc number> to all of the above, as previously described for module registration in Section 2 (e.g., the macro named mibsmi2ComplianceStatementIncrement1 is registered as { smi2TranslatedModules 0 1 1 }).

Populate the MODULE-IDENTITY macro with appropriate values for the LAST-UPDATED, ORGANIZATION, CONTACT-INFO and DESCRIPTION clauses.

In the MODULE-COMPLIANCE macro, specify one MODULE clause for the SNMPv1 translation, if provided, and one for the SNMPv2 translation, if provided. Do not specify MANDATORY-GROUPS. Specify the GROUPS clause only if a group is conditionally mandatory or if its usage is refined by SYNTAX, WRITE-SYNTAX or MIN-ACCESS clauses. If there is an existence dependency, cite normative text for the associated MANAGED OBJECT CLASS that describes the dependency. When referencing both SNMPv1 and SNMPv2 translated modules, list identical GROUPS (if any) for both MODULE clauses. Do not restate existence dependencies. Instead, assert that usage is identical for both SNMP frameworks.

A MODULE-COMPLIANCE macro referencing an SNMPv1 module is interpreted as if an SNMPv2 OBJECT-GROUP macro were defined for each class group according to the (Mapping Of The OBJECT-GROUP Macro) procedures above. Thus when both SNMPv1 and SNMPv2

modules are generated using the same inputs their GROUPS clauses (if present) reference the same labels and OBJECT IDENTIFIERs.

OBJECT, SYNTAX, WRITE-SYNTAX and/or MIN-ACCESS clauses may be specified. In such cases, justification must be provided in associated DESCRIPTION clauses. Such usage is restricted to refinements — rather than exceptions — to underlying constructs.

As expected, the OBJECT-GROUP is the unit of implementation. Contrary to SNMP custom, presence or absence of conceptual columns may vary from conceptual row to conceptual row depending on certain underlying OSI constructs (conditional PACKAGEs and the syntactical types ANY DEFINED BY, OPTIONAL and CHOICE).

Text for the DESCRIPTION clause shall consist of a completed Translation Compliance Proforma as specified below. The translator shall substitute appropriate text for embedded strings delimited by square brackets ("[]"). Any additional text provided in the value of this clause shall be external to the keywords BEGINPARSE and ENDPARSE.

This specification does not mandate the form in which the OBJECT IDENTIFIERs shared across common translated modules shall be specified. One approach is to specify a value reference for the output subtree in the compliance module, importing that value into the translated MIB module(s).

6.2.2 Translation Compliance Proforma

6.2.2.1 Proforma Overview

The DESCRIPTION clause of the MODULE-COMPLIANCE macro documents compliance of SNMP translated MANAGED OBJECT CLASSes to the translation procedures of this document. As such, it requires that the reader be familiar with terms and conventions of that document.

The clause has two purposes:

- To document inputs to the translation process; and
- To document exceptions to the translation procedures (if any) that have been made based on the judgment of the human translator. Enter "Not applicable." for those cases where human judgment was not required.

Exceptions to the translation procedures may be made by the translator in the following cases.

- If, in the translator's judgment, a literal application of the translation procedures would result in an output MIB specification too complex to be acceptable.
- If the stated translation procedures can not accommodate the range of permitted values required by the base specification (e.g., if mapping PrintableString to the alphabet and length restriction of DisplayString is not acceptable).
- If collisions (duplicates) in naming or value would result that would render the output specification ambiguous or invalid.

Exceptions may be made concerning syntax or semantics. However, every ACTION, ATTRIBUTE and NOTIFICATION associated with a translated MANAGED OBJECT CLASS must be mapped to corresponding construct(s) in the output MIB(s).

This clause shall document only deviations from (not refinements to) the normal outcome of the stated translation procedures. The other clauses of this macro accommodate such refinements. This clause shall document deviations in semantics as well as in syntax.

6.2.2.2 Proforma Text

BEGINPARSE
I. INPUTS TO THE TRANSLATION PROCESS

The input class subtree and output class subtree for this module are:
[Enter subtrees here as applicable.]

The associated input and output notification subtrees for this module are:

[Enter subtrees here as applicable.]

The associated input and output module subtrees for this module are:

[Enter subtrees here as applicable.]

The associated input and output ANY discriminator subtrees for this module are:
[Enter subtrees here as applicable.]

The document(s) providing input specifications for the translated module(s) are:
[Enter names and OBJECT IDENTIFIERS of those documents here.]

The <naming prefix> used to generate the translated module(s) is:
[Enter naming prefix here.]

II. EXCEPTIONS TO THE TRANSLATION PROCEDURES

Following are the cases where a 'natural key' is substituted for the default (arbitrary integer index):
[Name the affected tables and their natural keys here.]

Following are cases for which output syntax differs from the specified mapping for the input syntax:
[Name the (labels of the) input syntaxes here and describe the alternative syntaxes using ASN.1 notation. Express revised syntax in terms of alternative input syntaxes, i.e., as if different syntax had been specified for the input MIB.]

Example: ATTRIBUTE x is treated as if it had a syntax of DisplayString rather than a choice of NULL or PrintableString. Substitute an empty string where a CHOICE of NULL would otherwise be required.

Following are cases for which the translation procedure can not accommodate values permitted by the base specification.
[Name the (labels of the) input syntaxes here, and specify alternate syntaxes and their interpretation.]

Following are cases in which the translation procedure would result in duplicated labels:
[Name the labels that would result from the translation procedure and the alternative labels that are used.]

Human judgment has identified that the following ANY DEFINED BY constructs each resolve to a single option and are treated as if that syntactical option were referenced directly:

[Identify any such constructs here, supplying rationale.]

Following are cases for which the specified translation procedure would result in duplicated values (e.g., for named INTEGER types), and the alternative values used:

[Name the labels of those values; the values that would result from the normal translation procedure; and the alternative values to be used.

Following are other ways in which the translated MIB module does not comply with stated procedures.

[Document deviations and their justifications here.]

ENDPARSE

6.3 MAPPING OF THE AGENT-CAPABILITIES MACRO

The agent capabilities statement, supplied by the implementor, documents the conforming implementation — not the translated MIB module that is conformed to. A translator may supply a skeleton AGENT-CAPABILITIES macro to be completed by the implementor. When completed by the implementor, this statement includes the following information:

- SUPPORTS clause(s) shall be present stating all translated MIBs to which conformance is claimed.
- INCLUDES clause(s) shall be present, listing all MIB groups (and thus MANAGED OBJECT CLASSes) supported.

The statement may include the following information:

- The CREATION-REQUIRES, DEFVAL, VARIATION, SYNTAX, WRITE-SYNTAX and ACCESS clauses may be present so long as values specified are consistent with OSI usage.
- This (agent capabilities) statement may reference non-translated as well as translated MIBs. Compliance with this specification in no way constrains usage or lack thereof of AGENT-CAPABILITIES clauses for non-translated object groups.

6.4 MAPPING OF THE MANAGEMENT OBJECT CONFORMANCE STATEMENT (MOCS)

An OSI MOCS [9] is not translated. Rather, it is interpreted with respect to a translated MIB as follows:

- OSI constructs are treated as synonyms for corresponding generated OBJECT-TYPEs and TRAP-TYPEs or NOTIFICATIONs.
    - The class group label is the same as the managed object class template label, being the same thing for different protocols.

        - An attribute template label corresponds to all the OBJECT-TYPEs within the class group that name that label in scannable ATTRIBUTE constructs within their REFERENCE clauses.

        - An action template label corresponds to all OBJECT-TYPEs in the class group that name that label in scannable ACTION constructs within their REFERENCE clauses.

        - Action field name labels correspond to all OBJECT-TYPEs in the class group that name the label of action in question within their scannable REFERENCE clauses while providing a name equal to the action field name in the TYPES construct of the same clause.

        - A PARAMETERS label corresponds to all OBJECT-TYPEs in the class group that name that label in the scannable PARAMETER construct in their REFERENCE clauses.

- NAME BINDINGs are not translated, rather, are the same as for OSI specifications. The superior object class template label referenced in the pro forma is interpreted as the name of the superior class group to which the subordinate class group may be bound.

- An ATTRIBUTE is an atomic unit of implementation, i.e., if any OBJECT-TYPE corresponding to an ATTRIBUTE is implemented, than all shall be.

- The following columns of the MOCS do not pertain to translated MIBs, need not be filled out for purposes of conformance to translated MIBs, and shall be ignored with regard to SNMP interpretation if provided:

        - SetToDefault,
        - ATTRIBUTE GROUP, and
        - Any OBJECT (type) IDENTIFIER associated with a GDMO template label. These are only necessary in the rare case there is ambiguity (i.e., templates have identical labels). However, if present, these values shall reflect the input ISO/CCITT MIB, rather than any generated values.

7. COMPLIANCE AND CONFORMANCE

This section defines requirements which an implementation of these MIB translation procedures must satisfy in order to comply with this specification. Throughout the section, references to sections and subsections shall be understood to refer to parts of this specification. Likewise, use of the term "translate" shall be understood to mean application of procedures specified herein to create SNMPv1 or SNMPv2 output specifications based upon input GDMO specification(s).

7.1 COMPLIANCE

This subsection includes:

- Directives pertaining to all translated modules.
- Directives pertaining to just SNMPv1.
- Directives pertaining to just SNMPv2.

7.1.1 All Translated MIB Modules

Every translated MIB module shall include one OBJECT-GROUP macro per class table, and additional OBJECT-GROUP macros as needed, in compliance with subsection 6.1. This macro shall be in the form of ASN.1 comments for SNMPv1 modules and be in standard form for SNMPv2.

For each input module translated there shall be produced a single accompanying MODULE-COMPLIANCE module in accordance with subsection 6.2. This MODULE-COMPLIANCE module shall apply to both SNMPv1 and/or SNMPv2 outputs(s) that correspond to the input GDMO module.

Each translated module shall comply with the syntactical requirements of the applicable SNMP Structure of Management Information specification(s) and associated specifications [10, 12, 16].

7.1.2 SNMPv1 Translated MIB Modules

SNMPv1 translated modules claiming compliance to this specification shall:

(a) Be named and registered in accordance with subsection 2.2.

(b) Have output TRAP-TYPE macros corresponding to any input NOTIFICATIONs in compliance with subsection 2.12.

(c) Have OBJECT-TYPE macros:

- As are required due to translation of MANAGED OBJECT CLASS, ACTION or NOTIFICATION templates;
- That are labeled according to subsection 2.4;
- That carry over OSI semantics in their DESCRIPTION and REFERENCE clauses per subsections 2.7 and 2.8 (respectively); and

- That otherwise have their (SNMP macro) clauses populated according to Section 2.

7.1.3 SNMPv2 Translated MIB Modules

SNMPv2 translated modules claiming compliance to this specification shall comply with the requirements stated in sections 2, 3, 4 and 5. Where there is discrepancy, sections 4 and 5 take precedence over sections 2 and 3.

7.2 CONFORMANCE

This subsection defines requirements which an implementation of a translated MIB must satisfy in order to claim conformance to that MIB.

An implementation claiming conformance to translated MIBs or constituent objects shall be accompanied by an SNMPv2 AGENT-CAPABILITIES statement. Interpretation of that statement for SNMPv1 is the same as for SNMPv2 except any associated ACCESS clauses are understood to correspond to the associated SNMPv1 ACCESS (rather than MAX-ACCESS) clause, for which the read-create value shall not be specified.

The agent capabilities statement shall be accompanied by a completed Management Object Conformance Statement (MOCS), as defined by [9], for each translated class group.
An implementor may not claim conformance to the informative macros or types illustrated in Annex A.

Should there be inconsistencies between documents regarding conformance to OSI usage, documents shall be consulted in the following (descending) order of precedence to resolve them:

- Base standards,
- Input MIB specifications,
- Proforma specifier's MOCS,
- Output MIB specifications and MODULE-COMPLIANCE statements,
- Information supplier's MOCS, and
- Agent capabilities statements.

An implementor encountering any ambiguity in this specification regarding behavior of translated objects shall make the implementation as consistent with OSI usage as possible. Such action shall be documented in appropriate VARIATION clauses.

The MOCS proformas shall be accompanied by a Management Conformance Summary (MCS), as specified by [9], if the implementation has multi-stack support that includes CMIP. If the MCS is not provided, similar information should be provided in the overall DESCRIPTION clause of the agent capabilities statement.

An implementation must conform/comply with relevant components of IIMComibtransConventions, IIMComibtransSptSNMPv1, and/or IIMComibtransSptSNMPv2 modules, as follows.

- An implementation must comply with any textual conventions specified by IIMComibtransConventions and used by the translated MIB.

- An implementation must comply with all mandatory object groups specified by these modules.
- An implementation may choose to comply with any optional object groups specified by these modules.

If any of these statements apply, then the appropriate Annex B module(s) shall be referenced by the SUPPORTS clause of the agent capabilities macro, and one or more of these requirements shall be stated in the DESCRIPTION clause.

ANNEX A (INFORMATIVE): DEFINITION OF MANAGEMENT INFORMATION (DMI) EXAMPLE

A.1 INPUT ISO/CCITT MANAGEMENT INFORMATION BASE

```
-- ISO Definition of Management Information - DMI
-- REFERENCE "Rec. X.721 | ISO/IEC 10165-2 : 1992"

-- The following GMDO templates and ASN.1 modules have been
-- abbreviated to include only those definitions which are
-- needed for the example partial translation of DMI.
-- Specifically, only those templates and ASN.1 definitions
-- which relate to the alarmRecord and processingErrorAlarm
-- have been retained. The discriminatorConstruct attribute
-- has also been included.

-- Managed Object Classes

alarmRecord MANAGED OBJECT CLASS
DERIVED FROM eventLogRecord;
CHARACTERIZED BY
                -- The appropriate object identifier values for the
                -- eventType attribute, inherited from eventLogRecord
                -- managed object class are communicationAlarm,
                -- qualityofServiceAlarm, processingErrorAlarm,
                -- equipmentAlarm and environmentalAlarm
alarmRecordPackage    PACKAGE
                BEHAVIOUR alarmRecordBehaviour     BEHAVIOUR
                DEFINED AS  "This managed object is used to represent logged information that
                resulted from  alarm notifications or event reports";;
                ATTRIBUTES
                        probableCause           GET,
                        perceivedSeverity       GET;;;
CONDITIONAL PACKAGES
specificProblemsPackage PACKAGE
                ATTRIBUTES
                        specificProblems                GET;
                REGISTERED AS {smi2Package  1};
                PRESENT IF "the Specific problems parameter is present in the alarm notification or
                event report corresponding to the instance of alarm record",
backedUpStatusPackage PACKAGE
                ATTRIBUTES
                        backedUpStatus                  GET;
                REGISTERED AS {smi2Package  2};
                PRESENT IF "the backedUpStatus attribute has a value TRUE and the Backed up
                status parameter is present in the  alarm  notification or event report corresponding to
                the instance of alarm record",
backUpObjectPackage  PACKAGE
```

```
                ATTRIBUTES
                        backUpObject            GET;
                REGISTERED AS {smi2Package  3};
                PRESENT IF "Backup object parameter is present in the       alarm notification or
                event report corresponding to the instance of alarm record",
trendIndicationPackage PACKAGE
                ATTRIBUTES
                        trendIndication  GET;
                REGISTERED AS {smi2Package  4};
                PRESENT IF "the Trend indication parameter is present in the alarm notification or
                event report corresponding to the instance of alarm record",
thresholdInfoPackage PACKAGE
                ATTRIBUTES
                        thresholdInfo           GET;
                REGISTERED AS {smi2Package  5};
                PRESENT IF "the value for probableCause attribute is thresholdCrossed",
stateChangeDefinitionPackage PACKAGE
                ATTRIBUTES
                        stateChangeDefinition   GET;
                REGISTERED AS {smi2Package  6};
                PRESENT IF "there is a state transition for the states defined in State Management
                Function, corresponding to the alarm type specified in the alarm record",
monitoredAttributesPackage PACKAGE
                ATTRIBUTES
                        monitoredAttributes             GET;
                REGISTERED AS {smi2Package  7};
                PRESENT IF "the monitoredAttributes parameter is present in the alarm notification
                or event report corresponding to the instance of alarm record",
proposedRepairActionsPackage PACKAGE
                ATTRIBUTES
                        proposedRepairActions           GET;
                REGISTERED AS {smi2Package  8};
                PRESENT IF "the proposedRepairActions parameter is present in the alarm
                notification or event report corresponding to the instance of alarm record";
REGISTERED AS { smi2MObjectClass 1 };

eventLogRecord MANAGED OBJECT  CLASS
DERIVED FROM logRecord;
CHARACTERIZED BY eventLogRecordPackage PACKAGE
                BEHAVIOUR
                eventLogRecordBehaviour BEHAVIOUR
                DEFINED AS  "This managed object represents the information stored in the log as a
                result of receiving notifications or incoming event reports . ";;
                ATTRIBUTES
                        managedObjectClass      GET,
                        managedObjectInstance   GET,
                        eventType               GET;;;
CONDITIONAL PACKAGES
eventTimePackage PACKAGE
                ATTRIBUTES
                        eventTime               GET;
                REGISTERED AS {smi2Package  11};
                PRESENT IF "the event time parameter was present in the received event report",
notificationIdentifierPackage
```

```
                PRESENT IF "the notification Identifier parameter is present in the notification or
                event report corresponding to the instance of an event record or an instance of its
                subclasses",
correlatedNotificationsPackage
                PRESENT IF "the correlatedNotifications parameter is present in the notification or
                event report corresponding to the instance of an event record or an instance of its
                subclasses ",
additionalTextPackage
                PRESENT IF "the Additional text parameter is present in the notification or report
                corresponding to the instance of event record or an instance of its subclasses",
additionalInformationPackage
                PRESENT IF "the Additional information parameter is present in the notification or
                report corresponding to the instance of event record or an instance of its subclasses";
REGISTERED AS { smi2MObjectClass  5 };


logRecord MANAGED OBJECT CLASS
DERIVED FROM top;
CHARACTERIZED BY
logRecordPackage PACKAGE
                BEHAVIOUR logRecordBehaviour BEHAVIOUR
                DEFINED AS "This managed object represents the information stored in the logs";;
                ATTRIBUTES
                        logRecordId      GET,
                        loggingTime      GET;;;
REGISTERED AS { smi2MObjectClass 7 };


top MANAGED OBJECT CLASS
CHARACTERIZED BY
topPackage PACKAGE
                BEHAVIOUR topBehaviour BEHAVIOUR
                DEFINED AS "This is the top level of managed object  class hierarchy and every
                other managed object class is a specialization of either this generic class (top) or a
                specialization of subclass of top. The parameter miscellaneousError is to be used
                when a processing failure has occurred and the error condition encountered does not
                match any of object's defined specific error types.";;
                ATTRIBUTES
                        objectClass      GET,
                        nameBinding      GET ;;;
CONDITIONAL PACKAGES
packagesPackage PACKAGE
                ATTRIBUTES
                        packages         GET;
                REGISTERED AS {smi2Package   16};
                PRESENT IF "any registered package, other than this package has been
                instantiated",
allomorphicPackage PACKAGE
                ATTRIBUTES
                        allomorphs       GET;
                REGISTERED AS {smi2Package  17};
                PRESENT IF " if an object supports allomorphism" ;
REGISTERED AS { smi2MObjectClass 14};
```

-- Name Bindings

```
logRecord-log NAME BINDING
                SUBORDINATE OBJECT CLASS logRecord AND SUBCLASSES;
                NAMED BY
                SUPERIOR OBJECT CLASS log AND SUBCLASSES ;
                WITH ATTRIBUTE logRecordId;
                DELETE ONLY-IF-NO-CONTAINED-OBJECTS;
REGISTERED AS { smi2NameBinding 3 };
```

-- Packages

```
additionalInformationPackage PACKAGE
        ATTRIBUTES
                additionalInformation     GET;
REGISTERED AS {smi2Package  18};


additionalTextPackage PACKAGE
        ATTRIBUTES
                additionalText            GET;
REGISTERED AS {smi2Package  19};


correlatedNotificationsPackage PACKAGE
        ATTRIBUTES
                correlatedNotifications   GET;
REGISTERED AS {smi2Package  23};


notificationIdentifierPackage PACKAGE
        ATTRIBUTES
                notificationIdentifier    GET;
REGISTERED AS {smi2Package  24};
```

-- Attributes

```
allomorphs ATTRIBUTE
                WITH ATTRIBUTE SYNTAX Attribute-ASN1Module.Allomorphs;
                MATCHES FOR EQUALITY, SET-COMPARISON, SET-INTERSECTION;
REGISTERED AS       {smi2AttributeID  50};

logRecordId ATTRIBUTE
                WITH ATTRIBUTE SYNTAX Attribute-ASN1Module.LogRecordId;
                MATCHES FOR EQUALITY, ORDERING ;
REGISTERED  AS      { smi2AttributeID  3};

additionalInformation ATTRIBUTE
                WITH ATTRIBUTE SYNTAX Attribute-ASN1Module.AdditionalInformation;
                BEHAVIOUR additionalInformationBehaviour BEHAVIOUR
                DEFINED AS "This attribute is used to supply additional information in notifications.";;
REGISTERED AS { smi2AttributeID  6};

additionalText   ATTRIBUTE
```

```
                WITH ATTRIBUTE SYNTAX Attribute-ASN1Module.AdditionalText;
                MATCHES FOR EQUALITY, SUBSTRINGS;
                BEHAVIOUR additionalTextBehaviour BEHAVIOUR
                DEFINED AS "This attribute is used to specify additional textual information in
                notifications.";;
REGISTERED AS       { smi2AttributeID  7};

backedUpStatus ATTRIBUTE
                WITH ATTRIBUTE SYNTAX Attribute-ASN1Module.BackedUpStatus;
                MATCHES FOR EQUALITY ;
REGISTERED AS { smi2AttributeID  11};

correlatedNotifications ATTRIBUTE
                WITH ATTRIBUTE SYNTAX  Attribute-ASN1Module.CorrelatedNotifications;
                BEHAVIOUR correlatedNotificationsBehaviour BEHAVIOUR
                DEFINED AS "This attribute contains a set of notification identifiers and, if necessary,
                their associated managed object instance names, for the notifications that are
                considered to be correlated to this notification.";;
REGISTERED AS { smi2AttributeID  12};

eventTime       ATTRIBUTE
                WITH ATTRIBUTE SYNTAX Attribute-ASN1Module.EventTime;
                MATCHES FOR EQUALITY, ORDERING;
                BEHAVIOUR timeOrdering BEHAVIOUR
                DEFINED AS "The year, month, day, hour, minute and seconds field are compared in
                order to determine whether the specified value is greater or less than the value of the
                attribute.  The values for the year, month, day, hour, minute and seconds are
                determined from the character string representation and the year value is first
                compared. If equal the month value is compared and
                this process is continued if the compared fields are equal";;
REGISTERED AS { smi2AttributeID  13};

eventType       ATTRIBUTE
                WITH ATTRIBUTE SYNTAX Attribute-ASN1Module.EventTypeId;
                MATCHES FOR EQUALITY ;
REGISTERED AS { smi2AttributeID  14};

loggingTime ATTRIBUTE
                WITH ATTRIBUTE SYNTAX Attribute-ASN1Module.LoggingTime;
                MATCHES FOR EQUALITY, ORDERING;
REGISTERED AS { smi2AttributeID  59};

monitoredAttributes      ATTRIBUTE
                WITH ATTRIBUTE SYNTAX Attribute-ASN1Module.MonitoredAttributes;
REGISTERED AS { smi2AttributeID  15};

notificationIdentifier ATTRIBUTE
                WITH ATTRIBUTE SYNTAX Attribute-ASN1Module.NotificationIdentifier;
                MATCHES FOR EQUALITY;
                BEHAVIOUR notificationIdentifierBehaviour BEHAVIOUR
                DEFINED AS "This attribute contains a unique identifier for the notification, which
                may be present in the correlated notifications attribute of other notifications.";;
REGISTERED AS { smi2AttributeID  16};
```

```
perceivedSeverity ATTRIBUTE
                WITH ATTRIBUTE SYNTAX Attribute-ASN1Module.PerceivedSeverity;
                MATCHES FOR EQUALITY;
REGISTERED AS { smi2AttributeID  17};

probableCause ATTRIBUTE
                WITH ATTRIBUTE SYNTAX Attribute-ASN1Module.ProbableCause;
                MATCHES FOR EQUALITY ;
REGISTERED AS { smi2AttributeID  18};

proposedRepairActions ATTRIBUTE
                WITH ATTRIBUTE SYNTAX Attribute-ASN1Module.ProposedRepairActions;
                MATCHES FOR EQUALITY, SET-COMPARISON, SET-INTERSECTION;
REGISTERED AS { smi2AttributeID  19};

specificProblems ATTRIBUTE
                WITH ATTRIBUTE SYNTAX  Attribute-ASN1Module.SpecificProblems;
                MATCHES FOR EQUALITY, SET-COMPARISON, SET-INTERSECTION ;
REGISTERED AS { smi2AttributeID  27};

stateChangeDefinition ATTRIBUTE
                WITH ATTRIBUTE SYNTAX  Attribute-ASN1Module.AttributeValueChangeDefinition;
                BEHAVIOUR stateChangeDefinitionBehaviour BEHAVIOUR
                DEFINED AS "This attribute contains a set of state attribute identifiers and their old
                and new values.";;
REGISTERED AS { smi2AttributeID  28};

thresholdInfo ATTRIBUTE
                WITH ATTRIBUTE SYNTAX  Attribute-ASN1Module.ThresholdInfo;
REGISTERED AS { smi2AttributeID  29};

trendIndication ATTRIBUTE
                WITH ATTRIBUTE SYNTAX  Attribute-ASN1Module.TrendIndication;
                MATCHES FOR EQUALITY;
REGISTERED AS { smi2AttributeID  30};

backUpObject ATTRIBUTE
                WITH ATTRIBUTE SYNTAX  Attribute-ASN1Module.BackUpRelationshipObject;
                MATCHES FOR EQUALITY ;
REGISTERED AS { smi2AttributeID  40};

managedObjectClass ATTRIBUTE
                WITH ATTRIBUTE SYNTAX Attribute-ASN1Module.ObjectClass;
                MATCHES FOR EQUALITY;
REGISTERED AS { smi2AttributeID  60};

managedObjectInstance ATTRIBUTE
                WITH ATTRIBUTE SYNTAX Attribute-ASN1Module.ObjectInstance;
                MATCHES FOR EQUALITY;
REGISTERED AS { smi2AttributeID  61};

nameBinding ATTRIBUTE
                WITH ATTRIBUTE SYNTAX Attribute-ASN1Module.NameBinding;
                MATCHES FOR EQUALITY;
```

```
REGISTERED AS { smi2AttributeID  63};

objectClass ATTRIBUTE
                WITH ATTRIBUTE SYNTAX Attribute-ASN1Module.ObjectClass;
                MATCHES FOR EQUALITY;
REGISTERED AS { smi2AttributeID  65};

packages ATTRIBUTE
                WITH ATTRIBUTE SYNTAX Attribute-ASN1Module.Packages;
                MATCHES FOR EQUALITY, SET-COMPARISON, SET-INTERSECTION;
REGISTERED AS { smi2AttributeID  66};

-- The following attribute is included here only to illustrate choice and recursion in the output. The
-- managed objects which contain this attribute have not been included in this example input text.

discriminatorConstruct ATTRIBUTE
        WITH ATTRIBUTE SYNTAX Attribute-ASN1Module.DiscriminatorConstruct;
REGISTERED AS { smi2AttributeID  56};
```

-- Notifications

```
processingErrorAlarm NOTIFICATION
BEHAVIOUR processingErrorAlarmBehaviour BEHAVIOUR
                DEFINED AS "This notification type is used to report processing failure in a managed
                object.";;
                WITH INFORMATION SYNTAX Notification-ASN1Module.AlarmInfo
                AND ATTRIBUTE IDS
                        probableCause               probableCause,
                        specificProblems            specificProblems,
                        perceivedSeverity           perceivedSeverity,
                        backedUpStatus              backedUpStatus,
                        backUpObject                backUpObject,
                        trendIndication             trendIndication,
                        thresholdInfo               thresholdInfo,
                        notificationIdentifier      notificationIdentifier,
                        correlatedNotifications     correlatedNotifications,
                        stateChangeDefinition       stateChangeDefinition,
                        monitoredAttributes         monitoredAttributes,
                        proposedRepairActions       proposedRepairActions,
                        additionalText              additionalText,
                        additionalInformation       additionalInformation;
REGISTERED AS { smi2Notification  10};
```

-- ASN.1 Modules

```
ManagedObjectClassesDefinitions
{joint-iso-ccitt ms(9) smi(3) part2(2) asn1Module(2) 0} DEFINITIONS ::= BEGIN
--EXPORTS everything

smi2MObjectClass OBJECT IDENTIFIER ::=
{joint-iso-ccitt ms(9) smi(3) part2(2) managedObjectClass(3)}
smi2NameBinding OBJECT IDENTIFIER ::=
{joint-iso-ccitt ms(9) smi(3) part2(2) nameBinding(6)}
smi2Package OBJECT IDENTIFIER ::=
```

```
{joint-iso-ccitt ms(9) smi(3) part2(2) package(4)}

END -- Module ManagedObjectClassesDefinitions


Attribute-ASN1Module
{joint-iso-ccitt ms(9) smi(3) part2(2) asn1Module(2) 1}
DEFINITIONS IMPLICIT TAGS ::= BEGIN
--EXPORTS everything
IMPORTS

CMISFilter, Attribute, BaseManagedObjectId,
AttributeId, ObjectInstance, ObjectClass, EventTypeId
        FROM CMIP-1 {joint-iso-ccitt ms(9) cmip(1)
        version1 (1) protocol(3)}

DistinguishedName
        FROM InformationFramework {joint-iso-ccitt ds(5)
        modules(1) informationFramework(1)}

AE-title
        FROM ACSE-1 {joint-iso-ccitt association-control(2)
        abstract-syntax(1) apdus(0) version(1)};
        -- Note the syntax of AE-title to be used is from
        -- CCITT Rec. X.227| ISO 8650 corrigendum and not "ANY"

smi2AttributeID OBJECT IDENTIFIER ::=
{joint-iso-ccitt ms (9) smi(3) part2(2) attribute(7)}
smi2AttributeGroup  OBJECT IDENTIFIER ::=
{joint-iso-ccitt ms (9) smi(3) part2(2) attributeGroup(8)}

defaultDiscriminatorConstruct
DiscriminatorConstruct  ::= and : {}

-- The value assignments for the probable cause used in CCITT
-- Rec. X.733 | ISO/IEC 10164 -4  are specified below.
-- These are specified in accordance with the corrigendum to
-- ASN.1 ISO/IEC JTC1/SC21 N5901

arfProbableCause OBJECT IDENTIFIER  ::=
{ joint-iso-ccitt ms(9) smi(3) part2(2) standardSpecificExtension(0) arf(0)}

adapterError ProbableCause ::=
globalValue: { arfProbableCause 1 }
-- values 2-56 are also defined, omitted here for brevity
versionMismatch ProbableCause ::=
globalValue: { arfProbableCause  57 }

AdditionalText ::= GraphicString

AdditionalInformation ::= SET OF ManagementExtension

Allomorphs ::= SET OF ObjectClass
```

```
AttributeValueChangeDefinition::= SET OF SEQUENCE
{attributeID        AttributeId,
oldAttributeValue  [1] ANY DEFINED BY attributeID OPTIONAL, newAttributeValue  [2] ANY
DEFINED BY attributeID}

BackedUpStatus ::= BOOLEAN
-- True implies backed up
-- False implies not backed up

BackUpRelationshipObject ::= CHOICE
{objectName ObjectInstance, noObject NULL}

CorrelatedNotifications  ::=  SET OF SEQUENCE
{correlatedNotifications SET OF NotificationIdentifier,
sourceObjectInst ObjectInstance OPTIONAL}

DiscriminatorConstruct ::= CMISFilter
-- where CMISFilter is imported from CMIP as follows:
--
-- FilterItem ::= CHOICE {
--         equality        [0] IMPLICIT  Attribute,
--         substrings      [1] IMPLICIT SEQUENCE OF
--                 CHOICE {
--                 intialString  [0] IMPLICIT SEQUENCE {
--                         attributeId  AttributeId,
--                         string       ANY DEFINED BY attributeId },
--                 anyString     [1] IMPLICIT SEQUENCE {
--                         attributeId  AttributeId,
--                         string       ANY DEFINED BY attributeId },
--                 finalString   [2] IMPLICIT SEQUENCE {
--                         attributeId  AttributeId,
--                         string       ANY DEFINED BY attributeId } },
--         greaterOrEqual  [2] IMPLICIT Attribute,
--         lessOrEqual     [3] IMPLICIT Attribute,
--         present         [4] AttributeId,
--         subsetOf        [5] IMPLICIT Attribute,
--         supersetOf      [6] IMPLICIT Attribute,
--         nonNullSetIntersection  [7] IMPLICIT Attribute }
--
-- CMISFilter ::= CHOICE {
--         item    [8]  FilterItem,
--         and     [9]  IMPLICIT SET OF CMISFilter,
--         or      [10] IMPLICIT SET OF CMISFilter,
--         not     [11] CMISFilter }

EventTime ::= GeneralizedTime

LoggingTime ::= GeneralizedTime

LogRecordId ::= SimpleNameType(WITH COMPONENTS
{number PRESENT, string ABSENT})

ManagementExtension ::= SEQUENCE
{identifier         OBJECT IDENTIFIER,
```

```
significance      [1]  BOOLEAN  DEFAULT FALSE,
information       [2]  ANY DEFINED BY identifier}

MonitoredAttributes ::= SET OF Attribute

NameBinding ::= OBJECT IDENTIFIER

NotificationIdentifier ::= INTEGER
-- reuse only when no requirement exists
-- for correlating prior notification.

ObservedValue  ::= CHOICE
{integer INTEGER, real REAL }

Packages ::= SET OF OBJECT IDENTIFIER

PerceivedSeverity ::= ENUMERATED
{indeterminate (0),        -- used when it is not possible
                           -- to assign the following values
critical (1), major (2), minor (3), warning (4), cleared (5)}

ProbableCause  ::=  CHOICE
{globalValue     OBJECT IDENTIFIER,
localValue       INTEGER}

-- The values for the probable cause defined in CCITT Rec
-- X.733 | ISO/IEC 10164-4 are specified above.

ProposedRepairActions ::= SET OF SpecificIdentifier

SimpleNameType ::= CHOICE
{number          INTEGER,
string           GraphicString}

SpecificIdentifier ::= CHOICE { OBJECT IDENTIFIER, INTEGER}

SpecificProblems ::=  SET OF SpecificIdentifier

ThresholdInfo ::= SEQUENCE
{triggeredThreshold      AttributeId,
observedValue            ObservedValue,
thresholdLevel           [1] ThresholdLevelInd OPTIONAL,
-- Indication needed for multilevel thresholds
armTime                  [2] GeneralizedTime OPTIONAL}

ThresholdLevelInd ::= CHOICE
{up      [1]      SEQUENCE {  high     ObservedValue,
                              low      ObservedValue OPTIONAL
-- up is the only permitted choice for counter
},
down     [2]      SEQUENCE {  high     ObservedValue,
                              low      ObservedValue}}

TrendIndication ::= ENUMERATED
```

```
{lessSevere (0), noChange (1), moreSevere (2)}
END -- end of supporting productions


Notification-ASN1Module
{joint-iso-ccitt ms(9) smi(3) part2(2) asn1Module(2) 2}
DEFINITIONS IMPLICIT TAGS::= BEGIN
IMPORTS
ProbableCause,SpecificProblems, PerceivedSeverity, BackedUpStatus,  TrendIndication,
ThresholdInfo, NotificationIdentifier, CorrelatedNotifications,  MonitoredAttributes,
ProposedRepairActions, AdditionalText, AdditionalInformation, AttributeValueChangeDefinition FROM
        Attribute-ASN1Module

ObjectClass,AttributeId,ObjectInstance
        FROM CMIP-1 {joint-iso-ccitt ms(9) cmip(1)
        version1 (1) protocol(3)};

--EXPORTS everything

smi2Notification OBJECT IDENTIFIER ::=
{joint-iso-ccitt ms (9) smi(3) part2(2) notification(10)}

AlarmInfo ::=  SEQUENCE {
probableCause                   ProbableCause,
specificProblems        [1]     SpecificProblems OPTIONAL,
perceivedSeverity               PerceivedSeverity,
backedUpStatus                  BackedUpStatus OPTIONAL,
backUpObject            [2]     ObjectInstance OPTIONAL,
trendIndication         [3]     TrendIndication OPTIONAL,
thresholdInfo           [4]     ThresholdInfo OPTIONAL,
notificationIdentifier  [5]     NotificationIdentifier OPTIONAL,
correlatedNotifications [6]     CorrelatedNotifications  OPTIONAL,
stateChangeDefinition   [7]     AttributeValueChangeDefinition OPTIONAL,
monitoredAttributes     [8]     MonitoredAttributes OPTIONAL,
proposedRepairActions   [9]     ProposedRepairActions OPTIONAL,
additionalText                  AdditionalText OPTIONAL,
additionalInformation   [10]    AdditionalInformation OPTIONAL}
END -- Module smi2Notification
```

A.2 OUTPUT SNMPV1 MANAGEMENT INFORMATION BASE

```
-- ******************************************************
--                    Partial Translation of
--       CCITT Rec. X.721 | ISO/IEC 10165-2 : 1992
--             Definition of Management Information (DMI)
--                    into Internet SNMPv1 MIB
-- ******************************************************

MIBsmi2SNMPv1Increment1 DEFINITIONS ::= BEGIN
IMPORTS

smi2MObjectClass, smi2Package
```

```
        FROM ManagedObjectClassDefinitions
        {joint-iso-ccitt ms(9) smi(3) part2(2) asn1Module(2) 0}

smi2AttributeID
        FROM Attribute-ASN1Module
        {joint-iso-ccitt ms(9) smi(3) part2(2) asn1Module(2) 1}

smi2Notification
        FROM Notification-ASN1Module
        {joint-iso-ccitt ms(9) smi(3) part2(2) asn1Module(2) 2}

AnyDefinedBy, ChoiceOfIntOrOID, Complex, GraphicString,
Pointer, TableIndex, MultiComplex, Recursion,
RecursivelyNested, SourceIndicator, Parent
        FROM IIMComibtransConventions

Smi2SpecificIdentifier
        FROM MIBsmi2ConventionsIncrement1

InstancePointer, TimeStamp, RowStatus, TruthValue
        FROM SNMPv2-TC

iimcIIMCOMIBTRANS
        FROM IimcAssignedOIDs
        {iso(1) member-body(2) 124 forum(360501) iimcManual(15) iimcModule(0) 1}

smi2Translated, smi2TranslatedModules,
smi2TranslatedObjects, smi2TranslatedNotifications
        FROM MIBsmi2ComplianceIncrement1

OBJECT-TYPE
        FROM RFC1155-SMI

TRAP-TYPE
        FROM RFC-1215;

--alarmRecord OBJECT-GROUP
                --OBJECTS {
                --smi2AlaRecLoggingTime,
                --smi2AlaRecManagedObjectClass,
                --smi2AlaRecManagedObjectInstance,
                --smi2AlaRecEventType,
                --smi2AlaRecEventTime,
                --smi2AlaRecNotificationIdentifier,
                --smi2AlaRecCorrelatedNotifications,
                --smi2AlaRecAdditionalText,
                --smi2AlaRecAdditionalInformation,
                --smi2AlaRecProbableCause,
                --smi2AlaRecPerceivedSeverity,
                --smi2AlaRecSpecificProblems,
                --smi2AlaRecBackedUpStatus,
                --smi2AlaRecBackUpObject,
                --smi2AlaRecTrendIndication,
                --smi2AlaRecThresholdInfo,
```

```
                --smi2AlaRecStateChangeDefinition,
                --smi2AlaRecMonitoredAttributes,
                --smi2AlaRecProposedRepairActions,
                --smi2AlaRecParent,
                --smi2AlaRecRowStatus,
                --smi2AlaRecCorNotNotificationIdentifier,
                --smi2AlaRecCorNotSourceObject,
                --smi2AlaRecCorNotRowStatus,
                --smi2AlaRecAddInfSignificance,
                --smi2AlaRecAddInfInformation,
                --smi2AlaRecAddInfRowStatus,
                --smi2AlaRecSpeProSpecificIdentifier,
                --smi2AlaRecSpeProRowStatus,
                --smi2AlaRecThrInfTriggeredThreshold,
                --smi2AlaRecThrInfObservedValue,
                --smi2AlaRecThrInfThresholdLevelIndUpDown,
                --smi2AlaRecThrInfThresholdLevelIndHigh,
                --smi2AlaRecThrInfThresholdLevelIndLow,
                --smi2AlaRecThrInfArmTime,
                --smi2AlaRecThrInfRowStatus,
                --smi2AlaRecStaChaDefOldAttributeValue,
                --smi2AlaRecStaChaDefNewAttributeValue,
                --smi2AlaRecStaChaDefRowStatus,
                --smi2AlaRecMonAttAttribute,
                --smi2AlaRecMonAttRowStatus,
                --smi2AlaRecProRepActSpecificIdentifier,
                --smi2AlaRecProRepActRowStatus
                --}
                --STATUS mandatory
                --DESCRIPTION "This OBJECT-GROUP represents the
                --MANAGED OBJECT CLASS CCITT Rec. X.721|ISO/IEC
                --10165-2:1992:alarmRecord,  translated according to
                --the procedures specified by IIMCOMIBTRANS into
                --the OBJECT-TYPE smi2AlarmRecordTable. Refer to the
                --smi2AlarmRecordTable OBJECT-TYPE macro for
                --additional information."
                --REFERENCE "REFMACRO smi2Doc CCITT Rec. X.721|ISO/IEC 10165-2:1992"
--::= {smi2TranslatedObjects 1}

smi2ProcessingErrorAlarm TRAP-TYPE
        ENTERPRISE {smi2TranslatedNotifications}
        VARIABLES {
                smi2AlaRecManagedObjectInstance,
                smi2AlaRecEventTime,
                smi2AlaRecProbableCause,
                smi2AlaRecPerceivedSeverity
                }
                DESCRIPTION "This notification type is used to report processing failure in a
                managed object. This notification carries information associated with an
                smi2AlarmRecordTable log record. The VarBindList associated with this TRAP-TYPE
                may include the following additional information from that log record:
                smi2AlaRecBackedUpStatus,
                smi2AlaRecBackUpObject,
                smi2AlaRecTrendIndication,
```

```
                smi2AlaRecThresholdInfo,
                smi2AlaRecNotificationIdentifier,
                smi2AlaRecAdditionalText."
                REFERENCE
                "NOTIFICATION smi2Doc:processingErrorAlarm { smi2Notification 10 }
                FIELD smi2Doc:probableCause MAPS TO smi2AlaRecProbableCause,
                FIELD smi2Doc:perceivedSeverity MAPS TO smi2AlaRecPerceivedSeverity"
::=     10
```

-- Class Table

```
smi2AlarmRecordTable OBJECT-TYPE
                SYNTAX SEQUENCE OF Smi2AlarmRecordEntry
                ACCESS not-accessible
                STATUS mandatory
                DESCRIPTION
                "This managed object is used to represent logged information that resulted from alarm
                notifications or event reports. All objects of this object group are translated from the
                ISO/CCITT alarmRecord managed object class according to NMF IIMCOMIBTRANS
                and conform to behavioral requirements that it specifies.
                alarmRecordBehaviour BEHAVIOUR
                DEFINED AS This managed object is used to represent logged information that
                resulted from alarm notifications or event reports.
                logRecordBehaviour BEHAVIOUR
                DEFINED AS This managed object represents the information stored in the logs."
                REFERENCE
                "MANAGED OBJECT CLASS
                CCITT Rec. X.721|ISO/IEC 10165-2:1992:alarmRecord { smi2MObjectClass 1 }"
::=     {smi2TranslatedObjects 1 1}
```

-- Class Entry

```
smi2AlarmRecordEntry OBJECT-TYPE
                SYNTAX Smi2AlarmRecordEntry
                ACCESS not-accessible
                STATUS mandatory
                DESCRIPTION
                "This represents an entry in the smi2AlarmRecordTable."
                INDEX {smi2AlaRecTableIndex}
::=     {smi2TranslatedObjects 1 1 1}

Smi2AlarmRecordEntry ::= SEQUENCE
        {
                smi2AlaRecTableIndex                TableIndex,
                smi2AlaRecLoggingTime               TimeStamp,
                smi2AlaRecManagedObjectClass        ChoiceOfIntOrOID,
                smi2AlaRecManagedObjectInstance     InstancePointer,
                smi2AlaRecEventType                 ChoiceOfIntOrOID,
                smi2AlaRecEventTime                 TimeStamp,
                smi2AlaRecNotificationIdentifier    INTEGER,
                smi2AlaRecCorrelatedNotifications   MultiComplex,
                smi2AlaRecAdditionalText            GraphicString,
                smi2AlaRecAdditionalInformation     MultiComplex,
                smi2AlaRecProbableCause             ChoiceOfIntOrOID,
```

```
                smi2AlaRecPerceivedSeverity         PerceivedSeverity,
                smi2AlaRecSpecificProblems          MultiComplex,
                smi2AlaRecBackedUpStatus            TruthValue,
                smi2AlaRecBackUpObject              InstancePointer,
                smi2AlaRecTrendIndication           TrendIndication,
                smi2AlaRecThresholdInfo             Complex,
                smi2AlaRecStateChangeDefinition     MultiComplex,
                smi2AlaRecMonitoredAttributes       MultiComplex,
                smi2AlaRecProposedRepairActions     MultiComplex,
                smi2AlaRecParent                    Parent,
                smi2AlaRecRowStatus                 RowStatus
                }
```

-- Class Entry Columns

```
smi2AlaRecTableIndex OBJECT-TYPE
                SYNTAX TableIndex
                ACCESS not-accessible
                STATUS mandatory
                DESCRIPTION
                "This is an index of the smi2AlarmRecordTable."
::=     {smi2TranslatedObjects 1 1 1 1}

smi2AlaRecLoggingTime OBJECT-TYPE
                SYNTAX TimeStamp
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "The semantics of the loggingTime attribute type are specified in the Logging Time
                attribute in CCITT Rec. X.735|ISO/IEC 10164-6."
                REFERENCE
                "PACKAGE smi2Doc:logRecordPackage
                ATTRIBUTE smi2Doc:loggingTime { smi2AttributeID 59 }"
::= {smi2TranslatedObjects 1 1 1 2}

smi2AlaRecManagedObjectClass OBJECT-TYPE
                SYNTAX ChoiceOfIntOrOID
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "The managedObjectClass attribute type is specified to allow filtering of the Managed
                Object Class parameter when event reports are logged as records."
                REFERENCE
                "PACKAGE smi2Doc:eventLogRecordPackage
                ATTRIBUTE smi2Doc:managedObjectClass
                { smi2AttributeID 60 }"
::=     {smi2TranslatedObjects 1 1 1 3}

smi2AlaRecManagedObjectInstance OBJECT-TYPE
                SYNTAX InstancePointer
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
```

```
                "The managedObjectInstance attribute type is specified to allow filtering of the
                Managed Object Instance parameter when event reports are logged as records."
                REFERENCE
                "PACKAGE smi2Doc:eventLogRecordPackage
                ATTRIBUTE smi2Doc:managedObjectInstance
                { smi2AttributeID 61 }"
::=     {smi2TranslatedObjects 1 1 1 4}

smi2AlaRecEventType OBJECT-TYPE
                SYNTAX ChoiceOfIntOrOID
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "The semantics of the eventType attribute type are specified in the Event type
                parameter in CCITT Rec. X.710|ISO/IEC 9595."
                REFERENCE
                "PACKAGE smi2Doc:eventLogRecordPackage
                ATTRIBUTE smi2Doc:eventType { smi2AttributeID 14 }"
::=     {smi2TranslatedObjects 1 1 1 5}

smi2AlaRecEventTime OBJECT-TYPE
                SYNTAX TimeStamp
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "The semantics of the eventTime attribute type are specified in the Event Time
                parameter in CCITT Rec. X.710|ISO/IEC 9595. This object belongs to the
                eventTimePackage package, present if the event time parameter was present in the
                received event report."
                REFERENCE
                "PACKAGE smi2Doc:eventTimePackage {smi2Package 11}
                ATTRIBUTE smi2Doc:eventTime { smi2AttributeID 13 }"
::=     {smi2TranslatedObjects 1 1 1 6}

smi2AlaRecNotificationIdentifier OBJECT-TYPE
                SYNTAX INTEGER
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "The semantics of the notificationIdentifier attribute type are specified in the
                Notification identifier attribute in CCITT Rec. X.733|ISO/IEC 10164-4. This object
                belongs to the notificationIdentifierPackage package, present if the notification
                Identifier parameter is present in the notification or event report corresponding to the
                instance of an event record or an instance of its subclasses."
                REFERENCE
                "PACKAGE smi2Doc:notificationIdentifierPackage
                {smi2Package 24}
                ATTRIBUTE smi2Doc:notificationIdentifier
                { smi2AttributeID 16 }"
::=     {smi2TranslatedObjects 1 1 1 7}

smi2AlaRecCorrelatedNotifications OBJECT-TYPE
                SYNTAX MultiComplex
                ACCESS read-only
```

```
                STATUS mandatory
                DESCRIPTION
                "The semantics of the correlatedNotifications attribute type are specified in the
                Correlated Notifications parameter in CCITT Rec. X.733|ISO/IEC 10164-4. This object
                belongs to the correlatedNotificationsPackage package, present if the
                correlatedNotifications parameter is present in the notification or event report
                corresponding to the instance of an event record or an instance of its subclasses."
                REFERENCE
                "PACKAGE smi2Doc:correlatedNotificationsPackage
                {smi2Package 23}
                ATTRIBUTE smi2Doc:correlatedNotifications
                { smi2AttributeID 12 }"
        DEFVAL {smi2AlaRecCorrelatedNotificationsEntry}
::=     {smi2TranslatedObjects 1 1 1 8}

smi2AlaRecAdditionalText OBJECT-TYPE
                SYNTAX GraphicString
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "The semantics of the additionalText attribute type are specified in the Additional Text
                parameter in CCITT Rec. X.733|ISO/IEC 10164-4. This object belongs to the
                additionalTextPackage package, present if the Additional text parameter is present in
                the notification or event report corresponding to the instance of an event record or an
                instance of its subclasses."
                REFERENCE
                "PACKAGE smi2Doc:additionalTextPackage
                {smi2Package 19}
                ATTRIBUTE smi2Doc:additionalText
                { smi2AttributeID 7 }"
::=     {smi2TranslatedObjects 1 1 1 9}

smi2AlaRecAdditionalInformation
                OBJECT-TYPE
                SYNTAX MultiComplex
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "The semantics of the additionalInformation attribute type are specified in the
                Additional Information parameter in CCITT Rec. X.733|ISO/IEC 10164-4. This object
                belongs to the additionalInformationPackage package, present if the Additional
                information parameter is present in the notification or event report corresponding to
                the instance of an event record or an instance of its subclasses."
                REFERENCE
                "PACKAGE smi2Doc:additionalInformationPackage
                {smi2Package 18}
                ATTRIBUTE smi2Doc:additionalInformation
                { smi2AttributeID 6 }"
        DEFVAL {smi2AlaRecAdditionalInformationEntry}
::=     {smi2TranslatedObjects 1 1 1 10}

smi2AlaRecProbableCause
                OBJECT-TYPE
                SYNTAX ChoiceOfIntOrOID
```

```
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "The semantics of the probableCause attribute type are specified in the Probable
                Cause parameter in CCITT Rec. X.733|ISO/IEC 10164-4."
                REFERENCE
                "PACKAGE smi2Doc:alarmRecordPackage
                ATTRIBUTE smi2Doc:probableCause
                { smi2AttributeID 18 }"
::=     {smi2TranslatedObjects 1 1 1 11}

smi2AlaRecPerceivedSeverity
                OBJECT-TYPE
                SYNTAX PerceivedSeverity
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "The semantics of the perceivedSeverity attribute type are specified in the Perceived
                Severity parameter in CCITT Rec. X.733|ISO/IEC 10164-4. ENUMERATED maps to
                INTEGER, value (0) maps to (32767)."
                REFERENCE
                "PACKAGE smi2Doc:alarmRecordPackage
                ATTRIBUTE smi2Doc:perceivedSeverity
                { smi2AttributeID 17 }"
::=     {smi2TranslatedObjects 1 1 1 12}

PerceivedSeverity       ::=     INTEGER {
                                        indeterminate   (32767),
                                        critical        (1),
                                        major           (2),
                                        minor           (3),
                                        warning         (4),
                                        cleared         (5)
                                                }

smi2AlaRecSpecificProblems
                OBJECT-TYPE
                SYNTAX MultiComplex
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "The semantics of the specificProblems attribute type are specified in the Specific
                Problems parameter in CCITT Rec. X.733|ISO/IEC 10164-4. This object belongs to
                the specificProblemsPackage package, present if the Specific problems parameter is
                present in the alarm notification or event report corresponding to the instance of alarm
                record."
                REFERENCE
                "PACKAGE smi2Doc:specificProblemsPackage
                {smi2Package 1}
                ATTRIBUTE smi2Doc:specificProblems
                { smi2AttributeID 27 }"
                DEFVAL {smi2AlaRecSpecificProblemsEntry}
::=     {smi2TranslatedObjects 1 1 1 13}
```

```
smi2AlaRecBackedUpStatus
                OBJECT-TYPE
                SYNTAX TruthValue
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "The semantics of the backedUpStatus attribute type are specified in the Backed-up
                Status parameter in CCITT Rec. X.733|ISO/IEC 10164-4. This object belongs to the
                backedUpStatusPackage package, present if the  Backed up status parameter is
                present in the alarm notification or event report corresponding to the instance of alarm
                record."
                REFERENCE
                "PACKAGE smi2Doc:backedUpStatusPackage
                {smi2Package 2}
                ATTRIBUTE smi2Doc:backedUpStatus
                { smi2AttributeID 41 }"
::=     {smi2TranslatedObjects 1 1 1 14}

smi2AlaRecBackUpObject
                OBJECT-TYPE
                SYNTAX InstancePointer
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "The semantics of the backUpObject attribute type are specified in the Back-Up
                Object relationship attribute in CCITT Rec. X.732|ISO/IEC 10164-3. This attribute is
                also used in CCITT Rec. X.734|ISO/IEC 10164-4. This object belongs to the
                backUpObjectPackage package, present if the Backup object parameter is present in
                the alarm notification or event report corresponding to the instance of alarm record. A
                NULL value shall be indicated by {0 0}"
                REFERENCE
                "PACKAGE smi2Doc:backUpObjectPackage {smi2Package 3}
                ATTRIBUTE smi2Doc:backUpObject {smi2AttributeID 40}"
::=     {smi2TranslatedObjects 1 1 1 15}

smi2AlaRecTrendIndication OBJECT-TYPE
                SYNTAX TrendIndication
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "The semantics of the trendIndication attribute type are specified in the Trend
                Indication parameter in CCITT Rec. X.733|ISO/IEC 10164-4.  This object belongs to
                the trendIndicationPackage package, present if the Trend indication parameter is
                present in the alarm notification or event report corresponding to the instance of alarm
                record. ENUMERATED maps to INTEGER, value (0) maps to (32767)."
                REFERENCE
                "PACKAGE smi2Doc:trendIndicationPackage
                {smi2Package 4}
                ATTRIBUTE smi2Doc:trendIndication
                {smi2AttributeID 30}"
::=     {smi2TranslatedObjects 1 1 1 16}

TrendIndication ::=     INTEGER {
                lessSevere      (32767),
```

```
                noChange        (1),
                moreSevere      (2)
                }


smi2AlaRecThresholdInfo OBJECT-TYPE
                SYNTAX Complex
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "The semantics of the thresholdInfo attribute type are specified in the Threshold Info
                parameter in CCITT Rec. X.733|ISO/IEC 10164-4. This object belongs to the
                thresholdInfoPackage package, present if the value for probableCause attribute is
                thresholdCrossed."
                REFERENCE
                "PACKAGE smi2Doc:thresholdInfoPackage {smi2Package 5}
                ATTRIBUTE smi2Doc:thresholdInfo {smi2AttributeID 30}"
        DEFVAL {smi2AlaRecThresholdInfoEntry}
::=     {smi2TranslatedObjects 1 1 1 17}

smi2AlaRecStateChangeDefinition OBJECT-TYPE
                SYNTAX MultiComplex
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "The semantics of the stateChangeDefinition attribute type are specified in the State
                change definition parameter in CCITT Rec. X.731|ISO/IEC 10164-2. This object
                belongs to the stateChangeDefinitionPackage package, present if there is a state
                transition for the states defined in State Management Function, corresponding to the
                alarm type specified in the alarm record."
                REFERENCE
                "PACKAGE smi2Doc:stateChangeDefinitionPackage {smi2Package 6}
                ATTRIBUTE smi2Doc:stateChangeDefinition
                {smi2AttributeID 28}"
        DEFVAL {smi2AlaRecStateChangeDefinitionEntry}
::=     {smi2TranslatedObjects 1 1 1 18}

smi2AlaRecMonitoredAttributes OBJECT-TYPE
                SYNTAX MultiComplex
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "The semantics of the monitoredAttributes attribute type are specified in the
                Monitored Attributes parameter in CCITT Rec. X.733|ISO/IEC 10164-4. This object
                belongs to the monitoredAttributesPackage package, present if the
                monitoredAttributes parameter is present in the alarm notification or event report
                corresponding to the instance of alarm record."
                REFERENCE
                "PACKAGE smi2Doc:monitoredAttributesPackage {smi2Package 7}
                ATTRIBUTE smi2Doc:monitoredAttributes
                {smi2AttributeID 15}"
        DEFVAL {smi2AlaRecMonitoredAttributesEntry}
::=     {smi2TranslatedObjects 1 1 1 19}
```

```
smi2AlaRecProposedRepairActions
                OBJECT-TYPE
                SYNTAX MultiComplex
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "The semantics of the proposedRepairActions attribute type are specified in the
                Proposed Repair Actions parameter in CCITT Rec. X.733|ISO/IEC 10164-4. This
                object belongs to the proposedRepairActionsPackage package, present if the
                proposedRepairActions parameter is present in the alarm notification or event report
                corresponding to the instance of alarm record."
                REFERENCE
                "PACKAGE smi2Doc:proposedRepairActionsPackage {smi2Package 8}
                ATTRIBUTE smi2Doc:proposedRepairActions {smi2AttributeID 19}"
        DEFVAL {smi2AlaRecProposedRepairActionsEntry}
::=     {smi2TranslatedObjects 1 1 1 20}

smi2AlaRecParent OBJECT-TYPE
                SYNTAX Parent
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
        "This object is a pointer to the parent of this entry."
::=     {smi2TranslatedObjects 1 1 1 21}

smi2AlaRecRowStatus OBJECT-TYPE
                SYNTAX RowStatus
                ACCESS read-write
                STATUS mandatory
                DESCRIPTION
        "This object indicates the status of this entry."
::=     {smi2TranslatedObjects 1 1 1 22}
```

-- Alarm Record Correlated Notifications Side Table

```
smi2AlaRecCorrelatedNotificationsTable OBJECT-TYPE
                SYNTAX SEQUENCE OF Smi2AlaRecCorrelatedNotificationsEntry
                ACCESS not-accessible
                STATUS mandatory
                DESCRIPTION
                "CorrelatedNotifications ::= SET OF SEQUENCE {
                correlatedNotifications SET OF NotificationIdentifier,
                sourceObject ObjectInstance OPTIONAL}
                HUMAN DECISION: Simplify this syntax as SET OF SEQUENCE
                {NotificationIdentifer, ObjectInstance}. This preserves all information while reducing
                the nesting of side tables. NotificationIdentifiers for the same ObjectInstance may
                appear as multiple elements of this SET (side table entries)."
::=     {smi2TranslatedObjects 1 2}

smi2AlaRecCorrelatedNotificationsEntry OBJECT-TYPE
                SYNTAX Smi2AlaRecCorrelatedNotificationsEntry
                ACCESS not-accessible
                STATUS mandatory
```

```
                DESCRIPTION
                "This represents an entry in the smi2AlaRecCorrelatedNotificationsTable."
                INDEX {smi2AlaRecTableIndex, smi2AlaRecCorNotTableIndex}
::=     {smi2TranslatedObjects 1 2 1}

Smi2AlaRecCorrelatedNotificationsEntry ::=
                SEQUENCE    {
                        smi2AlaRecCorNotTableIndex
                                TableIndex,
                        smi2AlaRecCorNotNotificationIdentifier
                                INTEGER,
                        smi2AlaRecCorNotSourceObject
                                InstancePointer,
                        smi2AlaRecCorNotRowStatus
                                RowStatus
                        }

smi2AlaRecCorNotTableIndex OBJECT-TYPE
                SYNTAX TableIndex
                ACCESS not-accessible
                STATUS mandatory
                DESCRIPTION
                "This is an index of the smi2AlaRecCorrelatedNotificationsTable."
::=     {smi2TranslatedObjects 1 2 1 1}

smi2AlaRecCorNotNotificationIdentifier OBJECT-TYPE
                SYNTAX INTEGER
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "NotificationIdentifier ::= INTEGER"
::=     {smi2TranslatedObjects 1 2 1 2}

smi2AlaRecCorNotSourceObject OBJECT-TYPE
                SYNTAX InstancePointer
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "The syntax ObjectInstance is imported from X.500. This object corresponds to the
                sourceObject option, which may or may not be present in a given table entry."
::=     {smi2TranslatedObjects 1 2 1 3}

smi2AlaRecCorNotRowStatus OBJECT-TYPE
                SYNTAX RowStatus
                ACCESS read-write
                STATUS mandatory
                DESCRIPTION
                "This object indicates the status of this entry."
::=     {smi2TranslatedObjects 1 2 1 4}
```

-- Alarm Record Additional Information Side Table

```
smi2AlaRecAdditionalInformationTable OBJECT-TYPE
                SYNTAX SEQUENCE OF Smi2AlaRecAdditionalInformationEntry
```

```
                ACCESS not-accessible
                STATUS mandatory
                DESCRIPTION
                "SET OF ManagementExtension
                ManagementExtension ::= SEQUENCE {
                identifier      OBJECT IDENTIFIER,
                significance    BOOLEAN,
                information     ANY DEFINED BY OBJECT IDENTIFIER}"
::=     {smi2TranslatedObjects 1 3}

smi2AlaRecAdditionalInformationEntry OBJECT-TYPE
                SYNTAX Smi2AlaRecAdditionalInformationEntry
                ACCESS not-accessible
                STATUS mandatory
                DESCRIPTION
                "This represents an entry in the smi2AlaRecAdditionalInformationTable."
                INDEX {smi2AlaRecTableIndex, smi2AlaRecAddInfTableIndex}
::=     {smi2TranslatedObjects 1 3 1}

Smi2AlaRecAdditionalInformationEntry ::= SEQUENCE
                        {
                        smi2AlaRecAddInfTableIndex    TableIndex,
                        smi2AlaRecAddInfSignificance  TruthValue,
                        smi2AlaRecAddInfInformation   AnyDefinedBy,
                        smi2AlaRecAddInfRowStatus     RowStatus
                        }

smi2AlaRecAddInfTableIndex OBJECT-TYPE
                SYNTAX TableIndex
                ACCESS not-accessible
                STATUS mandatory
                DESCRIPTION
                "This is an index of the smi2AlaRecAdditionalInformationTable."
::=     {smi2TranslatedObjects 1 3 1 1}

smi2AlaRecAddInfSignificance
OBJECT-TYPE
                SYNTAX TruthValue
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "significance BOOLEAN DEFAULT FALSE"
                DEFVAL {0}
::=     {smi2TranslatedObjects 1 3 1 2}

smi2AlaRecAddInfInformation
OBJECT-TYPE
                SYNTAX AnyDefinedBy
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "information     ANY DEFINED BY OBJECT IDENTIFIER
                This object is a place holder for a construct not supported at the current level of this
                specification. An implementation (at the current level) shall not populate this value."
```

```
::=     {smi2TranslatedObjects 1 3 1 3}

smi2AlaRecAddInfRowStatus
OBJECT-TYPE
                SYNTAX RowStatus
                ACCESS read-write
                STATUS mandatory
                DESCRIPTION
                "This object indicates the status of this entry."
::=     {smi2TranslatedObjects 1 3 1 4}
```

-- Alarm Record Specific Problems Side Table

```
smi2AlaRecSpecificProblemsTable OBJECT-TYPE
                SYNTAX SEQUENCE OF Smi2AlaRecSpecificProblemsEntry
                ACCESS not-accessible
                STATUS mandatory
                DESCRIPTION
                "SpecificProblems ::= SET OF SpecificIdentifier"
::=     {smi2TranslatedObjects 1 4}

smi2AlaRecSpecificProblemsEntry OBJECT-TYPE
                SYNTAX Smi2AlaRecSpecificProblemsEntry
                ACCESS not-accessible
                STATUS mandatory
                DESCRIPTION
                "This represents an entry in the smi2AlaRecSpecificProblemsTable."
                INDEX {smi2AlaRecTableIndex, smi2AlaRecSpeProTableIndex}
::=     {smi2TranslatedObjects 1 4 1}

Smi2AlaRecSpecificProblemsEntry ::= SEQUENCE
                {
                smi2AlaRecSpeProTableIndex          TableIndex,
                smi2AlaRecSpeProSpecificIdentifier  Smi2SpecificIdentifier,
                smi2AlaRecSpeProRowStatus           RowStatus
                }

smi2AlaRecSpeProTableIndex OBJECT-TYPE
                SYNTAX TableIndex
                ACCESS not-accessible
                STATUS mandatory
                DESCRIPTION
                "This is an index of the smi2AlaRecSpecificProblemsTable."
::=     {smi2TranslatedObjects 1 4 1 1}

smi2AlaRecSpeProSpecificIdentifier OBJECT-TYPE
                SYNTAX Smi2SpecificIdentifier
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "Refer to the Smi2SpecificIdentifier textual convention."
::=     {smi2TranslatedObjects 1 4 1 2}

smi2AlaRecSpeProRowStatus OBJECT-TYPE
```

```
                SYNTAX RowStatus
                ACCESS read-write
                STATUS mandatory
                DESCRIPTION
                "This object indicates the status of this entry."
::=     {smi2TranslatedObjects 1 4 1 3}
```

-- Alarm Record Threshold Info Side Table

```
smi2AlaRecThresholdInfoTable    OBJECT-TYPE
                SYNTAX SEQUENCE OF Smi2AlaRecThresholdInfoEntry
                ACCESS not-accessible
                STATUS mandatory
                DESCRIPTION
                "ThresholdInfo ::=  SEQUENCE
                {triggeredThreshold     AttributeId,
                observedValue           ObservedValue,
                thresholdLevel          [1] ThresholdLevelInd OPTIONAL,
                armTime                 [2] GeneralizedTime OPTIONAL}
                ThresholdLevelInd ::= CHOICE
                {up                     [1] SEQUENCE
                {high ObservedValue, low  ObservedValue OPTIONAL}
                down                    [2] SEQUENCE
                {high ObservedValue, low  ObservedValue}}
                HUMAN DECISION: ThresholdLevelInd is treated as if it were SEQUENCE {
                ThresholdLevelIndUpDown BOOLEAN, ThresholdLevelIndHigh ObservedValue,
                ThresholdLevelIndLow ObservedValue }. This translation preserves all information
                while simplifying the number and nesting of side tables."
::=     {smi2TranslatedObjects 1 5}

smi2AlaRecThresholdInfoEntry OBJECT-TYPE
                SYNTAX Smi2AlaRecThresholdInfoEntry
                ACCESS not-accessible
                STATUS mandatory
                DESCRIPTION
                "This represents an entry in the smi2AlaRecThresholdInfoTable."
                INDEX {smi2AlaRecTableIndex}
::=     {smi2TranslatedObjects 1 5 1}

Smi2AlaRecThresholdInfoEntry    ::=
                SEQUENCE  {
                        smi2AlaRecThrInfTriggeredThreshold
                                ChoiceOfIntOrOID,
                        smi2AlaRecThrInfObservedValue
                                INTEGER,
                        smi2AlaRecThrInfThresholdLevelIndUpDown
                                TruthValue,
                        smi2AlaRecThrInfThresholdLevelIndHigh
                                INTEGER,
                        smi2AlaRecThrInfThresholdLevelIndLow
                                INTEGER,
                        smi2AlaRecThrInfArmTime
                                TimeStamp,
                        smi2AlaRecThrInfRowStatus
```

RowStatus
}

smi2AlaRecThrInfTriggeredThreshold OBJECT-TYPE
SYNTAX ChoiceOfIntOrOID
ACCESS read-only
STATUS mandatory
DESCRIPTION
"AttributeId ::= CHOICE {INTEGER, REAL}"
::= {smi2TranslatedObjects 1 5 1 2}

smi2AlaRecThrInfObservedValue OBJECT-TYPE
SYNTAX INTEGER
ACCESS read-only
STATUS mandatory
DESCRIPTION
"ObservedValue ::= CHOICE {INTEGER, REAL}
UNITS Scaled to represent REAL by factor 100 (i.e., INTEGER value 3 is represented as 300, REAL value .03 is represented as 3."

::= {smi2TranslatedObjects 1 5 1 3}

smi2AlaRecThrInfThresholdLevelIndUpDown OBJECT-TYPE
SYNTAX TruthValue
ACCESS read-only
STATUS mandatory
DESCRIPTION
"True means Up; False means Down. HUMAN DECISION: Refer to smi2AlaRecThresholdInfoTable for rationale. This object corresponds to the thresholdLevel option, which may or may not be present in a given table entry."
::= {smi2TranslatedObjects 1 5 1 4}

smi2AlaRecThrInfThresholdLevelIndHigh OBJECT-TYPE
SYNTAX INTEGER
ACCESS read-only
STATUS mandatory
DESCRIPTION
"ObservedValue ::= CHOICE {INTEGER, REAL}
UNITS Scaled to represent REAL by factor 100 (i.e., INTEGER value 3 is represented as 300, REAL value .03 is represented as 3. This object corresponds to the thresholdLevel option, which may or may not be present in a given table entry."
::= {smi2TranslatedObjects 1 5 1 5}

smi2AlaRecThrInfThresholdLevelIndLow OBJECT-TYPE
SYNTAX INTEGER
ACCESS read-only
STATUS mandatory
DESCRIPTION
"ObservedValue ::= CHOICE {INTEGER, REAL}
UNITS Scaled to represent REAL by factor 100 (i.e., INTEGER value 3 is represented as 300, REAL value .03 is represented as 3."

::= {smi2TranslatedObjects 1 5 1 6}

```
smi2AlaRecThrInfArmTime OBJECT-TYPE
                SYNTAX TimeStamp
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "armTime GeneralizedTime OPTIONAL
                This object corresponds to the armTime option, which may or may not be present in a
                given table entry."
::=     {smi2TranslatedObjects 1 5 1 7}

smi2AlaRecThrInfRowStatus OBJECT-TYPE
                SYNTAX RowStatus
                ACCESS read-write
                STATUS mandatory
                DESCRIPTION
        "This object indicates the status of this entry."
::=     {smi2TranslatedObjects 1 5 1 8}
```

-- Alarm Record State Change Definition Side Table

```
smi2AlaRecStateChangeDefinitionTable OBJECT-TYPE
                SYNTAX SEQUENCE OF Smi2AlaRecStateChangeDefinitionEntry
                ACCESS not-accessible
                STATUS mandatory
                DESCRIPTION
                "StateChangeDefinition ::= AttributeValueChangeDefinition
                AttributeValueChangeDefinition ::= SET OF SEQUENCE {
                attributeID             attributeId,
                oldAttributeValue       [1] ANY DEFINED BY attributeID OPTIONAL,
                newAttributeValue       [2] ANY DEFINED BY attributeID}"
::=     {smi2TranslatedObjects 1 6}

smi2AlaRecStateChangeDefinitionEntry    OBJECT-TYPE
                SYNTAX Smi2AlaRecStateChangeDefinitionEntry
                ACCESS not-accessible
                STATUS mandatory
                DESCRIPTION
                "This represents an entry in the smi2AlaRecStateChangeDefinitionTable."
                INDEX {smi2AlaRecTableIndex, smi2AlaRecStaChaDefTableIndex}
::=     {smi2TranslatedObjects 1 6 1}

Smi2AlaRecStateChangeDefinitionEntry ::=
                SEQUENCE {
                        smi2AlaRecStaChaDefTableIndex               TableIndex,
                        smi2AlaRecStaChaDefOldAttributeValue        AnyDefinedBy,
                        smi2AlaRecStaChaDefNewAttributeValue        AnyDefinedBy,
                        smi2AlaRecStaChaDefRowStatus                RowStatus
                        }

smi2AlaRecStaChaDefTableIndex OBJECT-TYPE
                SYNTAX TableIndex
                ACCESS not-accessible
                STATUS mandatory
                DESCRIPTION
```

```
                "This is an index of the smi2AlaRecStateChangeDefinitionTable."
::=     {smi2TranslatedObjects 1 6 1 1}

smi2AlaRecStaChaDefOldAttributeValue OBJECT-TYPE
                SYNTAX AnyDefinedBy
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "oldAttributeValue      [1] ANY DEFINED BY attributeID OPTIONAL
                This object corresponds to the oldAttributeValue option, which may or may not be
                present in a given table entry."
::=     {smi2TranslatedObjects 1 6 1 2}

smi2AlaRecStaChaDefNewAttributeValue OBJECT-TYPE
                SYNTAX AnyDefinedBy
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "newAttributeValue      [1] ANY DEFINED BY attributeID"
::=     {smi2TranslatedObjects 1 6 1 3}

smi2AlaRecStaChaDefRowStatus OBJECT-TYPE
                SYNTAX RowStatus
                ACCESS read-write
                STATUS mandatory
                DESCRIPTION
                "This object indicates the status of this entry."
::=     {smi2TranslatedObjects 1 6 1 4}
```

-- Alarm Record Monitored Attributes Side Table

```
smi2AlaRecMonitoredAttributesTable OBJECT-TYPE
                SYNTAX SEQUENCE OF Smi2AlaRecMonitoredAttributesEntry
                ACCESS not-accessible
                STATUS mandatory
                DESCRIPTION
                "MonitoredAttributes ::= SET OF Attribute"
::=     {smi2TranslatedObjects 1 7}

smi2AlaRecMonitoredAttributesEntry OBJECT-TYPE
                SYNTAX Smi2AlaRecMonitoredAttributesEntry
                ACCESS not-accessible
                STATUS mandatory
                DESCRIPTION
                "This represents an entry in the smi2AlaRecMonitoredAttributesTable."
                INDEX {smi2AlaRecTableIndex, smi2AlaRecMonAttTableIndex}
::=     {smi2TranslatedObjects 1 7 1}

Smi2AlaRecMonitoredAttributesEntry ::= SEQUENCE
                {
                smi2AlaRecMonAttTableIndex   TableIndex,
                smi2AlaRecMonAttAttribute            AnyDefinedBy,
                smi2AlaRecMonAttRowStatus            RowStatus
                }
```

```
smi2AlaRecMonAttTableIndex OBJECT-TYPE
                SYNTAX TableIndex
                ACCESS not-accessible
                STATUS mandatory
                DESCRIPTION
                "This is an index of the smi2AlaRecMonitoredAttributesTable."
::=     {smi2TranslatedObjects 1 7 1 1}

smi2AlaRecMonAttAttribute OBJECT-TYPE
                SYNTAX AnyDefinedBy
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "Attribute ::= SEQUENCE
                {attributeId AttributeId,
                attributeValue ANY DEFINED BY AttributeId}"
::=     {smi2TranslatedObjects 1 7 1 2}

smi2AlaRecMonAttRowStatus OBJECT-TYPE
                SYNTAX RowStatus
                ACCESS read-write
                STATUS mandatory
                DESCRIPTION
                "This object indicates the status of this entry."
::=     {smi2TranslatedObjects 1 7 1 3}
```

-- Alarm Record Proposed Repair Actions Side Table

```
smi2AlaRecProposedRepairActionsTable OBJECT-TYPE
                SYNTAX SEQUENCE OF Smi2AlaRecProposedRepairActionsEntry
                ACCESS not-accessible
                STATUS mandatory
                DESCRIPTION
                "ProposedRepairActions ::= SET OF SpecificIdentifier"
::=     {smi2TranslatedObjects 1 8}

smi2AlaRecProposedRepairActionsEntry OBJECT-TYPE
                SYNTAX Smi2AlaRecProposedRepairActionsEntry
                ACCESS not-accessible
                STATUS mandatory
                DESCRIPTION
                "This represents an entry in the smi2AlaRecProposedRepairActionsTable."
                INDEX {smi2AlaRecTableIndex, smi2AlaRecProRepActTableIndex}
::=     {smi2TranslatedObjects 1 8 1}

Smi2AlaRecProposedRepairActionsEntry ::= SEQUENCE
                {
                smi2AlaRecProRepActTableIndex
                                        TableIndex,
                smi2AlaRecProRepActSpecificIdentifier
                                        Smi2SpecificIdentifier,
                smi2AlaRecProRepActRowStatus
                                        RowStatus
```

```
                }

smi2AlaRecProRepActTableIndex OBJECT-TYPE
                SYNTAX TableIndex
                ACCESS not-accessible
                STATUS mandatory
                DESCRIPTION
                "This is an index of the smi2AlaRecProposedRepairActionsTable."
::=     {smi2TranslatedObjects 1 8 1 1}

smi2AlaRecProRepActSpecificIdentifier OBJECT-TYPE
                SYNTAX Smi2SpecificIdentifier
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "Refer to the Smi2SpecificIdentifier textual convention."
::=     {smi2TranslatedObjects 1 8 1 2}

smi2AlaRecProRepActRowStatus OBJECT-TYPE
                SYNTAX RowStatus
                ACCESS read-write
                STATUS mandatory
                DESCRIPTION
                "This object indicates the status of this entry."
::=     {smi2TranslatedObjects 1 8 1 3}
```

-- Discriminator Construct Side Table

```
-- The following table is included to illustrate the concepts
-- of choice and recursion.  This table is NOT a side table
-- which is part of the alarmRecord OBJECT-GROUP.  Rather,
-- this table would be a side table of another OBJECT-GROUP
-- representing the eventForwardingDiscriminator MO CLASS.
--
--      eventForwardingDiscriminator OBJECT-GROUP
--      ::= {smi2TranslatedObjects 4}
--
--      smi2EventForwardingDiscriminatorTable OBJECT-TYPE
--      ::= {smi2TranslatedObjects 4 1}
--
--      smi2EventForwardingDiscriminatorEntry OBJECT-TYPE
--      ::= {smi2TranslatedObjects 4 1 1}
--
--      Smi2EventForwardingDiscriminatorEntry ::= SEQUENCE
--              { ... other attributes ...
--              smi2EveForDisDiscriminatorConstruct Complex,
--                ... remaining attributes ... }
--
--      The DiscriminatorConstruct ATTRIBUTE would be expanded
--      as the side table which is shown below. In order to be
--      brief, only two component fields of the Discriminator
--      Construct are expanded in this example (and & not)
--      in order to illustrate CHOICE syntax and Recursion.
```

```
smi2EveForDisDiscriminatorConstructTable OBJECT-TYPE
                SYNTAX SEQUENCE OF Smi2EveForDisDiscriminatorConstructEntry
                ACCESS not-accessible
                STATUS mandatory
                DESCRIPTION
                "DiscriminatorConstruct ::= CMISFilter"
::= {smi2TranslatedObjects 4 2}

smi2EveForDisDiscriminatorConstructEntry OBJECT-TYPE
                SYNTAX Smi2EveForDisDiscriminatorConstructEntry
                ACCESS not-accessible
                STATUS mandatory
                DESCRIPTION
                "This represents an entry in the smi2EveForDisDiscriminatorConstructTable."
                INDEX {smi2EveForDisTableIndex}
                -- Note this Index has not been included in this
                -- example but was shown above in comments.
::= {smi2TranslatedObjects 4 2 1}

Smi2EveForDisDiscriminatorConstructEntry ::=
        SEQUENCE {
        smi2EveForDisDisConAnd                  MultiComplex,
        smi2EveForDisDisConNotRecursion         Recursion,
        smi2EveForDisDisConNotRecursivelyNested RecursivelyNested,
        smi2EveForDisDisConRowStatus            RowStatus
        }

smi2EveForDisDisConAnd OBJECT-TYPE
                SYNTAX MultiComplex
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "and IMPLICT SET OF CMISFilter
                This object corresponds to the and option of the CMISFilter choice. Only one such
                choice may be present in a table. Setting a new value for this choice implicitly deletes
                any pre-existing value."
-- DEFVAL {smi2EveForDisDisConAndEntry}
-- The smi2EveForDisDisConAndTable and smi2EveForDisDisConAndEntry would be
-- defined as side tables. There are not expanded here.
::= {smi2TranslatedObjects 4 2 1 1}

smi2EveForDisDisConNotRecursion OBJECT-TYPE
                SYNTAX Recursion
                ACCESS read-write
                STATUS mandatory
                DESCRIPTION
                "not CMISFilter
                This object corresponds to the not option of the CMISFilter choice. Only one such
                choice may be present in a table. Setting a new value for this choice implicitly deletes
                any pre-existing value."
::= {smi2TranslatedObjects 4 2 1 2}

smi2EveForDisDisConNotRecursivelyNested OBJECT-TYPE
                SYNTAX RecursivelyNested
```

```
                ACCESS read-only
                STATUS mandatory
                DESCRIPTION
                "not CMISFilter
                This object corresponds to the and option of the CMISFilter choice. Only one such
                choice may be present in a table. Setting a new value for this choice implicitly deletes
                any pre-existing value."
::= {smi2TranslatedObjects 4 2 1 3}

smi2EveForDisDisConRowStatus OBJECT-TYPE
                SYNTAX RowStatus
                ACCESS read-write
                STATUS mandatory
                DESCRIPTION
                "This object indicates the status of this entry."
::= {smi2TranslatedObjects 4 2 1 4}

END
```

A.3 OUTPUT SNMPV2 MANAGEMENT INFORMATION BASE

```
-- ******************************************************
--                    Partial Translation of
--       CCITT Rec. X.721 | ISO/IEC 10165-2 : 1992
--            Definition of Management Information (DMI)
--                    into Internet SNMPv2 MIB
-- ******************************************************

MIBsmi2SNMPv2Increment1 DEFINITIONS ::= BEGIN
IMPORTS

smi2MObjectClass, smi2Package
        FROM ManagedObjectClassDefinitions
        {joint-iso-ccitt ms(9) smi(3) part2(2) asn1Module(2) 0}

smi2AttributeID
        FROM Attribute-ASN1Module
        {joint-iso-ccitt ms(9) smi(3) part2(2) asn1Module(2) 1}

smi2Notification
        FROM Notification-ASN1Module
        {joint-iso-ccitt ms(9) smi(3) part2(2) asn1Module(2) 2}

AnyDefinedBy, ChoiceOfIntOrOID, Complex, GraphicString,
Pointer, TableIndex, MultiComplex, Recursion,
RecursivelyNested, SourceIndicator, Parent
        FROM IIMComibtransConventions

Smi2SpecificIdentifier
        FROM MIBsmi2ConventionsIncrement1

iimcIIMCOMIBTRANS
```

```
        FROM IimcAssignedOIDs
        {iso(1) member-body(2) 124 forum(360501) iimcManual(15) iimcModule(0) 1}

smi2Translated, smi2TranslatedModules,
smi2TranslatedObjects, smi2TranslatedNotifications
        FROM MIBsmi2ComplianceIncrement1

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

InstancePointer, TimeStamp, RowStatus, TruthValue
        FROM SNMPv2-TC

OBJECT-GROUP
        FROM SNMPv2-CONF;

-- Module Identity Macro

mibsmi2SNMPv2Increment1 MODULE-IDENTITY
                LAST-UPDATED "9310150000Z"
                ORGANIZATION "Network Management Forum"
                CONTACT-INFO
                "AIII Subteam
                1201 Mt. Kemble Avenue
                Morristown, NJ 07960-6628 USA
                Tel: +1 201-425-1900
                Fax: +1 201-425-1515
                Email: iimc@thumper.bellcore.com"
                DESCRIPTION
                "This module provides a partial translation of CCITT Rec. X.721|ISO/IEC 10165-
                2:1992, Definition of Management Information (DMI), for use with SNMPv2.

                WARNING:
                Currently, this module contains translated SNMPv2 entities corresponding to
                the DMI alarmRecord managed object class and processingErrorAlarm
                notification. This is intended to illustrate the translation procedures specified
                by IIMCOMIBTRANS. The definitions contained in this module (including all
                object identifer assignments and conformance macros) are intended only as an
                illustration, and are non-normative. CONFORMANCE CANNOT BE CLAIMED TO
                THIS MODULE."
::= { smi2TranslatedModules 2 1 }

alarmRecord OBJECT-GROUP
                OBJECTS {
                smi2AlaRecLoggingTime,
                smi2AlaRecManagedObjectClass,
                smi2AlaRecManagedObjectInstance,
                smi2AlaRecEventType,
                smi2AlaRecEventTime,
                smi2AlaRecNotificationIdentifier,
                smi2AlaRecCorrelatedNotifications,
                smi2AlaRecAdditionalText,
                smi2AlaRecAdditionalInformation,
```

```
                smi2AlaRecProbableCause,
                smi2AlaRecPerceivedSeverity,
                smi2AlaRecSpecificProblems,
                smi2AlaRecBackedUpStatus,
                smi2AlaRecBackUpObject,
                smi2AlaRecTrendIndication,
                smi2AlaRecThresholdInfo,
                smi2AlaRecStateChangeDefinition,
                smi2AlaRecMonitoredAttributes,
                smi2AlaRecProposedRepairActions,
                smi2AlaRecParent,
                smi2AlaRecRowStatus,
                smi2AlaRecCorNotNotificationIdentifier,
                smi2AlaRecCorNotSourceObject,
                smi2AlaRecCorNotRowStatus,
                smi2AlaRecAddInfSignificance,
                smi2AlaRecAddInfInformation,
                smi2AlaRecAddInfRowStatus,
                smi2AlaRecSpeProSpecificIdentifier,
                smi2AlaRecSpeProRowStatus,
                smi2AlaRecThrInfTriggeredThreshold,
                smi2AlaRecThrInfObservedValue,
                smi2AlaRecThrInfThresholdLevelIndUpDown,
                smi2AlaRecThrInfThresholdLevelIndHigh,
                smi2AlaRecThrInfThresholdLevelIndLow,
                smi2AlaRecThrInfArmTime,
                smi2AlaRecThrInfRowStatus,
                smi2AlaRecStaChaDefOldAttributeValue,
                smi2AlaRecStaChaDefNewAttributeValue,
                smi2AlaRecStaChaDefRowStatus,
                smi2AlaRecMonAttAttribute,
                smi2AlaRecMonAttRowStatus,
                smi2AlaRecProRepActSpecificIdentifier,
                smi2AlaRecProRepActRowStatus
                }
                STATUS current
                DESCRIPTION "This OBJECT-GROUP represents the MANAGED OBJECT CLASS
                CCITT Rec. X.721|ISO/IEC 10165-2:1992:alarmRecord,  translated according to the
                procedures specified by IIMCOMIBTRANS into the OBJECT-TYPE
                smi2AlarmRecordTable. Refer to the smi2AlarmRecordTable OBJECT-TYPE macro
                for additional information."
                REFERENCE "REFMACRO smi2Doc CCITT Rec. X.721|ISO/IEC 10165-2:1992"
::=     {smi2TranslatedObjects 1}


smi2ProcessingErrorAlarm NOTIFICATION-TYPE
        OBJECTS {
                smi2AlaRecManagedObjectInstance,
                smi2AlaRecEventTime,
                smi2AlaRecProbableCause,
                smi2AlaRecPerceivedSeverity
                }
                STATUS current
```

```
                DESCRIPTION "This notification type is used to report processing failure in a
                managed object. This notification carries information associated with an
                smi2AlarmRecordTable log record. The VarBindList associated with this
                NOTIFICATION-TYPE may include the following additional information from that log
                record:
                smi2AlaRecBackedUpStatus,
                smi2AlaRecBackUpObject,
                smi2AlaRecTrendIndication,
                smi2AlaRecThresholdInfo,
                smi2AlaRecNotificationIdentifier,
                smi2AlaRecAdditionalText."
                REFERENCE
                "NOTIFICATION smi2Doc:processingErrorAlarm { smi2Notification 10 }
                FIELD smi2Doc:probableCause MAPS TO smi2AlaRecProbableCause,
                FIELD smi2Doc:perceivedSeverity MAPS TO smi2AlaRecPerceivedSeverity"
::=     { smi2TranslatedNotifications 10 }

-- Class Table

smi2AlarmRecordTable OBJECT-TYPE
                SYNTAX SEQUENCE OF Smi2AlarmRecordEntry
                MAX-ACCESS not-accessible
                STATUS current
                DESCRIPTION
                "This managed object is used to represent logged information that resulted from alarm
                notifications or event reports. All objects of this object group are translated from the
                ISO/CCITT alarmRecord managed object class according to NMF IIMCOMIBTRANS
                and conform to behavioral requirements that it specifies.
                alarmRecordBehaviour BEHAVIOUR
                DEFINED AS This managed object is used to represent logged information that
                resulted from alarm notifications or event reports.
                logRecordBehaviour BEHAVIOUR
                DEFINED AS This managed object represents the information stored in the logs."
                REFERENCE
                "MANAGED OBJECT CLASS
                CCITT Rec. X.721|ISO/IEC 10165-2:1992:alarmRecord { smi2MObjectClass 1 }"
::=     {smi2TranslatedObjects 1 1}

-- Class Entry

smi2AlarmRecordEntry OBJECT-TYPE
                SYNTAX Smi2AlarmRecordEntry
                MAX-ACCESS not-accessible
                STATUS current
                DESCRIPTION
                "This represents an entry in the smi2AlarmRecordTable."
                INDEX {smi2AlaRecTableIndex}
::=     {smi2TranslatedObjects 1 1 1}

Smi2AlarmRecordEntry ::= SEQUENCE
        {
                smi2AlaRecTableIndex                TableIndex,
                smi2AlaRecLoggingTime               TimeStamp,
```

```
                smi2AlaRecManagedObjectClass        ChoiceOfIntOrOID,
                smi2AlaRecManagedObjectInstance     InstancePointer,
                smi2AlaRecEventType                 ChoiceOfIntOrOID,
                smi2AlaRecEventTime                 TimeStamp,
                smi2AlaRecNotificationIdentifier    Integer32,
                smi2AlaRecCorrelatedNotifications   MultiComplex,
                smi2AlaRecAdditionalText            GraphicString,
                smi2AlaRecAdditionalInformation     MultiComplex,
                smi2AlaRecProbableCause             ChoiceOfIntOrOID,
                smi2AlaRecPerceivedSeverity         PerceivedSeverity,
                smi2AlaRecSpecificProblems          MultiComplex,
                smi2AlaRecBackedUpStatus            TruthValue,
                smi2AlaRecBackUpObject              InstancePointer,
                smi2AlaRecTrendIndication           TrendIndication,
                smi2AlaRecThresholdInfo             Complex,
                smi2AlaRecStateChangeDefinition     MultiComplex,
                smi2AlaRecMonitoredAttributes       MultiComplex,
                smi2AlaRecProposedRepairActions     MultiComplex,
                smi2AlaRecParent                    Parent,
                smi2AlaRecRowStatus                 RowStatus
                }
```

-- Class Entry Columns

```
smi2AlaRecTableIndex OBJECT-TYPE
                SYNTAX TableIndex
                MAX-ACCESS not-accessible
                STATUS current
                DESCRIPTION
                "This is an index of the smi2AlarmRecordTable."
::=     {smi2TranslatedObjects 1 1 1 1}

smi2AlaRecLoggingTime OBJECT-TYPE
                SYNTAX TimeStamp
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "The semantics of the loggingTime attribute type are specified in the Logging Time
                attribute in CCITT Rec. X.735|ISO/IEC 10164-6."
                REFERENCE
                "PACKAGE smi2Doc:logRecordPackage
                ATTRIBUTE smi2Doc:loggingTime { smi2AttributeID 59 }"
::= {smi2TranslatedObjects 1 1 1 2}

smi2AlaRecManagedObjectClass OBJECT-TYPE
                SYNTAX ChoiceOfIntOrOID
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "The managedObjectClass attribute type is specified to allow filtering of the Managed
                Object Class parameter when event reports are logged as records."
                REFERENCE
                "PACKAGE smi2Doc:eventLogRecordPackage
                ATTRIBUTE smi2Doc:managedObjectClass
```

```
                { smi2AttributeID 60 }"
::=     {smi2TranslatedObjects 1 1 1 3}

smi2AlaRecManagedObjectInstance OBJECT-TYPE
                SYNTAX InstancePointer
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "The managedObjectInstance attribute type is specified to allow filtering of the
                Managed Object Instance parameter when event reports are logged as records."
                REFERENCE
                "PACKAGE smi2Doc:eventLogRecordPackage
                ATTRIBUTE smi2Doc:managedObjectInstance
                { smi2AttributeID 61 }"
::=     {smi2TranslatedObjects 1 1 1 4}

smi2AlaRecEventType OBJECT-TYPE
                SYNTAX ChoiceOfIntOrOID
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "The semantics of the eventType attribute type are specified in the Event type
                parameter in CCITT Rec. X.710|ISO/IEC 9595."
                REFERENCE
                "PACKAGE smi2Doc:eventLogRecordPackage
                ATTRIBUTE smi2Doc:eventType { smi2AttributeID 14 }"
::=     {smi2TranslatedObjects 1 1 1 5}

smi2AlaRecEventTime OBJECT-TYPE
                SYNTAX TimeStamp
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "The semantics of the eventTime attribute type are specified in the Event Time
                parameter in CCITT Rec. X.710|ISO/IEC 9595. This object belongs to the
                eventTimePackage package, present if the event time parameter was present in the
                received event report."
                REFERENCE
                "PACKAGE smi2Doc:eventTimePackage {smi2Package 11}
                ATTRIBUTE smi2Doc:eventTime { smi2AttributeID 13 }"
::=     {smi2TranslatedObjects 1 1 1 6}

smi2AlaRecNotificationIdentifier OBJECT-TYPE
                SYNTAX Integer32
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "The semantics of the notificationIdentifier attribute type are specified in the
                Notification identifier attribute in CCITT Rec. X.733|ISO/IEC 10164-4. This object
                belongs to the notificationIdentifierPackage package, present if the notification
                Identifier parameter is present in the notification or event report corresponding to the
                instance of an event record or an instance of its subclasses."
                REFERENCE
                "PACKAGE smi2Doc:notificationIdentifierPackage
```

```
                {smi2Package 24}
                ATTRIBUTE smi2Doc:notificationIdentifier
                { smi2AttributeID 16 }"
::=     {smi2TranslatedObjects 1 1 1 7}

smi2AlaRecCorrelatedNotifications OBJECT-TYPE
                SYNTAX MultiComplex
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "The semantics of the correlatedNotifications attribute type are specified in the
                Correlated Notifications parameter in CCITT Rec. X.733|ISO/IEC 10164-4. This object
                belongs to the correlatedNotificationsPackage package, present if the
                correlatedNotifications parameter is present in the notification or event report
                corresponding to the instance of an event record or an instance of its subclasses."
                REFERENCE
                "PACKAGE smi2Doc:correlatedNotificationsPackage
                {smi2Package 23}
                ATTRIBUTE smi2Doc:correlatedNotifications
                { smi2AttributeID 12 }"
        DEFVAL {smi2AlaRecCorrelatedNotificationsEntry}
::=     {smi2TranslatedObjects 1 1 1 8}

smi2AlaRecAdditionalText OBJECT-TYPE
                SYNTAX GraphicString
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "The semantics of the additionalText attribute type are specified in the Additional Text
                parameter in CCITT Rec. X.733|ISO/IEC 10164-4. This object belongs to the
                additionalTextPackage package, present if the Additional text parameter is present in
                the notification or event report corresponding to the instance of an event record or an
                instance of its subclasses."
                REFERENCE
                "PACKAGE smi2Doc:additionalTextPackage
                {smi2Package 19}
                ATTRIBUTE smi2Doc:additionalText
                { smi2AttributeID 7 }"
::=     {smi2TranslatedObjects 1 1 1 9}

smi2AlaRecAdditionalInformation
                OBJECT-TYPE
                SYNTAX MultiComplex
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "The semantics of the additionalInformation attribute type are specified in the
                Additional Information parameter in CCITT Rec. X.733|ISO/IEC 10164-4. This object
                belongs to the additionalInformationPackage package, present if the Additional
                information parameter is present in the notification or event report corresponding to
                the instance of an event record or an instance of its subclasses."
                REFERENCE
                "PACKAGE smi2Doc:additionalInformationPackage
                {smi2Package 18}
```

```
                ATTRIBUTE smi2Doc:additionalInformation
                { smi2AttributeID 6 }"
        DEFVAL {smi2AlaRecAdditionalInformationEntry}
::=     {smi2TranslatedObjects 1 1 1 10}

smi2AlaRecProbableCause
                OBJECT-TYPE
                SYNTAX ChoiceOfIntOrOID
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "The semantics of the probableCause attribute type are specified in the Probable
                Cause parameter in CCITT Rec. X.733|ISO/IEC 10164-4."
                REFERENCE
                "PACKAGE smi2Doc:alarmRecordPackage
                ATTRIBUTE smi2Doc:probableCause
                { smi2AttributeID 18 }"
::=     {smi2TranslatedObjects 1 1 1 11}

smi2AlaRecPerceivedSeverity
                OBJECT-TYPE
                SYNTAX PerceivedSeverity
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "The semantics of the perceivedSeverity attribute type are specified in the Perceived
                Severity parameter in CCITT Rec. X.733|ISO/IEC 10164-4. ENUMERATED maps to
                INTEGER, value (0) maps to (32767)."
                REFERENCE
                "PACKAGE smi2Doc:alarmRecordPackage
                ATTRIBUTE smi2Doc:perceivedSeverity
                { smi2AttributeID 17 }"
::=     {smi2TranslatedObjects 1 1 1 12}

PerceivedSeverity       ::=     INTEGER {
                                        indeterminate   (32767),
                                        critical        (1),
                                        major           (2),
                                        minor           (3),
                                        warning         (4),
                                        cleared         (5)
                                                }

smi2AlaRecSpecificProblems
                OBJECT-TYPE
                SYNTAX MultiComplex
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "The semantics of the specificProblems attribute type are specified in the Specific
                Problems parameter in CCITT Rec. X.733|ISO/IEC 10164-4. This object belongs to
                the specificProblemsPackage package, present if the Specific problems parameter is
                present in the alarm notification or event report corresponding to the instance of alarm
                record."
```

```
                REFERENCE
                "PACKAGE smi2Doc:specificProblemsPackage
                {smi2Package 1}
                ATTRIBUTE smi2Doc:specificProblems
                { smi2AttributeID 27 }"
                DEFVAL {smi2AlaRecSpecificProblemsEntry}
::=     {smi2TranslatedObjects 1 1 1 13}

smi2AlaRecBackedUpStatus
                OBJECT-TYPE
                SYNTAX TruthValue
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "The semantics of the backedUpStatus attribute type are specified in the Backed-up
                Status parameter in CCITT Rec. X.733|ISO/IEC 10164-4. This object belongs to the
                backedUpStatusPackage package, present if the  Backed up status parameter is
                present in the alarm notification or event report corresponding to the instance of alarm
                record."
                REFERENCE
                "PACKAGE smi2Doc:backedUpStatusPackage
                {smi2Package 2}
                ATTRIBUTE smi2Doc:backedUpStatus
                { smi2AttributeID 41 }"
::=     {smi2TranslatedObjects 1 1 1 14}

smi2AlaRecBackUpObject
                OBJECT-TYPE
                SYNTAX InstancePointer
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "The semantics of the backUpObject attribute type are specified in the Back-Up
                Object relationship attribute in CCITT Rec. X.732|ISO/IEC 10164-3. This attribute is
                also used in CCITT Rec. X.734|ISO/IEC 10164-4. This object belongs to the
                backUpObjectPackage package, present if the Backup object parameter is present in
                the alarm notification or event report corresponding to the instance of alarm record. A
                NULL value shall be indicated by {0 0}"
                REFERENCE
                "PACKAGE smi2Doc:backUpObjectPackage {smi2Package 3}
                ATTRIBUTE smi2Doc:backUpObject {smi2AttributeID 40}"
::=     {smi2TranslatedObjects 1 1 1 15}

smi2AlaRecTrendIndication OBJECT-TYPE
                SYNTAX TrendIndication
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "The semantics of the trendIndication attribute type are specified in the Trend
                Indication parameter in CCITT Rec. X.733|ISO/IEC 10164-4.  This object belongs to
                the trendIndicationPackage package, present if the Trend indication parameter is
                present in the alarm notification or event report corresponding to the instance of alarm
                record. ENUMERATED maps to INTEGER, value (0) maps to (32767)."
                REFERENCE
```

```
                "PACKAGE smi2Doc:trendIndicationPackage
                {smi2Package 4}
                ATTRIBUTE smi2Doc:trendIndication
                {smi2AttributeID 30}"
::=     {smi2TranslatedObjects 1 1 1 16}

TrendIndication ::=     INTEGER {
                lessSevere      (32767),
                noChange        (1),
                moreSevere      (2)
                }


smi2AlaRecThresholdInfo OBJECT-TYPE
                SYNTAX Complex
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "The semantics of the thresholdInfo attribute type are specified in the Threshold Info
                parameter in CCITT Rec. X.733|ISO/IEC 10164-4. This object belongs to the
                thresholdInfoPackage package, present if the value for probableCause attribute is
                thresholdCrossed."
                REFERENCE
                "PACKAGE smi2Doc:thresholdInfoPackage {smi2Package 5}
                ATTRIBUTE smi2Doc:thresholdInfo {smi2AttributeID 30}"
        DEFVAL {smi2AlaRecThresholdInfoEntry}
::=     {smi2TranslatedObjects 1 1 1 17}

smi2AlaRecStateChangeDefinition OBJECT-TYPE
                SYNTAX MultiComplex
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "The semantics of the stateChangeDefinition attribute type are specified in the State
                change definition parameter in CCITT Rec. X.731|ISO/IEC 10164-2. This object
                belongs to the stateChangeDefinitionPackage package, present if there is a state
                transition for the states defined in State Management Function, corresponding to the
                alarm type specified in the alarm record."
                REFERENCE
                "PACKAGE smi2Doc:stateChangeDefinitionPackage {smi2Package 6}
                ATTRIBUTE smi2Doc:stateChangeDefinition
                {smi2AttributeID 28}"
        DEFVAL {smi2AlaRecStateChangeDefinitionEntry}
::=     {smi2TranslatedObjects 1 1 1 18}

smi2AlaRecMonitoredAttributes OBJECT-TYPE
                SYNTAX MultiComplex
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "The semantics of the monitoredAttributes attribute type are specified in the
                Monitored Attributes parameter in CCITT Rec. X.733|ISO/IEC 10164-4. This object
                belongs to the monitoredAttributesPackage package, present if the
```

```
                monitoredAttributes parameter is present in the alarm notification or event report
                corresponding to the instance of alarm record."
                REFERENCE
                "PACKAGE smi2Doc:monitoredAttributesPackage {smi2Package 7}
                ATTRIBUTE smi2Doc:monitoredAttributes
                {smi2AttributeID 15}"
        DEFVAL {smi2AlaRecMonitoredAttributesEntry}
::=     {smi2TranslatedObjects 1 1 1 19}

smi2AlaRecProposedRepairActions
                OBJECT-TYPE
                SYNTAX MultiComplex
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "The semantics of the proposedRepairActions attribute type are specified in the
                Proposed Repair Actions parameter in CCITT Rec. X.733|ISO/IEC 10164-4. This
                object belongs to the proposedRepairActionsPackage package, present if the
                proposedRepairActions parameter is present in the alarm notification or event report
                corresponding to the instance of alarm record."
                REFERENCE
                "PACKAGE smi2Doc:proposedRepairActionsPackage {smi2Package 8}
                ATTRIBUTE smi2Doc:proposedRepairActions {smi2AttributeID 19}"
        DEFVAL {smi2AlaRecProposedRepairActionsEntry}
::=     {smi2TranslatedObjects 1 1 1 20}

smi2AlaRecParent OBJECT-TYPE
                SYNTAX Parent
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
        "This object is a pointer to the parent of this entry."
::=     {smi2TranslatedObjects 1 1 1 21}

smi2AlaRecRowStatus OBJECT-TYPE
                SYNTAX RowStatus
                MAX-ACCESS read-create
                STATUS current
                DESCRIPTION
        "This object indicates the status of this entry."
::=     {smi2TranslatedObjects 1 1 1 22}
```

-- Alarm Record Correlated Notifications Side Table

```
smi2AlaRecCorrelatedNotificationsTable OBJECT-TYPE
                SYNTAX SEQUENCE OF Smi2AlaRecCorrelatedNotificationsEntry
                MAX-ACCESS not-accessible
                STATUS current
                DESCRIPTION
                "CorrelatedNotifications ::= SET OF SEQUENCE {
                correlatedNotifications SET OF NotificationIdentifier,
                sourceObject ObjectInstance OPTIONAL}
```

```
                HUMAN DECISION: Simplify this syntax as SET OF SEQUENCE
                {NotificationIdentifer, ObjectInstance}. This preserves all information while reducing
                the nesting of side tables. NotificationIdentifiers for the same ObjectInstance may
                appear as multiple elements of this SET (side table entries)."
::=     {smi2TranslatedObjects 1 2}

smi2AlaRecCorrelatedNotificationsEntry OBJECT-TYPE
                SYNTAX Smi2AlaRecCorrelatedNotificationsEntry
                MAX-ACCESS not-accessible
                STATUS current
                DESCRIPTION
                "This represents an entry in the smi2AlaRecCorrelatedNotificationsTable."
                INDEX {smi2AlaRecTableIndex, smi2AlaRecCorNotTableIndex}
::=     {smi2TranslatedObjects 1 2 1}

Smi2AlaRecCorrelatedNotificationsEntry ::=
                SEQUENCE    {
                        smi2AlaRecCorNotTableIndex
                                TableIndex,
                        smi2AlaRecCorNotNotificationIdentifier
                                Integer32,
                        smi2AlaRecCorNotSourceObject
                                InstancePointer,
                        smi2AlaRecCorNotRowStatus
                                RowStatus
                        }

smi2AlaRecCorNotTableIndex OBJECT-TYPE
                SYNTAX TableIndex
                MAX-ACCESS not-accessible
                STATUS current
                DESCRIPTION
                "This is an index of the smi2AlaRecCorrelatedNotificationsTable."
::=     {smi2TranslatedObjects 1 2 1 1}

smi2AlaRecCorNotNotificationIdentifier OBJECT-TYPE
                SYNTAX Integer32
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "NotificationIdentifier ::= INTEGER"
::=     {smi2TranslatedObjects 1 2 1 2}

smi2AlaRecCorNotSourceObject OBJECT-TYPE
                SYNTAX InstancePointer
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "The syntax ObjectInstance is imported from X.500. This object corresponds to the
                sourceObject option, which may or may not be present in a given table entry."
::=     {smi2TranslatedObjects 1 2 1 3}

smi2AlaRecCorNotRowStatus OBJECT-TYPE
                SYNTAX RowStatus
```

```
                MAX-ACCESS read-create
                STATUS current
                DESCRIPTION
                "This object indicates the status of this entry."
::=     {smi2TranslatedObjects 1 2 1 4}

-- Alarm Record Additional Information Side Table

smi2AlaRecAdditionalInformationTable OBJECT-TYPE
                SYNTAX SEQUENCE OF Smi2AlaRecAdditionalInformationEntry
                MAX-ACCESS not-accessible
                STATUS current
                DESCRIPTION
                "SET OF ManagementExtension
                ManagementExtension ::= SEQUENCE {
                identifier      OBJECT IDENTIFIER,
                significance    BOOLEAN,
                information     ANY DEFINED BY OBJECT IDENTIFIER}"
::=     {smi2TranslatedObjects 1 3}

smi2AlaRecAdditionalInformationEntry OBJECT-TYPE
                SYNTAX Smi2AlaRecAdditionalInformationEntry
                MAX-ACCESS not-accessible
                STATUS current
                DESCRIPTION
                "This represents an entry in the smi2AlaRecAdditionalInformationTable."
                INDEX {smi2AlaRecTableIndex, smi2AlaRecAddInfTableIndex}
::=     {smi2TranslatedObjects 1 3 1}

Smi2AlaRecAdditionalInformationEntry ::= SEQUENCE
                        {
                        smi2AlaRecAddInfTableIndex      TableIndex,
                        smi2AlaRecAddInfSignificance    TruthValue,
                        smi2AlaRecAddInfInformation     AnyDefinedBy,
                        smi2AlaRecAddInfRowStatus       RowStatus
                        }

smi2AlaRecAddInfTableIndex OBJECT-TYPE
                SYNTAX TableIndex
                MAX-ACCESS not-accessible
                STATUS current
                DESCRIPTION
                "This is an index of the smi2AlaRecAdditionalInformationTable."
::=     {smi2TranslatedObjects 1 3 1 1}

smi2AlaRecAddInfSignificance
OBJECT-TYPE
                SYNTAX TruthValue
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "significance BOOLEAN DEFAULT FALSE"
                DEFVAL {0}
::=     {smi2TranslatedObjects 1 3 1 2}
```

```
smi2AlaRecAddInfInformation
OBJECT-TYPE
                SYNTAX AnyDefinedBy
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "information      ANY DEFINED BY OBJECT IDENTIFIER
                This object is a place holder for a construct not supported at the current level of this
                specification. An implementation (at the current level) shall not populate this value."
::=     {smi2TranslatedObjects 1 3 1 3}

smi2AlaRecAddInfRowStatus
OBJECT-TYPE
                SYNTAX RowStatus
                MAX-ACCESS read-create
                STATUS current
                DESCRIPTION
                "This object indicates the status of this entry."
::=     {smi2TranslatedObjects 1 3 1 4}
```

-- Alarm Record Specific Problems Side Table

```
smi2AlaRecSpecificProblemsTable OBJECT-TYPE
                SYNTAX SEQUENCE OF Smi2AlaRecSpecificProblemsEntry
                MAX-ACCESS not-accessible
                STATUS current
                DESCRIPTION
                "SpecificProblems ::= SET OF SpecificIdentifier"
::=     {smi2TranslatedObjects 1 4}

smi2AlaRecSpecificProblemsEntry OBJECT-TYPE
                SYNTAX Smi2AlaRecSpecificProblemsEntry
                MAX-ACCESS not-accessible
                STATUS current
                DESCRIPTION
                "This represents an entry in the smi2AlaRecSpecificProblemsTable."
                INDEX {smi2AlaRecTableIndex, smi2AlaRecSpeProTableIndex}
::=     {smi2TranslatedObjects 1 4 1}

Smi2AlaRecSpecificProblemsEntry ::= SEQUENCE
                {
                smi2AlaRecSpeProTableIndex          TableIndex,
                smi2AlaRecSpeProSpecificIdentifier  Smi2SpecificIdentifier,
                smi2AlaRecSpeProRowStatus           RowStatus
                }

smi2AlaRecSpeProTableIndex OBJECT-TYPE
                SYNTAX TableIndex
                MAX-ACCESS not-accessible
                STATUS current
                DESCRIPTION
                "This is an index of the smi2AlaRecSpecificProblemsTable."
::=     {smi2TranslatedObjects 1 4 1 1}
```

```
smi2AlaRecSpeProSpecificIdentifier OBJECT-TYPE
                SYNTAX Smi2SpecificIdentifier
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "Refer to the Smi2SpecificIdentifier textual convention."
::=     {smi2TranslatedObjects 1 4 1 2}

smi2AlaRecSpeProRowStatus OBJECT-TYPE
                SYNTAX RowStatus
                MAX-ACCESS read-create
                STATUS current
                DESCRIPTION
                "This object indicates the status of this entry."
::=     {smi2TranslatedObjects 1 4 1 3}
```

-- Alarm Record Threshold Info Side Table

```
smi2AlaRecThresholdInfoTable    OBJECT-TYPE
                SYNTAX SEQUENCE OF Smi2AlaRecThresholdInfoEntry
                MAX-ACCESS not-accessible
                STATUS current
                DESCRIPTION
                "ThresholdInfo ::=  SEQUENCE
                {triggeredThreshold       AttributeId,
                observedValue             ObservedValue,
                thresholdLevel            [1] ThresholdLevelInd OPTIONAL,
                armTime                   [2] GeneralizedTime OPTIONAL}
                ThresholdLevelInd ::= CHOICE
                {up                       [1] SEQUENCE
                {high ObservedValue, low  ObservedValue OPTIONAL}
                down                      [2] SEQUENCE
                {high ObservedValue, low  ObservedValue}}
                HUMAN DECISION: ThresholdLevelInd is treated as if it were SEQUENCE {
                ThresholdLevelIndUpDown BOOLEAN, ThresholdLevelIndHigh ObservedValue,
                ThresholdLevelIndLow ObservedValue }. This translation preserves all information
                while simplifying the number and nesting of side tables."
::=     {smi2TranslatedObjects 1 5}

smi2AlaRecThresholdInfoEntry OBJECT-TYPE
                SYNTAX Smi2AlaRecThresholdInfoEntry
                MAX-ACCESS not-accessible
                STATUS current
                DESCRIPTION
                "This represents an entry in the smi2AlaRecThresholdInfoTable."
                AUGMENTS {smi2AlarmRecordEntry}
::=     {smi2TranslatedObjects 1 5 1}

Smi2AlaRecThresholdInfoEntry    ::=
                SEQUENCE  {
                        smi2AlaRecThrInfTriggeredThreshold
                                        ChoiceOfIntOrOID,
                        smi2AlaRecThrInfObservedValue
```

```
                        Integer32,
                smi2AlaRecThrInfThresholdLevelIndUpDown
                        TruthValue,
                smi2AlaRecThrInfThresholdLevelIndHigh
                        Integer32,
                smi2AlaRecThrInfThresholdLevelIndLow
                        Integer32,
                smi2AlaRecThrInfArmTime
                        TimeStamp,
                smi2AlaRecThrInfRowStatus
                        RowStatus
                }

smi2AlaRecThrInfTriggeredThreshold OBJECT-TYPE
        SYNTAX ChoiceOfIntOrOID
        MAX-ACCESS read-only
        STATUS current
        DESCRIPTION
        "AttributeId ::= CHOICE {INTEGER, REAL}"
::=     {smi2TranslatedObjects 1 5 1 2}

smi2AlaRecThrInfObservedValue OBJECT-TYPE
        SYNTAX Integer32
        UNITS
        "Scaled to represent REAL by factor 100 (i.e., INTEGER value 3 is represented as
        300, REAL value .03 is represented as 3."
        MAX-ACCESS read-only
        STATUS current
        DESCRIPTION
        "ObservedValue ::= CHOICE {INTEGER, REAL}"
::=     {smi2TranslatedObjects 1 5 1 3}

smi2AlaRecThrInfThresholdLevelIndUpDown OBJECT-TYPE
        SYNTAX TruthValue
        MAX-ACCESS read-only
        STATUS current
        DESCRIPTION
        "True means Up; False means Down. HUMAN DECISION: Refer to
        smi2AlaRecThresholdInfoTable for rationale. This object corresponds to the
        thresholdLevel option, which may or may not be present in a given table entry."
::=     {smi2TranslatedObjects 1 5 1 4}

smi2AlaRecThrInfThresholdLevelIndHigh OBJECT-TYPE
        SYNTAX Integer32
        UNITS
        "Scaled to represent REAL by factor 100 (i.e., INTEGER value 3 is represented as
        300, REAL value .03 is represented as 3. This object corresponds to the
        thresholdLevel option, which may or may not be present in a given table entry."
        MAX-ACCESS read-only
        STATUS current
        DESCRIPTION
        "ObservedValue ::= CHOICE {INTEGER, REAL}"
::=     {smi2TranslatedObjects 1 5 1 5}
```

```
smi2AlaRecThrInfThresholdLevelIndLow OBJECT-TYPE
                SYNTAX Integer32
                UNITS
                "Scaled to represent REAL by factor 100 (i.e., INTEGER value 3 is represented as
                300, REAL value .03 is represented as 3."
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "ObservedValue ::= CHOICE {INTEGER, REAL}"
::=     {smi2TranslatedObjects 1 5 1 6}

smi2AlaRecThrInfArmTime OBJECT-TYPE
                SYNTAX TimeStamp
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "armTime GeneralizedTime OPTIONAL
                This object corresponds to the armTime option, which may or may not be present in a
                given table entry."
::=     {smi2TranslatedObjects 1 5 1 7}

smi2AlaRecThrInfRowStatus OBJECT-TYPE
                SYNTAX RowStatus
                MAX-ACCESS read-create
                STATUS current
                DESCRIPTION
        "This object indicates the status of this entry."
::=     {smi2TranslatedObjects 1 5 1 8}
```

-- Alarm Record State Change Definition Side Table

```
smi2AlaRecStateChangeDefinitionTable OBJECT-TYPE
                SYNTAX SEQUENCE OF Smi2AlaRecStateChangeDefinitionEntry
                MAX-ACCESS not-accessible
                STATUS current
                DESCRIPTION
                "StateChangeDefinition ::= AttributeValueChangeDefinition
                AttributeValueChangeDefinition ::= SET OF SEQUENCE {
                attributeID             attributeId,
                oldAttributeValue       [1] ANY DEFINED BY attributeID OPTIONAL,
                newAttributeValue       [2] ANY DEFINED BY attributeID}"
::=     {smi2TranslatedObjects 1 6}

smi2AlaRecStateChangeDefinitionEntry    OBJECT-TYPE
                SYNTAX Smi2AlaRecStateChangeDefinitionEntry
                MAX-ACCESS not-accessible
                STATUS current
                DESCRIPTION
                "This represents an entry in the smi2AlaRecStateChangeDefinitionTable."
                INDEX {smi2AlaRecTableIndex, smi2AlaRecStaChaDefTableIndex}
::=     {smi2TranslatedObjects 1 6 1}

Smi2AlaRecStateChangeDefinitionEntry ::=
                SEQUENCE {
```

```
                    smi2AlaRecStaChaDefTableIndex          TableIndex,
                    smi2AlaRecStaChaDefOldAttributeValue   AnyDefinedBy,
                    smi2AlaRecStaChaDefNewAttributeValue   AnyDefinedBy,
                    smi2AlaRecStaChaDefRowStatus           RowStatus
                    }

smi2AlaRecStaChaDefTableIndex OBJECT-TYPE
            SYNTAX TableIndex
            MAX-ACCESS not-accessible
            STATUS current
            DESCRIPTION
            "This is an index of the smi2AlaRecStateChangeDefinitionTable."
::=     {smi2TranslatedObjects 1 6 1 1}

smi2AlaRecStaChaDefOldAttributeValue OBJECT-TYPE
            SYNTAX AnyDefinedBy
            MAX-ACCESS read-only
            STATUS current
            DESCRIPTION
            "oldAttributeValue      [1] ANY DEFINED BY attributeID OPTIONAL
            This object corresponds to the oldAttributeValue option, which may or may not be
            present in a given table entry."
::=     {smi2TranslatedObjects 1 6 1 2}

smi2AlaRecStaChaDefNewAttributeValue OBJECT-TYPE
            SYNTAX AnyDefinedBy
            MAX-ACCESS read-only
            STATUS current
            DESCRIPTION
            "newAttributeValue      [1] ANY DEFINED BY attributeID"
::=     {smi2TranslatedObjects 1 6 1 3}

smi2AlaRecStaChaDefRowStatus OBJECT-TYPE
            SYNTAX RowStatus
            MAX-ACCESS read-create
            STATUS current
            DESCRIPTION
            "This object indicates the status of this entry."
::=     {smi2TranslatedObjects 1 6 1 4}

-- Alarm Record Monitored Attributes Side Table

smi2AlaRecMonitoredAttributesTable OBJECT-TYPE
            SYNTAX SEQUENCE OF Smi2AlaRecMonitoredAttributesEntry
            MAX-ACCESS not-accessible
            STATUS current
            DESCRIPTION
            "MonitoredAttributes ::= SET OF Attribute"
::=     {smi2TranslatedObjects 1 7}

smi2AlaRecMonitoredAttributesEntry OBJECT-TYPE
            SYNTAX Smi2AlaRecMonitoredAttributesEntry
            MAX-ACCESS not-accessible
            STATUS current
```

```
                DESCRIPTION
                "This represents an entry in the smi2AlaRecMonitoredAttributesTable."
                INDEX {smi2AlaRecTableIndex, smi2AlaRecMonAttTableIndex}
::=     {smi2TranslatedObjects 1 7 1}

Smi2AlaRecMonitoredAttributesEntry ::= SEQUENCE
                {
                smi2AlaRecMonAttTableIndex          TableIndex,
                smi2AlaRecMonAttAttribute           AnyDefinedBy,
                smi2AlaRecMonAttRowStatus           RowStatus
                }

smi2AlaRecMonAttTableIndex OBJECT-TYPE
                SYNTAX TableIndex
                MAX-ACCESS not-accessible
                STATUS current
                DESCRIPTION
                "This is an index of the smi2AlaRecMonitoredAttributesTable."
::=     {smi2TranslatedObjects 1 7 1 1}

smi2AlaRecMonAttAttribute OBJECT-TYPE
                SYNTAX AnyDefinedBy
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "Attribute ::= SEQUENCE
                {attributeId AttributeId,
                attributeValue ANY DEFINED BY AttributeId}"
::=     {smi2TranslatedObjects 1 7 1 2}

smi2AlaRecMonAttRowStatus OBJECT-TYPE
                SYNTAX RowStatus
                MAX-ACCESS read-create
                STATUS current
                DESCRIPTION
                "This object indicates the status of this entry."
::=     {smi2TranslatedObjects 1 7 1 3}

-- Alarm Record Proposed Repair Actions Side Table

smi2AlaRecProposedRepairActionsTable OBJECT-TYPE
                SYNTAX SEQUENCE OF Smi2AlaRecProposedRepairActionsEntry
                MAX-ACCESS not-accessible
                STATUS current
                DESCRIPTION
                "ProposedRepairActions ::= SET OF SpecificIdentifier"
::=     {smi2TranslatedObjects 1 8}

smi2AlaRecProposedRepairActionsEntry OBJECT-TYPE
                SYNTAX Smi2AlaRecProposedRepairActionsEntry
                MAX-ACCESS not-accessible
                STATUS current
                DESCRIPTION
                "This represents an entry in the smi2AlaRecProposedRepairActionsTable."
```

```
                INDEX {smi2AlaRecTableIndex, smi2AlaRecProRepActTableIndex}
::=     {smi2TranslatedObjects 1 8 1}

Smi2AlaRecProposedRepairActionsEntry ::= SEQUENCE
                {
                smi2AlaRecProRepActTableIndex
                                        TableIndex,
                smi2AlaRecProRepActSpecificIdentifier
                                        Smi2SpecificIdentifier,
                smi2AlaRecProRepActRowStatus
                                        RowStatus
                }

smi2AlaRecProRepActTableIndex OBJECT-TYPE
                SYNTAX TableIndex
                MAX-ACCESS not-accessible
                STATUS current
                DESCRIPTION
                "This is an index of the smi2AlaRecProposedRepairActionsTable."
::=     {smi2TranslatedObjects 1 8 1 1}

smi2AlaRecProRepActSpecificIdentifier OBJECT-TYPE
                SYNTAX Smi2SpecificIdentifier
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "Refer to the Smi2SpecificIdentifier textual convention."
::=     {smi2TranslatedObjects 1 8 1 2}

smi2AlaRecProRepActRowStatus OBJECT-TYPE
                SYNTAX RowStatus
                MAX-ACCESS read-create
                STATUS current
                DESCRIPTION
                "This object indicates the status of this entry."
::=     {smi2TranslatedObjects 1 8 1 3}
```

-- Discriminator Construct Side Table

```
-- The following table is included to illustrate the concepts
-- of choice and recursion.  This table is NOT a side table
-- which is part of the alarmRecord OBJECT-GROUP.  Rather,
-- this table would be a side table of another OBJECT-GROUP
-- representing the eventForwardingDiscriminator MO CLASS.
--
--      eventForwardingDiscriminator OBJECT-GROUP
--      ::= {smi2TranslatedObjects 4}
--
--      smi2EventForwardingDiscriminatorTable OBJECT-TYPE
--      ::= {smi2TranslatedObjects 4 1}
--
--      smi2EventForwardingDiscriminatorEntry OBJECT-TYPE
--      ::= {smi2TranslatedObjects 4 1 1}
--
```

```
--      Smi2EventForwardingDiscriminatorEntry ::= SEQUENCE
--              { ... other attributes ...
--              smi2EveForDisDiscriminatorConstruct Complex,
--                ... remaining attributes ... }
--
--      The DiscriminatorConstruct ATTRIBUTE would be expanded
--      as the side table which is shown below. In order to be
--      brief, only two component fields of the Discriminator
--      Construct are expanded in this example (and & not)
--      in order to illustrate CHOICE syntax and Recursion.

smi2EveForDisDiscriminatorConstructTable OBJECT-TYPE
                SYNTAX SEQUENCE OF Smi2EveForDisDiscriminatorConstructEntry
                MAX-ACCESS not-accessible
                STATUS current
                DESCRIPTION
                "DiscriminatorConstruct ::= CMISFilter"
::= {smi2TranslatedObjects 4 2}

smi2EveForDisDiscriminatorConstructEntry OBJECT-TYPE
                SYNTAX Smi2EveForDisDiscriminatorConstructEntry
                MAX-ACCESS not-accessible
                STATUS current
                DESCRIPTION
                "This represents an entry in the smi2EveForDisDiscriminatorConstructTable."
                AUGMENTS {smi2EventForwardingDiscriminatorEntry}
                -- Note this Entry has not been included in this
                -- example but was shown above in comments.
::= {smi2TranslatedObjects 4 2 1}

Smi2EveForDisDiscriminatorConstructEntry ::=
        SEQUENCE {
        smi2EveForDisDisConAnd                          MultiComplex,
        smi2EveForDisDisConNotRecursion                 Recursion,
        smi2EveForDisDisConNotRecursivelyNested         RecursivelyNested,
        smi2EveForDisDisConRowStatus                    RowStatus
        }

smi2EveForDisDisConAnd OBJECT-TYPE
                SYNTAX MultiComplex
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "and IMPLICT SET OF CMISFilter
                This object corresponds to the and option of the CMISFilter choice. Only one such
                choice may be present in a table. Setting a new value for this choice implicitly deletes
                any pre-existing value."
-- DEFVAL {smi2EveForDisDisConAndEntry}
-- The smi2EveForDisDisConAndTable and smi2EveForDisDisConAndEntry would be
-- defined as side tables. There are not expanded here.
::= {smi2TranslatedObjects 4 2 1 1}

smi2EveForDisDisConNotRecursion OBJECT-TYPE
                SYNTAX Recursion
```

```
                MAX-ACCESS read-write
                STATUS current
                DESCRIPTION
                "not CMISFilter
                This object corresponds to the not option of the CMISFilter choice. Only one such
                choice may be present in a table. Setting a new value for this choice implicitly deletes
                any pre-existing value."
::= {smi2TranslatedObjects 4 2 1 2}

smi2EveForDisDisConNotRecursivelyNested OBJECT-TYPE
                SYNTAX RecursivelyNested
                MAX-ACCESS read-only
                STATUS current
                DESCRIPTION
                "not CMISFilter
                This object corresponds to the and option of the CMISFilter choice. Only one such
                choice may be present in a table. Setting a new value for this choice implicitly deletes
                any pre-existing value."
::= {smi2TranslatedObjects 4 2 1 3}

smi2EveForDisDisConRowStatus OBJECT-TYPE
                SYNTAX RowStatus
                MAX-ACCESS read-create
                STATUS current
                DESCRIPTION
                "This object indicates the status of this entry."
::= {smi2TranslatedObjects 4 2 1 4}

END
```

A.4 OUTPUT TEXTUAL-CONVENTIONS MACROS

```
-- ******************************************************
--              Example Textual Conventions Module for
--       CCITT Rec. X.721 | ISO/IEC 10165-2 : 1992
--             Definition of Management Information (DMI)
--               Translated into Internet SNMPv2 MIB
-- ******************************************************

MIBsmi2ConventionsIncrement1 DEFINITIONS ::= BEGIN
IMPORTS

ChoiceOfIntOrOID
        FROM IIMComibtransConventions

smi2TranslatedModules
        FROM MIBsmi2ComplianceIncrement1

MODULE-IDENTITY
        FROM SNMPv2-SMI;
```

-- Module Identity Macro

```
mibsmi2ConventionsIncrement1 MODULE-IDENTITY
                LAST-UPDATED "9310150000Z"
                ORGANIZATION "Network Management Forum"
                CONTACT-INFO
                "AIII Subteam
                1201 Mt. Kemble Avenue
                Morristown, NJ 07960-6628 USA
                Tel: +1 201-425-1900
                Fax: +1 201-425-1515
                Email: iimc@thumper.bellcore.com"
                DESCRIPTION
                "This module provides textual conventions for a partial translation of CCITT Rec.
                X.721|ISO/IEC 10165-2:1992, Definition of Management Information (DMI).

                WARNING:
                Currently, this module contains textual conventions used by the translated DMI
                alarmRecord managed object class and processingErrorAlarm notification.
                This is intended to illustrate the translation procedures specified by
                IIMCOMIBTRANS. The definitions contained in this module (including all object
                identifer assignments) are intended only as an illustration, and are non-
                normative. CONFORMANCE CANNOT BE CLAIMED TO THIS MODULE."
::= { smi2TranslatedModules 3 1 }

-- The following textual convention is for illustration. In a
-- full translation of smi2, many other textual conventions
-- would be generated for reused attributes such as
-- LoggingTime, ManagedObjectClass, ManagedObjectInstance, and
-- EventType, as well as generic state/relationship attributes.

Smi2SpecificIdentifier ::= TEXTUAL-CONVENTION
        STATUS current
        DESCRIPTION
        "SpecificIdentifier ::= CHOICE {OBJECT IDENTIFIER, INTEGER}"
SYNTAX ChoiceOfIntOrOID

END
```

A.5 OUTPUT MODULE-COMPLIANCE MACRO

```
-- ******************************************************
--                  Example Compliance Module for
--       CCITT Rec. X.721 | ISO/IEC 10165-2 : 1992
--              Definition of Management Information (DMI)
--                Translated into Internet SNMPv2 MIB
-- ******************************************************

MIBsmi2ComplianceIncrement1 DEFINITIONS ::= BEGIN
IMPORTS
        iimcIIMCOMIBTRANS
                FROM IimcAssignedOIDs
                {iso(1) member-body(2) 124 forum(360501) iimcManual(15) iimcModule(0) 1}
```

```
        MODULE-COMPLIANCE, MODULE-IDENTITY
                FROM SNMPv2-CONF;


-- OBJECT IDENTIFIER Assignments

smi2Translated                  -- output registry (place holder)
        OBJECT IDENTIFIER ::= {?}
smi2TranslatedModules  -- output registry for MODULEs
        OBJECT IDENTIFIER ::= {smi2Translated 2}
smi2TranslatedObjects   -- output registry for OBJECT CLASS
        OBJECT IDENTIFIER ::= {smi2Translated 3}
smi2TranslatedNotifications -- output registry for NOTIFICATION
        OBJECT IDENTIFIER ::= {smi2Translated 10}

-- Module Identity Macro

mibsmi2CompilanceIncrement1 MODULE-IDENTITY
                LAST-UPDATED "9310150000Z"
                ORGANIZATION "Network Management Forum"
                CONTACT-INFO
                "AIII Subteam
                1201 Mt. Kemble Avenue
                Morristown, NJ 07960-6628 USA
                Tel: +1 201-425-1900
                Fax: +1 201-425-1515
                Email: iimc@thumper.bellcore.com"
                DESCRIPTION
                "This module provides a MODDULE-COMPLIANCE macro for the partial translation
                of CCITT Rec. X.721|ISO/IEC  10165-2:1992, Definition of Management Information
                (DMI).

                WARNING:
                The definitions contained in this module (including all object identifer
                assignments and conformance macros) are intended only as an illustration,
                and are non-normative. CONFORMANCE CANNOT BE CLAIMED TO THIS
                MODULE."
::= { smi2TranslatedModules 0 0 1 }


mibsmi2ComplianceStatementIncrement1 MODULE-COMPLIANCE
                STATUS current
                DESCRIPTION
                "BEGINPARSE
                I. INPUTS TO THE TRANSLATION PROCESS

                The input class subtree and associated output class subtree for this module are:
                        INPUT: smi2MObjectClass
                        OUTPUT:         smi2TranslatedObjects (place holder)

                The associated input and output notification subtrees for this module are:
                        INPUT: smi2Notification
                        OUTPUT:         smi2TranslatedNotifications
```

(place holder)

The associated input and output module subtrees for this module are:
INPUT: asn1Module
OUTPUT: smi2TranslatedModules (place holder)

The associated input and output ANY discriminator subtrees for this module are:
Not Applicable.

The document(s) providing input specifications for the translated module(s) are:
CCITT Rec. X.721|ISO/IEC 10165-2:1992,
Definition of Management Information (DMI)

The <naming prefix> used to generate the translated module(s) is:
smi2

II. EXCEPTIONS TO THE TRANSLATION PROCESS

Following are the cases where a 'natural key' is substituted for the default (arbitrary integer index):
Not Applicable.

Following are cases for which output syntax differs from the specified mapping for the input syntax:
1. smi2AlaRecCorrelatedNotificationsTable
   The CorrelatedNotifications syntax has been flattened to reduce table nesting.
2. smi2AlaRecThresholdInfoTable
   The ThresholdLevelInd syntax has been flattened to reduce table nesting.

Following are cases in which the specified translation procedure cannot accommodate the values permitted by the base specification:
1. smi2AlaRecPerceivedSeverity indeterminate(0) mapped to 32767
2. smi2AlaRecTrendIndication lessSevere(0) mapped to 32767

Following are cases for which the translation procedure would result in duplicate labels:
Not Applicable.

Human judgement has identified that each of the following ANY DEFINED BY constructs resolve to a single option, treated as if that syntactical option were referenced directly:
Not Applicable.

Following are cases for which the specified translation would result in duplicated values (e.g., for named INTEGER types), and the alternative values used:
Not Applicable.

Following are other ways in which the translated MIB module does not comply with stated translation procedures:

This translation is incomplete, dealing with a subset input specification, and is provided for illustration purposes only.

```
                ENDPARSE"

                REFERENCE
                "CCITT Rec. X.721|ISO/IEC 10165-2:1992"
                MODULE MIBsmiSNMPv2
                GROUP alarmRecord
                DESCRIPTION "The alarmRecord group is conditionally mandatory for systems which
                implement the processingErrorAlarm notification and/or log record specified by CCITT
                Rec. X.721|ISO/IEC 10165-2:1992."
                MODULE MIBsmiSNMPv1
                GROUP alarmRecord
                DESCRIPTION
                "The alarmRecord group usage is the same for both SNMPv1 and SNMPv2
                frameworks."
                -- other groups listed here in a full translation
::= {smi2TranslatedModules 0 1 1}
END
```

A.6 OUTPUT AGENT-CAPABILITIES MACRO

```
-- Example Agent Capabilities Macro

-- The following macro is an example of an AGENT-CAPABILITIES
-- macro that a vendor might supply with an implementation
-- of this translated MIB.

exampleAgent AGENT-CAPABILITIES
                PRODUCT-RELEASE "Specific Agent Product Release"
                STATUS current
                DESCRIPTION "This implementation complies with any textual conventions specified
                by IIMComibtransConventions and used by MIBsmi2SNMPv2. Specific Agent Product
                Description..."
                -- SUPPORTS IIMComibtransConventions
                -- INCLUDES { no groups, only conventions }
                SUPPORTS MIBsmi2SNMPv2
                INCLUDES { alarmRecord }
                -- VARIATION clauses are included as needed here
::= { 0 0 0 }

-- In an actual implementation, the this would be a unique
-- OID value assigned to the agent implementation by the vendor
```

A.7 APPLICABILITY OF MOCS

Currently, [7] does not contain a MOCS proforma corresponding to the DMI input MIB. Instead, draft MOCS proformas corresponding to DMI objects are being progressed as addenda to the Systems Management Function base standards ISO10164-1,2,3,4,5,6,7. A supplier wishing to claim conformance to the translated DMI MIB could obtain MOCS proforma tables from the relevant draft addenda.

ANNEX B (NORMATIVE): SUPPORTING MODULES

B.1. SNMPV1 SUPPORT OBJECTS

```
IIMComibtransSptSNMPv1 DEFINITIONS ::= BEGIN
IMPORTS
        omibtransSpt
                FROM IIMComibtransSptCompliance
        Pointer, ClassInstancePointer, Parent, TableIndex
                FROM IIMComibtransConventions
        InstancePointer, TruthValue
                FROM SNMPv2-TC
        OBJECT-TYPE, internet
                FROM RFC1155-SMI;

-- The registration of this ASN.1 module is:
--      { omibtransSpt 0 1 }

-- ********************************************************
--
--      This module contains the following supportive tables:
--
--      • Child Table
--
--      • DestroyObject Table
--
--      • NameBinding Table.
--
--      • TableIndex Table.
--
--      • TrapSuppression Table
--
-- These tables comply with standard SNMPv2 textual
-- conventions and additional textual conventions
-- specified in IIMComibtransSptSNMPv2.
--
-- Note that OBJECT-GROUPs are registered using a different
-- registration structure than is used for translated MIBs.
--
-- ********************************************************

-- children OBJECT-GROUP
--      OBJECTS{ omibtransSptChild }
--      STATUS current
--      DESCRIPTION "See the description of omibtransSptChildTable below."
--      ::= { omibtransSpt 1 0}
```

```
omibtransSptChildTable OBJECT-TYPE
        SYNTAX SEQUENCE OF OmibtransSptChildEntry
        ACCESS not-accessible
        STATUS mandatory
        DESCRIPTION
        "Provides a means to locate class instances that are       immediately subordinate to a
        given class instance.  As such, provides a subset of the CMIS scoping capability."

::= { omibtransSpt  1 }

omibtransSptChildEntry OBJECT-TYPE
        SYNTAX OmibtransSptChildEntry
        ACCESS not-accessible
        STATUS mandatory
        DESCRIPTION
        "Entry of omibtransSptChildTable."
        INDEX {         omibtransSptParent,
                        omibtransSptChildIndex
                }
::= { omibtransSptChildTable 1 }

OmibtransSptChildEntry ::=
        SEQUENCE {
                omibtransSptParent      OBJECT IDENTIFIER,
                omibtransSptChildIndex  INTEGER,
                omibtransSptChild       OBJECT IDENTIFIER
                }

omibtransSptParent OBJECT-TYPE
        SYNTAX Parent
        ACCESS not-accessible
        STATUS mandatory
        DESCRIPTION
        "Supplies the ClassInstancePointer for a class entry."
::= { omibtransSptChildEntry 1 }

omibtransSptChildIndex OBJECT-TYPE
        SYNTAX TableIndex
        ACCESS not-accessible
        STATUS mandatory
        DESCRIPTION
        "Supplies unique table indices for different child entries."
::= { omibtransSptChildEntry 2 }

omibtransSptChild OBJECT-TYPE
        SYNTAX ClassInstancePointer
        ACCESS read-only
        STATUS mandatory
        DESCRIPTION
        "Holds ClassInstancePointers for child entries, i.e., class entries that are immediately
        subordinate to the parent class entry."
::= { omibtransSptChildEntry 3 }

-- ********************************************************
```

```
-- destruction OBJECT-GROUP
--         OBJECTS{ omibtransSptDestroyObject }
--         STATUS current
--         DESCRIPTION "See the description of omibtransSptDestroyObjectTable
--         below."
--         ::= { omibtransSpt 2 0}

omibtransSptDestroyObjectTable OBJECT-TYPE
        SYNTAX SEQUENCE OF
                OmibtransSptDestroyObjectEntry
        ACCESS not-accessible
        STATUS mandatory
        DESCRIPTION
        "Provides a means to delete (make not present) a scalar object or columnar entry.

        One use of this is to destroy an object corresponding to OPTIONAL syntax that is no
        longer wanted."
::= { omibtransSpt  2 }

omibtransSptDestroyObjectEntry OBJECT-TYPE
        SYNTAX OmibtransSptDestroyObjectEntry
        ACCESS not-accessible
        STATUS mandatory
        DESCRIPTION
        "Entry of omibtransSptDestroyObjectTable."
        INDEX { omibtransSptDestroyObjectIndex }
::= { omibtransSptDestroyObjectTable 1 }

OmibtransSptDestroyObjectEntry ::=
        SEQUENCE {
                omibtransSptDestroyObjectIndex
                        OBJECT IDENTIFIER,
                omibtransSptDestroyObject
        INTEGER
        }

omibtransSptDestroyObjectIndex OBJECT-TYPE
        SYNTAX Pointer
        ACCESS not-accessible
        STATUS mandatory
        DESCRIPTION
        "This object points to the scalar object or columnar entry for which destruction is
        desired."
::= { omibtransSptDestroyObjectEntry 1 }

omibtransSptDestroyObject OBJECT-TYPE
        SYNTAX TruthValue
        ACCESS read-write
        STATUS mandatory
        DESCRIPTION
        "When this value is set to true and omibtransSptDestroyObjectIndex points to an object
        for which destruction is permitted (e.g., a columnar object corresponding to OPTIONAL
```

```
        syntax) the referenced object is destroyed.  If destruction is not permitted, error-status of
        genErr is returned for SNMPv1 or inconsistentValue for SNMPv2.

        A value of false always results when reading this object."
::= { omibtransSptDestroyObjectEntry 2 }


-- ********************************************************

-- nameBinding OBJECT-GROUP
--         OBJECTS{ omibtransSptNameBinding }
--         STATUS current
--         DESCRIPTION "See the description of omibtransSptNameBindingTable
--         below."
--         ::= { omibtransSpt 3 0}

omibtransSptNameBindingTable OBJECT-TYPE
        SYNTAX SEQUENCE OF OmibtransSptNameBindingEntry
        ACCESS not-accessible
        STATUS mandatory
        DESCRIPTION
        "Provides a mechanism for specifying what NAME BINDING to use for manager-initiated
        creation of a class entry, should there be ambiguity.  It also allows discovery of the
        NAME BINDING in use for a given class entry."
::= { omibtransSpt  3 }

omibtransSptNameBindingEntry OBJECT-TYPE
        SYNTAX OmibtransSptNameBindingEntry
        ACCESS not-accessible
        STATUS mandatory
        DESCRIPTION
        "Entry of omibtransSptNameBindingTable."
        INDEX { omibtransSptNameBindingIndex }
::= { omibtransSptNameBindingTable 1 }

OmibtransSptNameBindingEntry ::= SEQUENCE {
        omibtransSptNameBindingIndex
                OBJECT IDENTIFIER,
        omibtransSptNameBinding
                OBJECT IDENTIFIER
        }

omibtransSptNameBindingIndex OBJECT-TYPE
        SYNTAX ClassInstancePointer
        ACCESS not-accessible
        STATUS mandatory
        DESCRIPTION
        "Provides the ClassInstancePointer to a class entry being created by a management
        station."
::= { omibtransSptNameBindingEntry 1 }

omibtransSptNameBinding OBJECT-TYPE
        SYNTAX OBJECT IDENTIFIER
        ACCESS read-write
```

```
        STATUS mandatory
        DESCRIPTION
        "Provides the OBJECT IDENTIFIER for the NAME BINDING that will be used when
        creating a new class instance.  For an existing class instance, returns the identifier of the
        NAME BINDING in use."
::= { omibtransSptNameBindingEntry 2 }

-- ********************************************************

-- tableIndex OBJECT-GROUP
--         OBJECTS{ omibtransSptTableIndexValue }
--         STATUS current
--         DESCRIPTION "See the description of omibtransSptTableIndexTable
--         below."
--         ::= { omibtransSpt 4 0}

omibtransSptTableIndexTable OBJECT-TYPE
        SYNTAX SEQUENCE OF OmibtransSptTableIndexEntry
        ACCESS not-accessible
        STATUS mandatory
        DESCRIPTION
        "Provides a class table or side table index for purposes of manager-initiated creation of
        rows in tables (i.e., new managed object instances or new values for multi-valued
        attributes).  Successive reads to this table return different values that are unique within
        the scope of the table within that agent.  Such values are assigned arbitrarily by the
        agent, so a manager should make no assumption about the sequence or magnitude of
        values returned."
::= { omibtransSpt  4 }

omibtransSptTableIndexEntry OBJECT-TYPE
        SYNTAX OmibtransSptTableIndexEntry
        ACCESS not-accessible
        STATUS mandatory
        DESCRIPTION
        "Entry of omibtransSptTableIndexTable."
        INDEX { omibtransSptTableIndexIndex }
::= { omibtransSptTableIndexTable 1 }

OmibtransSptTableIndexEntry ::=
        SEQUENCE {
        omibtransSptTableIndexIndex
                OBJECT IDENTIFIER,
        omibtransSptTableIndexValue
                OBJECT IDENTIFIER
        }

omibtransSptTableIndexIndex OBJECT-TYPE
        SYNTAX InstancePointer
        ACCESS not-accessible
        STATUS mandatory
        DESCRIPTION
        "Provides InstancePointer to the conceptual row for which a new entry will be created."
::= { omibtransSptTableIndexEntry 1 }
```

```
omibtransSptTableIndexValue OBJECT-TYPE
        SYNTAX OBJECT IDENTIFIER
        ACCESS read-only
        STATUS mandatory
        DESCRIPTION
        "Returns a TableIndex for the indicated conceptual table, to be used in creating a new
        entry.  Thus, this object typically returns different values each time it is read."
::= { omibtransSptTableIndexEntry 2 }

-- ********************************************************

-- trapSuppression OBJECT-GROUP
--         OBJECTS{ omibtransSptTrapSuppressionFlag }
--         STATUS current
--         DESCRIPTION "See the description of
--         omibtransSptTrapSuppressionTable below."
--         ::= { omibtransSpt 5 0}

omibtransSptTrapSuppressionTable OBJECT-TYPE
        SYNTAX SEQUENCE OF OmibtransSptTrapSuppressionEntry
        ACCESS not-accessible
        STATUS mandatory
        DESCRIPTION
        "Provides a means to selectively suppress issuance of traps under the SNMPv1
        framework."
::= { omibtransSpt  5 }

omibtransSptTrapSuppressionEntry OBJECT-TYPE
        SYNTAX OmibtransSptTrapSuppressionEntry
        ACCESS not-accessible
        STATUS mandatory
        DESCRIPTION
        "Entry of omibtransSptTrapSuppressionTable."
        INDEX {
                omibtransSptTrapSuppressionEnterprise,
                omibtransSptTrapSuppressionTrapNumber
                }
::= { omibtransSptTrapSuppressionTable 1 }

OmibtransSptTrapSuppressionEntry ::=
        SEQUENCE {
                omibtransSptTrapSuppressionEnterprise
                        OBJECT IDENTIFIER,
                omibtransSptTrapSuppressionTrapNumber
                        INTEGER,
                omibtransSptTrapSuppressionFlag
                        INTEGER
                }

omibtransSptTrapSuppressionEnterprise OBJECT-TYPE
        SYNTAX OBJECT IDENTIFIER
        ACCESS not-accessible
        STATUS mandatory
        DESCRIPTION
```

```
        "Supplies the Enterprise OBJECT IDENTIFIER for the trap of interest."
::= { omibtransSptTrapSuppressionEntry 1 }

omibtransSptTrapSuppressionTrapNumber OBJECT-TYPE
        SYNTAX INTEGER
        ACCESS not-accessible
        STATUS mandatory
        DESCRIPTION
        "Supplies the trap number (value of the TRAP-TYPE macro) for the trap of interest."
::= { omibtransSptTrapSuppressionEntry 2 }

omibtransSptTrapSuppressionFlag OBJECT-TYPE
        SYNTAX TruthValue
        ACCESS read-write
        STATUS mandatory
        DESCRIPTION
        "This object determines whether traps of a given type  will be issued or not. A value of
        true indicates a trap of the given type shall be suppressed."
::= { omibtransSptTrapSuppressionEntry 3 }

END
```

B.2. SNMPV2 SUPPORT OBJECTS

```
IIMComibtransSptSNMPv2 DEFINITIONS ::= BEGIN

IMPORTS
        omibtransSpt FROM IIMComibtransSptComplianceModule
        MODULE-IDENTITY, OBJECT-TYPE, NOTIFICATION-TYPE,
        Integer32, internet, snmpModules
                FROM SNMPv2-SMI
        InstancePointer, RowStatus, TruthValue
                FROM SNMPv2-TC
        ClassInstancePointer, Parent, TableIndex
                FROM IIMComibtransConventions
        OBJECT-GROUP
                FROM SNMPv2-CONF;

iimcomibtransSptSNMPv2 MODULE-IDENTITY
        LAST-UPDATED "9310150000Z"
        ORGANIZATION "Network Management Forum"
        CONTACT-INFO
        "AIII Subteam
        1201 Mt. Kemble Avenue
        Morristown, NJ 07960-6628 USA
        Tel: +1 201-425-1900
        Fax: +1 201-425-1515
        Email: iimc@thumper.bellcore.com"
        DESCRIPTION
        "This module facilitates mechanized translation of MIBs from GDMO to the SNMPv2
        framework.  It contains the following supportive tables:
                • Child Table
```

- DestroyObject Table
- NameBinding Table.
- TableIndex Table."

::= { omibtransSpt 0 2 }

-- Note that OBJECT-GROUPs are registered using a different
-- registration structure than is used for translated MIBs.

children OBJECT-GROUP
OBJECTS{ omibtransSptChild }
STATUS current
DESCRIPTION "See the description of omibtransSptChildTable below."
::= { omibtransSpt 1 0}

omibtransSptChildTable OBJECT-TYPE
SYNTAX SEQUENCE OF OmibtransSptChildEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Provides a means to locate class instances that are immediately subordinate to a given class instance. As such, provides a subset of the CMIS scoping capability."
::= { omibtransSpt 1 }

omibtransSptChildEntry OBJECT-TYPE
SYNTAX OmibtransSptChildEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Entry of omibtransSptChildTable."
INDEX { omibtransSptParent,
omibtransSptChildIndex
}
::= { omibtransSptChildTable 1 }

OmibtransSptChildEntry ::=
SEQUENCE {
omibtransSptParent ClassInstancePointer,
omibtransSptChildIndex TableIndex,
omibtransSptChild ClassInstancePointer
}

omibtransSptParent OBJECT-TYPE
SYNTAX ClassInstancePointer
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Supplies the ClassInstancePointer for a class entry."
::= { omibtransSptChildEntry 1 }

omibtransSptChildIndex OBJECT-TYPE
SYNTAX TableIndex
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION

```
        "Supplies unique table indices for different child entries."
::= { omibtransSptChildEntry 2 }

omibtransSptChild OBJECT-TYPE
        SYNTAX ClassInstancePointer
        MAX-ACCESS read-only
        STATUS current
        DESCRIPTION
        "Holds ClassInstancePointers for child entries, i.e., class entries that are immediately
        subordinate to the parent class entry."
::= { omibtransSptChildEntry 3 }

-- ********************************************************

destruction OBJECT-GROUP
        OBJECTS{ omibtransSptDestroyObject }
        STATUS current
        DESCRIPTION
        "See the description of omibtransSptDestroyObjectTable below."
::= { omibtransSpt 2 0}

omibtransSptDestroyObjectTable OBJECT-TYPE
        SYNTAX SEQUENCE OF
                OmibtransSptDestroyObjectEntry
        MAX-ACCESS not-accessible
        STATUS current
        DESCRIPTION
        "Provides a means to delete (make not present) a scalar object or columnar entry.

        One use of this is to destroy an object corresponding to OPTIONAL syntax that is no
        longer wanted."
::= { omibtransSpt 2 }

omibtransSptDestroyObjectEntry OBJECT-TYPE
        SYNTAX OmibtransSptDestroyObjectEntry
        MAX-ACCESS not-accessible
        STATUS current
        DESCRIPTION
        "Entry of omibtransSptDestroyObjectTable."
        INDEX { omibtransSptDestroyObjectIndex }
::= { omibtransSptDestroyObjectTable 1 }

OmibtransSptDestroyObjectEntry ::=
        SEQUENCE {
                omibtransSptDestroyObjectIndex
                        Pointer,
                omibtransSptDestroyObject
        TruthValue
        }

omibtransSptDestroyObjectIndex OBJECT-TYPE
        SYNTAX Pointer
        MAX-ACCESS not-accessible
        STATUS current
```

```
        DESCRIPTION
        "This object points to the scalar object or columnar entry for which destruction is
        desired."
::= { omibtransSptDestroyObjectEntry 1 }

omibtransSptDestroyObject OBJECT-TYPE
        SYNTAX TruthValue
        MAX-ACCESS read-write
        STATUS current
        DESCRIPTION
        "When this value is set to true and omibtransSptDestroyObjectIndex points to an object
        for which destruction is permitted (e.g., a columnar object corresponding to OPTIONAL
        syntax) the referenced object is destroyed.  If destruction is not permitted, error-status of
        genErr is returned for SNMPv1 or inconsistentValue for SNMPv2.

        A value of false always results when reading this object."
::= { omibtransSptDestroyObjectEntry 2 }


-- ********************************************************

nameBinding OBJECT-GROUP
        OBJECTS{ omibtransSptNameBinding }
        STATUS current
        DESCRIPTION "See the description of omibtransSptNameBindingTable
        below."
::= { omibtransSpt 3 0}

omibtransSptNameBindingTable OBJECT-TYPE
        SYNTAX SEQUENCE OF OmibtransSptNameBindingEntry
        MAX-ACCESS not-accessible
        STATUS current
        DESCRIPTION
        "Provides a mechanism for specifying what NAME BINDING to use for manager-initiated
        creation of a class entry, should there be ambiguity.  It also allows discovery of the
        NAME BINDING in use for a given class entry."
::= { omibtransSpt  3 }

omibtransSptNameBindingEntry OBJECT-TYPE
        SYNTAX OmibtransSptNameBindingEntry
        MAX-ACCESS not-accessible
        STATUS current
        DESCRIPTION
        "Entry of omibtransSptNameBindingTable."
        INDEX { omibtransSptNameBindingIndex }
::= { omibtransSptNameBindingTable 1 }

OmibtransSptNameBindingEntry ::= SEQUENCE {
        omibtransSptNameBindingIndex
                ClassInstancePointer,
        omibtransSptNameBinding
                OBJECT IDENTIFIER
        }
```

```
omibtransSptNameBindingIndex OBJECT-TYPE
        SYNTAX ClassInstancePointer
        MAX-ACCESS not-accessible
        STATUS current
        DESCRIPTION
        "Provides the ClassInstancePointer to a class entry being created by a management
        station."
::= { omibtransSptNameBindingEntry 1 }

omibtransSptNameBinding OBJECT-TYPE
        SYNTAX OBJECT IDENTIFIER
        MAX-ACCESS read-write
        STATUS current
        DESCRIPTION
        "Provides the OBJECT IDENTIFIER for the NAME BINDING that will be used when
        creating a new class instance. For an existing class instance, returns the identifier of the
        NAME BINDING in use."
::= { omibtransSptNameBindingEntry 2 }

-- ********************************************************

tableIndex OBJECT-GROUP
        OBJECTS{ omibtransSptTableIndexValue }
        STATUS current
        DESCRIPTION
        "See the description of omibtransSptTableIndexTable below."
::= { omibtransSpt 4 0}

omibtransSptTableIndexTable OBJECT-TYPE
        SYNTAX SEQUENCE OF OmibtransSptTableIndexEntry
        MAX-ACCESS not-accessible
        STATUS current
        DESCRIPTION
        "Provides a class table or side table index for purposes of manager-initiated creation of
        rows in tables (i.e., new managed object instances or new values for multi-valued
        attributes).  Successive reads to this table return different values that are unique within
        the scope of the table within that agent.  Such values are assigned arbitrarily by the
        agent, so a manager should make no assumption about the sequence or magnitude of
        values returned."
::= { omibtransSpt  4 }

omibtransSptTableIndexEntry OBJECT-TYPE
        SYNTAX OmibtransSptTableIndexEntry
        MAX-ACCESS not-accessible
        STATUS current
        DESCRIPTION
        "Entry of omibtransSptTableIndexTable."
        INDEX { omibtransSptTableIndexIndex }
::= { omibtransSptTableIndexTable 1 }

OmibtransSptTableIndexEntry ::=
        SEQUENCE {
        omibtransSptTableIndexIndex
                InstancePointer,
```

```
        omibtransSptTableIndexValue
                TableIndex
        }

omibtransSptTableIndexIndex OBJECT-TYPE
        SYNTAX InstancePointer
        MAX-ACCESS not-accessible
        STATUS current
        DESCRIPTION
        "Provides InstancePointer to the conceptual row for which a new entry will be created."
::= { omibtransSptTableIndexEntry 1 }

omibtransSptTableIndexValue OBJECT-TYPE
        SYNTAX TableIndex
        MAX-ACCESS read-only
        STATUS current
        DESCRIPTION
        "Returns a TableIndex for the indicated conceptual table, to be used in creating a new
        entry.  Thus, this object typically returns different values each time it is read."
::= { omibtransSptTableIndexEntry 2 }

END
```

B.3. TEXTUAL CONVENTIONS

```
IIMComibtransConventions DEFINITIONS ::= BEGIN

IMPORTS
        omibtransSpt
                FROM IIMComibtransSptComplianceModule
        InstancePointer, TruthValue
                FROM SNMPv2-TC;

iimcomibtransConventions MODULE-IDENTITY
        LAST-UPDATED "9310150000Z"
        ORGANIZATION "Network Management Forum"
        CONTACT-INFO
        "AIII Subteam
        1201 Mt. Kemble Avenue
        Morristown, NJ 07960-6628 USA
        Tel: +1 201-425-1900
        Fax: +1 201-425-1515
        Email: iimc@thumper.bellcore.com"
        DESCRIPTION
        "This module contains the textual conventions that support IIMCOMIBTRANS:

                • Pointer (a generic construct further refined below)

                • AnyDefinedBy
                • ActionTrigger
                • ActionStatus
                • BitString
```

- ChoiceOfIntOrOID
- ClassInstancePointer
- Complex
- GraphicString
- MultiComplex
- Parent
- Recursion
- RecursivelyNested
- TableIndex"

::= { omibtransSpt 0 3 }

Pointer ::= TEXTUAL-CONVENTION
STATUS current
DESCRIPTION
"An object following this convention, when present, has one of two usages:

1) It follows the ObjectInstance textual convention.
2) It contains the (entire) name of a scalar MIB object
or the (entire) name of a conceptual column entry.

A Pointer is always defined as a conceptual column entry. Absence of this entry for a particular conceptual row indicates the absence of something to point to, i.e., end of a pointer chain."
SYNTAX OBJECT IDENTIFIER

ActionTrigger ::= TEXTUAL-CONVENTION
STATUS current
DESCRIPTION
"This convention provides a mechanism for triggering simulated ACTIONs in translated MIBs. A columnar entry following this convention controls the simulation of a particular ACTION for a particular class entry. An ActionTrigger object is always accompanied by a corresponding object that observes the ActionStatus textual convention.

The ActionTrigger textual convention is a refinement of the TestAndIncr convention. To update this object requires that usual TestAndIncr requirements be met, also, that the associated ActionStatus object does not hold the value inProgress. If these prerequisites are not met, attempt to update will result in error-status of genErr for SNMPv1 or inconsistentValue for SNMPv2.

Successfully setting the ActionTrigger initiates emulation of the translated ACTION, at which point the value of the ActionStatus object becomes inProgress. The value of the ActionTrigger however does not increment - nor does the SetRequest operation terminate - until the simulated ACTION ends due to completion or failure. At that point, the ActionTrigger increments and the value of ActionStatus indicates the outcome.

An ActionTrigger object has DEFVAL of 0."
SYNTAX TestAndIncr

ActionStatus ::= TEXTUAL-CONVENTION
STATUS current
DESCRIPTION

"This convention provides a mechanism for reporting status of simulated ACTIONs in translated MIBs. An ActionStatus object is always accompanied by a corresponding object that observes the ActionTrigger textual convention.

The ActionTrigger textual convention explains defines under which ActionStatus changes value. The values ActionStatus can assume are:

The values ActionStatus can assume are:

-2: inProgress.
-1: noProblemDetected.
Other non-negative: specificError.

If the ACTION being emulated is confirmed, the value noProblemDetected indicates successful completion of the n-1th operation, where the ActionTrigger is of current value n. If the ACTION being emulated is unconfirmed, this value (noProblemDetected) indicates no emulated ACTION is in progress, or the status of the last action is not known.

An ActionStatus object has DEFVAL of noProblemDetected."
SYNTAX Integer32

AnyDefinedBy ::= TEXTUAL-CONVENTION
STATUS current
DESCRIPTION
"This convention is a Pointer (as defined above) that points to an SNMP structure mapping to the ASN.1 ANY DEFINED BY construct, e.g., of the form

```
Y       ::= SEQUENCE            {
            <Id>    OBJECT IDENTIFIER,   -- or perhaps INTEGER
            <x>     ANY DEFINED BY <id>
                    }
```

The structure pointed to corresponds to the value <x>. The value <Id> is not directly mapped. The value pointed to will be in the same row as the AnyDefinedBy object if it is of simple syntax, i.e., is expressed as a single columnar object. Otherwise it is complex and is expressed in its own side table.

Retrieval of the values associated with the columnar objects of textual convention AnyDefinedBy is a two (or more) stage process. First, the Internet name contained in the object with the AnyDefinedBy convention is retrieved. Then the retrieved Pointer shall be used to retrieve the actual value(s) desired. If the Pointer references an object that represents a simple syntax, then only the object referenced need be retrieved. If the Pointer references an object that represents complex syntax, then all columnar objects of all rows associated with the complex syntax will have to be retrieved.

An AnyDefinedBy pointer within an active or notInService row shall always point to a valid syntactical option that itself is active or notInService. Attempt to point such a AnyDefinedBy object to invalid syntax results in error-status of badValue for SNMPv1 and inconsistentValue for SNMP. However, the pointer may be absent if the ANY DEFINED BY construct is part of optional syntax, or the syntax is not supported by the implementation.

When such an object is successfully changed, any syntax previously pointed to (directly or indirectly) is automatically deleted by the agent."
SYNTAX Pointer

BitString ::= TEXTUAL-CONVENTION
STATUS current
DESCRIPTION
"This convention is equivalent to the ASN.1 BIT STRING type, with the same rules for bit ordering and padding."
SYNTAX OCTET STRING

ChoiceOfIntOrOID ::= TEXTUAL-CONVENTION
STATUS current
DESCRIPTION
"This convention allows an ASN.1 choice of Integer32 or OBJECT IDENTIFIER to be expressed as a single SNMP object.

If the value to be expressed is of type Integer32, that value is expressed as an OBJECT IDENTIFIER with a prefix of either omibtransSptNonnegativeInteger or omibtransSptNegativeInteger, depending whether the value is nonnegative or not. The integer's value is carried in the arc immediately following the prefix. If the value is negative, it is understood that the value of the trailing arc is to be negated in arriving at the actual value.

If the OBJECT IDENTIFIER begins with any prefix other than these two special values, it is understood tobe a (normal) OBJECT IDENTIFIER choice."
SYNTAX OBJECT IDENTIFIER

omibtransSptNonnegativeInteger OBJECT IDENTIFIER ::= { omibtransSpt 6 }
omibtransSptNegativeInteger OBJECT IDENTIFIER ::= { omibtransSpt 7 }

ClassInstancePointer ::= TEXTUAL-CONVENTION
STATUS current
DESCRIPTION
"This value contains one of:
- An InstancePointer to an occurrence of a (translated) class table.
- An InstancePointer that references an occurrence of a tabular object group that was not arrived at through IIMCOMIBTRANS translation.
- An Internet name (OBJECT IDENTIFIER) that references the first object (object with the lowest lexicographic ordering) of a non tabular object group.

In every case, a ClassInstancePointer references an instance of an object group."
SYNTAX Pointer

Complex ::= TEXTUAL-CONVENTION
STATUS current
DESCRIPTION
"A Complex object is an invariant read-only pointer to a side table containing a single row of objects per entry in the superior table. Those objects represent translation of syntax (i.e., CHOICE, SEQUENCE, SET) that is complex but not multiply occuring.

An object of this type is not-accessible. Its constant pointer value (name of the side table) appears in its DEFVAL clause."
SYNTAX OBJECT IDENTIFIER

GraphicString ::= TEXTUAL-CONVENTION
STATUS current
DESCRIPTION
"This convention is equivalent to the ISO/CCITT GraphicString type except the string is tagged as an OCTET STRING rather than a GraphicString."
SYNTAX OCTET STRING

MultiComplex ::= TEXTUAL-CONVENTION
STATUS current
DESCRIPTION
"A MultiComplex object is an invariant read-only pointer to a side table that may contain multiple rows of objects per entry in the superior table. Those rows represent translation of syntax (i.e., SEQUENCE OF or SET OF) that is both complex and multiply occuring.

An object of this type is not-accessible. Its constant pointer value (name of the side table) appears in its DEFVAL clause."
SYNTAX OBJECT IDENTIFIER

Parent ::= TEXTUAL-CONVENTION
STATUS current
DESCRIPTION
"This convention is used to specify the superior class entry instance when creating a subordinate class entry. It may be queried to determine the superior class instance, if any, for an existing class instance.

If a Parent pointer is not populated for a particular class entry, that object is considered to be subordinate to the local root, e.g., the internet system.

The Parent pointer of a historical entry in a class table has the value zero (0)."
SYNTAX Pointer

Recursion ::= TEXTUAL-CONVENTION
STATUS current
DESCRIPTION
"This convention is a refinement of the Pointer convention for pointing to nested recursive syntax. One such column shall be defined per side table if that table represents syntax that is directly or indirectly recursive. For tables in which it is defined, one corresponding RecursivelyNested column shall also be defined.

The recursion pointer points to the next (lower) level of nested syntax (absence of a pointer indicates end of chain). It may point to a different entry in the same table (direct recursion) or to another table (indirect recursion). The innermost level of syntax may be in a non-recursive side table, i.e., one that does not have a Recursion pointer.

A Recursive pointer within an active or notInService row shall always point to a valid syntactical option that itself is active or notInService. Attempt to point such a Recursive object to invalid syntax results in error-status of badValue for SNMPv1 and inconsistentValue for SNMPv2.

```
        When such an object is successfully changed, any syntax previously pointed to (directly
        or indirectly) is automatically deleted by the agent."
SYNTAX Pointer

RecursivelyNested ::= TEXTUAL-CONVENTION
        STATUS          current
        DESCRIPTION
        "This convention is used to flag whether a conceptual row represents nested recursive
        syntax, i.e., whether this row is pointed to by an object of type Recursion.  Columns
        following this convention are present only in rows that also have Recursive columns.
        This value is maintained by the agent, thus is read-only."
SYNTAX TruthValue

TableIndex ::= TEXTUAL-CONVENTION
        STATUS          current
        DESCRIPTION
        "This convention is used to identify an Internet columnar object that represents the index
        for a conceptual entry in a table.  It may have no particular meaning other than to
        discriminate between entries of the row.

        An object using this textual convention shall be not-accessible."
SYNTAX Integer32

END
```

B.4. MODULE-COMPLIANCE STATEMENT FOR SUPPORT OBJECTS

```
IIMComibtransSptCompliance DEFINITIONS ::= BEGIN
IMPORTS
        iimcIIMCOMIBTRANS
        FROM IimcAssignedOIDs
        {iso(1) member-body(2) 124 forum(360501) iimcManual(15) iimcModule(0) 1}

        MODULE-COMPLIANCE
        FROM SNMPv2-SMI;

iimcomibtransSptCompliance MODULE-IDENTITY
        LAST-UPDATED "9310150000Z"
        ORGANIZATION "Network Management Forum"
        CONTACT-INFO
        "AIII Subteam
        1201 Mt. Kemble Avenue
        Morristown, NJ 07960-6628 USA
        Tel: +1 201-425-1900
        Fax: +1 201-425-1515
        Email: iimc@thumper.bellcore.com"
        DESCRIPTION
        "This module defines compliance to the IIMCOMIBTRANS
        support MIB."
::= { omibtransSpt 0 0 0 }

omibtransSpt OBJECT IDENTIFIER ::=
```

```
{iso(1) member-body(2) 124 forum(360501) iimcManual(15) iimcModule(0) 5}

iimcomibtransSptComplianceStatement MODULE-COMPLIANCE
        STATUS current
        DESCRIPTION
        "This statement defines compliance to IIMComibtransSptSNMPv1 and
        IIMComibtransSptSNMPv2. Both modules provide support for MIB modules translated
        from ISO/CCITT systems management to SNMP according to procedures specified in
        IIMCOMIBTRANS. An implementation claiming conformance to any of these support
        objects shall conform to IIMComibtransSptSNMPv1 and/or IIMComibtransSptSNMPv2,
        depending on the SNMP framework(s) supported."

        MODULE IIMComibtransSptSNMPv1
                GROUP child
                        DESCRIPTION
                        "This group is unconditionally optional."
                GROUP destruction
                        DESCRIPTION
                        "This group is conditionally mandatory if the translated MIB that is
                        supported includes scalar types that correspond to OPTIONAL syntax."
                GROUP nameBinding
                        DESCRIPTION
                        "This group is conditionally mandatory if, for the translated MIB
                        supported, there is ever ambiguity about what NAME BINDING might
                        apply for any combination of containing and contained object classes."
                GROUP tableIndex
                        DESCRIPTION
                        "This group is unconditionally optional."
                GROUP trapSuppression
                        DESCRIPTION
                        "This group is unconditionally optional."
        MODULE IIMComibtransSptSNMPv2
                GROUP child
                        DESCRIPTION
                        "This group is unconditionally optional."
                GROUP destruction
                        DESCRIPTION
                        "This group is conditionally mandatory if the translated MIB that is
                        supported includes scalar types that correspond to OPTIONAL syntax."
                GROUP nameBinding
                        DESCRIPTION
                        "This group is conditionally mandatory if, for the translated MIB
                        supported, there is ever ambiguity about what NAME BINDING might
                        apply for any combination of containing and contained object classes."
                GROUP tableIndex
                        DESCRIPTION
                        "This group is unconditionally optional."
::= { omibtransSpt 0 0 1 }

END
```

ANNEX C: GLOSSARY

ASN.1	Abstract Syntax Notation One
CCITT	Consultative Committee on Telephony and Telegraphy
CMIP	Common Management Information Protocol
CMIS	Common Management Information Service
DN	Distinguished Name
GDMO	Guidelines for the Definition of Managed Objects
GNMP	Government Network Management Profile
IIMC	ISO/CCITT and Internet Management Coexistence
ISO	International Standards Organization
MCS	Management Conformance Statement
MIB	Management Information Base
MOCS	Managed Object Conformance Statement
NMF	Network Management Forum
NVT	Network Virtual Terminal
OID	Object Identifier
OSI	Open Systems Interconnection
PDU	Protocol Data Unit
RDN	Relative Distinguished Name
RFC	Request For Comments
SMI	Structure of Management Information
SNMP	Simple Network Management Protocol
SNMPv1	Simple Network Management Protocol Version 1
SNMPv2	Simple Network Management Protocol Version 2
TCP/IP	Transmission Control Protocol/Internetwork Protocol

ANNEX D: REFERENCES

1) CCITT Recommendation X.700, Management Framework Definition for Open Systems Interconnection (OSI).

ISO/IEC 7498-4: 1989, Information Processing Systems — Open Systems Interconnection - Basic Reference Model Part 4 — Management Framework.

2) ISO/IEC 8824: Information Technology — Open System Interconnection — Specification of Abstract Syntax Notation One (ASN.1),1990.

3) CCITT Recommendation X.209 (1988), Specification of basic encoding rules for abstract syntax notation one (ASN.1).

ISO/IEC 8825: 1990, Information Technology — Open System Interconnection — Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1).

4) CCITT Recommendation X.710, (1991), Common Management Information Service Definition for CCITT Applications.

ISO/IEC 9595: 1991, Information Technology — Open System Interconnection — Common Management Information Service Definition.

5) CCITT Recommendation X.711 | ISO/IEC 9596-1: 1991, Information Technology — Open Systems Interconnection — Common Management Information Protocol — Part 1: Specification.

6) CCITT Recommendation X.720 (1992) | ISO/IEC 10165-1: 1992, Information Technology — Open Systems Interconnection — Structure of Management Information — Part 1: Management Information Model.

7) CCITT Recommendation X.721 (1992) | ISO/IEC 10165-2: 1992, Information Technology — Open Systems Interconnection — Structure of Management Information — Part 2: Definition of Management Information.

8) CCITT Recommendation X.721 (1992) | ISO/IEC 10165-4: 1992, Information Technology — Open Systems Interconnection — Structure of Management Information — Part 4: Guidelines for the Definition of Managed Objects.

9) CCITT Recommendation X.723 (1993) | ISO/IEC 10165-6: 1993, Information Technology — Open Systems Interconnection — Structure of Management Information — Part 6: Requirements and Guidelines for Implementation Conformance Statement Proformas associated with OSI Management.

10) RFC1155, M. Rose and K. McCloghrie, Structure and Identification of Management Information for TCP/IP based internets, May 1990.

11) RFC1157, J.D. Case, M.S. Fedor, M.L. Schoffstall,C. Davin, Simple Network Management Protocol (SNMP), May 1990.

12) RFC1212, M. Rose, K. McCloghrie — Editors, Concise MIB Definitions, March 1991.

13) RFC1213, K. McCloghrie and M. Rose — Editors, Management Information Base for Network Management of TCP/IP-based internets: MIB-II, March 1991.

14) RFC1215, M. Rose - Editor, A convention for Defining Traps for use with the SNMP, March 1991.

15) RFC1441, J.D. Case, K. McCloghrie, M.T. Rose, S.L.Waldbusser, Introduction to version 2 of the Internet-standard Network Management Framework, April 1993.

16) RFC1442, J.D. Case, K. McCloghrie, M.T. Rose, S.L.Waldbusser, Structure of Management Information for version 2 of the Simple Network Management Protocol (SNMPv2), April 1993.

17) RFC1443, J.D. Case, K. McCloghrie, M.T. Rose, S.L.Waldbusser, Textual Conventions for version 2 of the Simple Network Management Protocol (SNMPv2), April 1993.

18) RFC1444, J.D. Case, K. McCloghrie, M.T. Rose, S.L.Waldbusser, Conformance Statements for SNMPv2, April 1993.

19) RFC1452, J.D. Case, K. McCloghrie, M.T. Rose, S.L.Waldbusser, Coexistence between version 1 and version 2 of the Internet Network Management Framework, April 1993.

20) Network Management Forum: Forum 029, Translation of Internet MIB-II (RFC 1213) to ISO/CCITT GDMO MIB, Issue 1.0, October 1993.

21) Network Management Forum: Forum 028, ISO/CCITT to Internet Management Proxy, Issue 1.0, 1993.

22) Network Management Forum: Forum 027, ISO/CCITT to Internet Management Security, Issue 1.0, October 1993.

23) Network Management Forum: Forum 026, Translation of Internet MIBs to ISO/CCITT GDMO MIBs, Issue 1.0, October 1993.

24) NM Forum and X/Open, ISO/CCITT and Internet Management: Coexistence and Interworking Strategy, Issue 1.0, October, 1992.

25) Federal Information Processing Standards Publication 179 — Government Network Management Profile v1.0, December 1992.