

**A Recommended Amendment to TS 3.2 - 1996
of the Joint Committee on the NTCIP
for Joint Adoption by AASHTO, ITE, and NEMA**

**NEMA Standards Publication TS 3.2-1996
– Amendment 1**

*National Transportation Communications for ITS Protocol
(NTCIP)
Simple Transportation Management Framework*

Draft Version 98.01.08 November 2, 1998

This is a draft document, which is distributed for review and comment purposes only. You may reproduce and distribute this document within your organization, but only for the purposes of and only to the extent necessary to facilitate review and comment to the **NTCIP Coordinator**. Please ensure that all copies reproduced or distributed bear this legend. This document contains information that is subject to change.

Published by

American Association of State Highway and Transportation Officials (AASHTO)
444 North Capitol St., N.W., Suite 249
Washington, D.C. 20001

Institute of Transportation Engineers (ITE)
525 School St., S.W., Suite 410
Washington, D.C. 20024-2797

National Electrical Manufacturers Association (NEMA)

© Copyright 1998 by the National Electrical Manufacturers Association, and by the American Association of State Highway and Transportation Officials, and by the Institute of Transportation Engineers. All rights, including translation into other languages, reserved under the Universal Copyright Convention, the Berne Convention or the Protection of Literary and Artistic Works, and the International and Pan American Copyright Conventions. Do not copy without written permission of either NEMA, AASHTO, or ITE.

TS 3.2 - 1996 Amendment 1
Page 2

1300 N. 17th Street, Suite 1847
Rosslyn, Virginia 22209-3801

MIB DISCLAIMER

- Add the following paragraphs after the title page of the document:

Joint AASHTO, ITE, NEMA Copyright
and
NTCIP Management Information Base Distribution
NOTICE

© Copyright 1998 by the American Association of State Highway and Transportation Officials (AASHTO), the Institute of Transportation Engineers (ITE), and the National Electrical Manufacturers Association (NEMA). All intellectual property rights, including, but not limited to, the rights of reproduction, translation and display are reserved under the laws of the United States of America, the Universal Copyright Convention, the Berne Convention, and the International and Pan American Copyright Conventions. Except as provided, you may not copy these materials without written permission from either AASHTO, ITE or NEMA. Use of these materials does not give you any rights of ownership or claim of copyright in or to these materials.

To the extent and in the limited event these materials are distributed by AASHTO/ITE/NEMA in the form of a Management Information Base ("MIB"), AASHTO/ITE/NEMA extends the following permissions:

- (i) you may make and/or distribute unlimited copies (including derivative works) of the MIB, including copies for commercial distribution, provided that (a) each copy you make and/or distribute contains this Notice and (b) each derivative work of the MIB uses the same module name followed by "-", followed by your Internet Assigned Number Authority (IANA)-assigned enterprise number;
- (ii) use of the MIB is restricted in that the syntax field may be modified only to reflect a more restrictive subrange or enumerated values;
- (iii) the description field may be modified but only to the extent that: (a) only those bit values or enumerated values that are supported are listed; and (b) the more restrictive subrange is expressed.

These materials are delivered "AS IS" without any warranties as to their use or performance.

AASHTO/ITE/NEMA AND THEIR SUPPLIERS DO NOT WARRANT THE PERFORMANCE OR RESULTS YOU MAY OBTAIN BY USING THESE MATERIALS. AASHTO/ITE/NEMA AND THEIR SUPPLIERS MAKE NO WARRANTIES, EXPRESS OR IMPLIED, AS TO NONINFRINGEMENT OF THIRD PARTY RIGHTS, MERCHANTABILITY, OR FITNESS FOR ANY PARTICULAR PURPOSE. IN NO EVENT WILL AASHTO, ITE OR NEMA OR THEIR SUPPLIERS BE LIABLE TO YOU OR ANY THIRD PARTY FOR ANY CLAIM OR FOR ANY CONSEQUENTIAL, INCIDENTAL OR SPECIAL DAMAGES, INCLUDING ANY LOST PROFITS OR LOST SAVINGS, ARISING FROM YOUR REPRODUCTION OR USE OF THESE MATERIALS, EVEN IF AN AASHTO, ITE, OR NEMA REPRESENTATIVE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Some states or jurisdictions do not allow the exclusion or limitation of incidental, consequential or special damages, or the exclusion of implied warranties, so the above limitations may not apply to you.

Use of these materials does not constitute an endorsement or affiliation by or between AASHTO, ITE, or NEMA and you, your company, or your products and services.

If you are unwilling to accept the foregoing restrictions, you should immediately return these materials.

NTCIP is a trademark of AASHTO/ITE/NEMA.

1.2 REFERENCES

- Add the following paragraphs to the top of this clause:

For approved revisions, contact:

NTCIP Coordinator
National Electrical Manufacturers Association
1300 North 17th Street, Suite 1847
Rosslyn, VA 22209-3801

For draft revisions, which are under discussion by the relevant NTCIP Working Group, and recommended revisions of the NTCIP Joint Committee, visit the World Wide Web at <http://www.ntcip.org>.

1.2.1 Normative References

- Replace the references to the Internet RFCs with:

IAB STD 15 (RFC 1157) *Simple Network Management Protocol*, J. Case, M. Fedor, M. Schoffstall, J. Davin, May 1990

IAB STD 16 (RFC 1155) *Structure and Identification of Management Information for TCP/IP based Internets*, M. Rose, K. McCloghrie, May 1990, (RFC 1212) *Concise MIB Definitions*, M. Rose and K. McCloghrie, March 1991

1.4 ABBREVIATIONS AND ACRONYMS

- Delete the *PER* acronym as it is no longer referenced.

- Add the following acronym to the list.

OER —Octet Encoding Rules, a variation of BER developed for use on low bandwidth communications links, specified in this Standards Publication.

3.2 THE NEMA AUTHORITY

- Change the words "*mgmt*", "*experimental*", and "*private*" to "*nemaMgmt*", "*nemaExperimental*", and "*nemaPrivate*" in Figure 3-1.

3.3 STRUCTURE AND IDENTIFICATION OF MANAGEMENT INFORMATION

- Modify the words "*mgmt*", "*experimental*", and "*private*" to "*nemaMgmt*", "*nemaExperimental*", and "*nemaPrivate*" throughout the Clause when they reference to the nodes under the *nema* node.

SECTION 4

- Replace the term "*TMIB*" with the term "*TMIB-II*" throughout this Section.

- *Modify the first paragraph to read:*

The Transportation MIB version 2 (TMIB-II) provides a framework for organizing and identifying information in locating objects for transportation specific equipment. The TMIB-II is located under the *nema* node of the global object tree. This MIB is controlled by NEMA Transportation Management Systems and Associated Control Devices Section (3-TS). NEMA 3-TS may designate subnodes of the TMIB-II to be administered by other groups or organizations.

4.1.1 Byte and UByte Types

- *Change the lower limit of the Byte from -127 to -128.*

4.1.2 Short and UShort Types

- *Change the lower limit of the Short from -32767 to -32768.*

4.1.3 Long and ULong Types

- *Change the lower limit of the Long from -2147483647 to -2147483648*

4.1.4 EntryStatus Type

- *Change the Clause title to "ConfigEntryStatus Type".*

- *Replace all of the text in this Clause with the following:*

The *EntryStatus* Type, as defined in TMIB(-I), has been replaced with the *ConfigEntryStatus* type in TMIB-II. The *ConfigEntryStatus* type shall be used to manage the *dynObjDef*(Table) that allows new rows to be created by management applications running remotely. For each row in the *dynObjConfigTable* there shall be a columnar object that is defined with a SYNTAX of *ConfigEntryStatus* (e.g., see dynamic object configuration table in 4.2). *ConfigEntryStatus* shall be an enumerated INTEGER that can have one of three values: *valid*, *underCreation* and *invalid*.

Other objects in the row shall have operations limited by the current value of the *EntryStatus* object in the row. The meaning of the values is as follows:

If the status object is *invalid* then the information in the corresponding rows of the *dynObjDef*(Table) with the same index *dynObjNumber* shall be considered undefined. Setting the status object to *invalid* has the effect of invalidating the corresponding rows. It is implementation specific whether the agent clears the values contained in an invalidated row, de-allocates the memory associated with invalidated rows, or simply leaves the last values within the row. When in the *invalid* state, the agent shall reject any request to go to the *valid* state.

The *underCreation* state shall indicate that the memory for the corresponding rows of the *dynObjDef*(Table) with the same index *dynObjNumber* is allocated, but may contain some invalid data. When in this state, the management application is allowed to modify the values of the objects contained in the associated rows of the table. Once this operation is completed, the management station may set the state to *valid*; alternatively, the management station may cancel the operation by setting the state to *invalid*. If the agent determines that an entry has been in the *underCreation* state for an abnormally long time, it may set this object to *invalid*.

The *valid* state shall indicate that the corresponding rows of the *dynObjDef*(Table) with the same index *dynObjNumber* contain information that is believed to be *valid*.

The following table indicates the actions that shall take place upon receipt of a set request to change the state of the corresponding rows of the dynObjDef(Table) with the same index *dynObjNumber*. The value of each cell in the table shows the result of receiving the indicated set request (column headings) when the device is in the indicated state (row headings).

Table 1 -- State Transition Table

		COMMANDED STATE		
		<i>invalid</i>	<i>underCreation</i>	<i>valid</i>
CURENET STATE	<i>invalid</i>	invalid (1)	underCreation (1)	invalid (3)
	<i>underCreation</i>	invalid (2)	underCreation (3)	valid (4) or underCreation (5)
	<i>valid</i>	invalid (2)	valid (3)	valid (1)

Notes:

- (1) no action takes place and response indicates noError.
- (2) all entries associated with the ConfigEntryStatus object are deleted or de-referenced and response indicates noError
- (3) no action takes place but response indicates badValue.
- (4) if consistency check succeeds then state changes to *valid* and response indicates noError.
- (5) if consistency check fails then state remains *underCreation* and response indicates genErr.

Upon receipt of set request for the *valid* state when in the *underCreation* state, the agent shall perform a consistency check on the data contained in the associated rows of the dynObjDef(Table) with the same index *dynObjNumber* of the table. If the consistency check is successful, the state shall change to *valid*, otherwise, the state remains in the *underCreation* state and the device shall return a genErr.

4.2 STRUCTURE OF TRANSPORTATION MANAGEMENT INFORMATION

- Modify the second full paragraph to read:

The TMIB-II shall have ~~two~~three nodes located under the *transportation* node. The *protocols* node shall be the beginning of a subtree that holds information about various communications protocols. The *devices* node shall be the beginning of a subtree that holds information about various standard transportation device objects. The *tcip* node shall be the beginning of a subtree that holds information about various standard transit objects. The ASN.1 notation for these ~~two~~three nodes shall be defined as:

- Add the following definition to the end of this Clause:

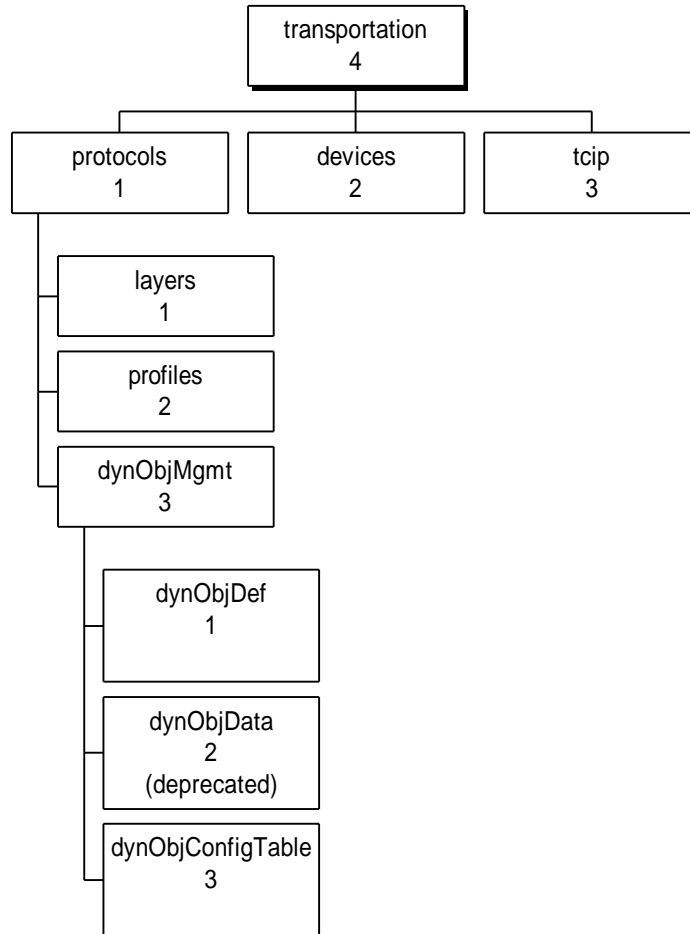
```
tcip      OBJECT IDENTIFIER ::= { transportation 3 }
```

4.2.1.1 Dynamic Object Management (dynObjMgmt)

- Modify the first paragraph to read:

The *dynObjMgmt* subtree is used by the STMP to reduce the bandwidth requirements of communicating sets of objects between a management station and agents. The *dynObjMgmt* tree contains ~~two~~three nodes: *dynObjDef*, ~~and *dynObjData*, and *dynObjConfigTable*.~~ All of the nodes under *dynObjData* have been deprecated.

- Modify the figure to the following:



4.2.1.1.1 Dynamic Object Definition (dynObjDef)

- Delete *dynObjOwner* and *dynObjStatus* from the table diagram.

- Delete paragraphs four through the end of the clause and replace with the following text.

The *dynObjIndex* shall be of type **INTEGER (1..255)**. It identifies one of the possible 255 object identifiers that is associated with each of the dynamic objects. The sequence of indexes used to fill out the table is not critical. However, once completed, the indexes must be in order with no gaps.

The *dynObjVariable* shall be of type **OBJECT IDENTIFIER**. It must point to an object supported by the device. For some columnar objects, only the prefix portion of an object identifier can be validated. The suffix or index portion of an object identifier need not be instantiated or existing at the time a *dynObjVariable* is defined.

The *dynObjOwner* and *dynObjStatus* objects (from TMIB version 1) have been deprecated and replaced with similar objects contained in the *dynObjConfigTable* as defined in 4.2.1.1.3. This has the affect of associating a single owner and status with each dynamic object rather than with each indexed item within a dynamic object.

4.2.1.1.2 Dynamic Object Data (dynObjData)

- Replace all of the text in this clause with the following text:

The objects under this node have been deprecated.

4.2.1.1.3 Dynamic Object Configuration Table

- Add this new Clause with the following text:

The *dynObjConfigTable* is a table indicating the owner and status of each dynamic object. It is an integral part of the *dynObjDef*(Table) and uses the same *dynObjNumber* as its index. The *dynObjConfigTable* shall have conceptual rows that contain the following objects:

dynObjNumber	dynObjConfigOwner	dynObjConfigStatus
---------------------	--------------------------	---------------------------

The INDEX for a particular row in the *dynObjDef* table is defined by *dynObjNumber*. As stated previously, the *dynObjNumber* shall be of type **INTEGER (1..13)**. It identifies with which of the 13 dynamic objects this row of the table is associated.

The *dynObjConfigOwner* shall be of type **OwnerString**. Its only purpose is to indicate the identity of the management station that defined the dynamic object. The default value shall be a zero (0) length string.

The *dynObjConfigStatus* shall be of type **ConfigEntryStatus**. It shall be used as described in 4.1.4. The *invalid* state shall indicate the associated dynamic object is not defined. The *underCreation* state shall indicate the management station is currently configuring the dynamic object (i.e., setting the values of the *dynObjConfigOwner* and various *dynObjVariable* objects associated with the subject dynamic object). The *valid* state shall indicate that the management station has completed the setup, and that the agent has successfully completed a validity check on the data.

When validating the data entered for a dynamic object, an agent shall perform the following consistency checks:

1. *dynObjIndex* 1 has a *dynObjVariable* entry.
2. All *dynObjVariables* have sequential *dynObjIndexes*
3. There are no skipped *dynObjIndexes*.

Failure to pass these consistency checks shall prevent the state from changing to valid.

A management station must take into account the variable binding list processing nature of SNMP. In SNMP, all objects contained in a single set-request data packet appear to be set to their new values simultaneously. Therefore, a management station must not combine a state change request with a request to set an instance value associated with that state change. If such an operation is attempted, the operation may not be correctly processed.

4.2.3 TCIP Subtree

- Add this new Clause with the following text:

This subtree shall be used for standard objects that are defined for specific types of transit data. The definition of the objects under this tree is left to other documents. The registration authority for this node is the Institute of Transportation Engineers.

SECTION 5

- Modify the first paragraph to read:

Applications conforming to this standard shall support the Simple Network Management Protocol (SNMP) ~~Simple Transportation Management Protocol (STMP)~~ and optionally support the Simple Transportation Management Protocol (STMP) ~~Simple Network Management Protocol (SNMP)~~. The conformance level of a device shall determine whether it supports only ~~STMP~~ SNMP or both STMP and SNMP.

5.1.1 Protocol Data Unit

- Modify the lower portion of the table to read:

3-0	Object ID	
	0000	NEMA node used as root <u>reserved</u>
	0001-1101	ID of STMP "dynamic object"
	1110	absolute root used <u>reserved</u>
	1111	reserved

5.1.1.1 PDU Format Bit

- Modify this clause to read:

The value of the PDU Format bit shall be used to determine if the protocol is STMP ~~as described in 0.~~

5.1.1.2 Message Type Bit Field

- Delete the paragraph pertaining to the get error response (paragraph eight).

- Modify paragraph nine to read:

An error response shall be sent from the agent to the manager in response to a set request, ~~set request, get request, or get next~~ request that contained an error. The error information field is described in 5.1.1.5.

5.1.1.3 Object ID Field

- For all instances within this Clause, change the word "Packed" to "Octet" and the acronym "PER" to "OER".

- Modify the first paragraph to read:

The object ID field shall be used to determine how the following information in the packet is encoded. STMP supports ~~two encoding rules (BER and P~~ Octet Encoding Rules (OER) and provides ~~some compression of OBJECT IDENTIFIERS~~ to reduce the bandwidth requirements of the protocol.

- Delete paragraph two and paragraph five.

- Modify the last sentence of the second paragraph as shown.

The value of the OBJECT ID field selects one of the thirteen objects. ~~under the iso(1).org(3).dod(6).~~

~~internet(1).private(4).enterprises(1).nema(1206).transportation(4).protocols(1).dynObjMgmt(3)
dynObjData(2).~~

- Modify paragraph six to read:

If the value of the OBJECT ID field is 0 (0x0), 14 (0xE), or 15 (0xF) then the packet shall be ignored.

5.1.1.4 PDU Information Field

- Modify the first part of the text in this clause, through the end of the alphabetized list, with the following text:

The form of the PDU Information field for a *set request*, *set request no reply*, and *get response* shall be an ASN.1 SEQUENCE of the objects referenced by the associated *dynObjVariables*, in order. For example, if the dynamic object is defined to have two indexed variables, the first referencing an object of SYNTAX INTEGER (0..255) and the other referencing an object of SYNTAX OCTET STRING (SIZE (0..127)), the STMP PDU Information field would be defined as:

```
SEQUENCE {  
    value1    INTEGER (0..255),  
    value2    OCTET STRING (SIZE (0..127))  
}
```

- In the definition of "STMPErrorResponse", delete the word "IMPLICIT".

- Modify the next paragraph to read:

Set response, get request, and get-next request messages shall not include a PDU Information field.

5.1.1.5 Error Responses

- Replace the third paragraph that follows item e. with the following paragraph:

The index number of the error response will indicate the dynObjIndex of the entry in the Dynamic Object Table that resulted in the error. In other words, the index number of the error response will indicate the field within the dynamic object that resulted in an error. The index number shall be encoded as INTEGER (0..255).

5.1.2 Operations and Semantics

- Delete items (a) and (d) from the list.

5.1.2.1 Basic Encoding Rules

- Delete the entire clause.

5.1.2.2 Packed Encoding Rules

- For all instances within this Clause, change the word "Packed" to "Octet" and the acronym "PER" to "OER". This change will effect the Clause title as well as paragraphs one and three of this Clause.

- Modify the first sentence of the first paragraph to read:

Within the STMF, The POER only applies to works with STMP. ~~and only on dynamic objects.~~

- Change the third paragraph to read:

~~POER~~ takes advantage of the ~~definition of dynamic objects~~ predefined data types to eliminate certain elements of the BER encoding. Since dynamic objects must be defined prior to their use ~~created at run time~~, both the manager and the agent know what each type is and for types of fixed length (e.g. Integer32) what the length of the data is. This allows the ~~POER~~ encoding to eliminate the type element and on fixed length data objects the length element can be eliminated as well. When a syntax definition includes some type of limitation on the range of values, this is referred to as a visible constraint.

- Modify the alphabetized list as follows:

- b. All objects that resolve to the Primitive type NULL ~~or INTEGER that are explicitly sized~~ shall not be encoded as a fixed length field. ~~The length is determined by the maximum value that an object can support. For example, the maximum size of an INTEGER (SIZE (4)) is four octets and the maximum size of an INTEGER (0..255) is one octet.~~
- c. All objects that resolve to ~~the a~~ Primitive type of variable sized OCTET STRING or OBJECT IDENTIFIER ~~or unsized INTEGER~~ shall be encoded as a variable length field.

- Delete the last sentence of item (e) as this can not occur.

- Add the following to the alphabetized list:

- f. All objects that resolve to the Primitive Type INTEGER and contain named values shall be constrained to the range (0..127). This provides backward compatibility with those standards already approved. It is expected that a mechanism will be provided in the future to support larger ranges.
- g. All objects that resolve to the Primitive Type INTEGER, not covered above, shall be encoded according to the rules defined in 5.1.2.3.
- h. All objects that resolve to a Primitive Type of a fixed sized OCTET STRING (e.g., OCTET STRING (SIZE (5))) shall be encoded as a fixed length field.

- Add the following clauses to the end of Clause 5.1.2

5.1.2.3 Encoding of an Integer Value

The encoding of an integer type is dependent upon the range of valid values for the type, including any possible future extensions based on the presence of an extension marker.

5.1.2.3.1

If an OER-visible constraint specification does not allow any negative values, the following rules apply.

If the constraint specification limits the possible values to less than or equal to 255, the length octets shall be omitted and the contents octet shall be encoded as a one-byte unsigned integer (see 5.1.2.4.2.1).

If the constraint specification allows values greater than 255, but limits the values to less than or equal to 65535, the length octets shall be omitted and the contents octet shall be encoded as a two-byte unsigned integer (see 5.1.2.4.2.2)

If the constraint specification allows values greater than 65535, but limits the values to less than or equal to 4294967295, the length octets shall be omitted and the contents octet shall be encoded as a four-byte unsigned integer (see 5.1.2.4.2.3)

If the constraint specification allows values greater than 4294967295, the length octets shall be present and the contents octet shall be encoded as an unrestricted unsigned integer (see 5.1.2.4.2.4)

5.1.2.3.2

If an OER-visible constraint specification allows negative values, the following rules apply. Note that, by definition, these rules apply when there is no OER-visible constraints.

If the constraint specification limits the maximum possible value to less than or equal to 127 and limits the minimum possible value to greater than or equal to -128, the length octets shall be omitted and the contents octet shall be encoded as a one-byte signed integer (see 5.1.2.4.1.1).

If the conditions of the preceding paragraph are not met **and** the constraint specification limits the maximum value to less than or equal to 32767 **and** the constraint specification limits the minimum value to greater than -32768, the length octets shall be omitted and the contents octet shall be encoded as a two-byte signed integer (see 5.1.2.4.1.2)

If the conditions of the preceding paragraph are not met **and** the constraint specification limits the maximum value to less than or equal to 2147483647 **and** the constraint specification limits the minimum value to greater than -2147483648, the length octets shall be omitted and the contents octet shall be encoded as a four-byte signed integer (see 5.1.2.4.1.3)

If the conditions of the preceding paragraph are not met, the length octets shall be present and the contents octet shall be encoded as an unrestricted signed integer (see 5.1.2.4.1.4)

5.1.2.3.3 Examples

INTEGER would be encoded as an unrestricted signed integer.

Counter would be encoded as a four-byte unsigned integer.

TimeTicks would be encoded as a four-byte unsigned integer.

Gauge would be encoded as a four-byte unsigned integer.

INTEGER (0..MAX) would be encoded as an unrestricted unsigned integer

INTEGER (0..255) would be encoded as a one-byte unsigned integer.

Counter (0..255) would be encoded as a one-byte unsigned integer.

INTEGER (0..2000) would be encoded as a two-byte unsigned integer.

INTEGER (1999..2000) would be encoded as a two-byte unsigned integer.

Gauge (1200..1250) would be encoded as a two-byte unsigned integer.

INTEGER (0..255, ...) would be encoded as an unrestricted signed integer due to the extension marker.

INTEGER (-128..127) would be encoded as a one-byte signed integer

INTEGER (-1000..1000) would be encoded as a two-byte signed integer.

INTEGER {a(1), b(2)} would be encoded as a one-byte unsigned integer. Although the "NamedNumberList" is not significant according to ASN.1 rules, Section 5.1.2.2, Item f. is applied to constrain the range to INTEGER (0..127).

5.1.2.4 Encoding of Whole Numbers

5.1.2.4.1 Encoding of a signed integer

5.1.2.4.1.1 Encoding of a one-byte signed integer

The value shall be encoded as an eight-bit 2's complement integer.

5.1.2.4.1.2 Encoding of a two-byte signed integer

The value shall be encoded as an sixteen-bit 2's complement integer.

5.1.2.4.1.3 Encoding of a four-byte signed integer

The value shall be encoded as an thirty-two bit 2's complement integer.

5.1.2.4.1.4 Encoding of an unrestricted signed integer

The value shall be encoded according to clauses 8.3.2 and 8.3.3 of BER

5.1.2.4.2 Encoding of a unsigned integer

5.1.2.4.2.1 Encoding of a one-byte unsigned integer

The value shall be encoded as an eight-bit binary-integer.

5.1.2.4.2.2 Encoding of a two-byte unsigned integer

The value shall be encoded as a sixteen-bit binary-integer.

5.1.2.4.2.3 Encoding of a four-byte unsigned integer

The value shall be encoded as a thirty-two-bit binary-integer.

5.1.2.4.2.4 Encoding of a unrestricted unsigned integer

The value shall be encoded as a minimum octet non-negative-binary-integer.

6.1.1 Conformance Level 1

- Modify this clause to read:

In order to be at conformance level 1 the following shall be supported:

- a. SNMP version 1: All supported objects including Internet standard objects are accessible. This level shall support the Internet system (group) objects.
- b. ~~STMP: All STMP operations shall be supported.~~
- c. ~~TMIB: All mandatory objects and groups defined within the TMIB shall be present.~~

- ~~d. Dynamic objects including: definition creation; definition deletion and object get and set data functions.~~

6.1.2 Conformance Level 2

- *Modify this clause to read:*

In order to be at conformance level 2 the following shall be supported:

- a. Conformance level 1
- b. STMP: All STMP operations
- c. TMIB: All mandatory objects and groups defined within the TMIB
- d. Dynamic objects including: definition creation; definition deletion and object get and set data functions.
- ~~b. SNMP version 1: All supported objects including Internet standard objects are accessible. This level shall support the Internet system (group) objects.~~

6.2 Management Application Conformance

- *Modify the paragraph as shown.*

Management Applications shall conform to either Conformance level 1 or 2 as described in 6.1.1 and 6.1.2. In addition, management applications may provisionally support the ability ~~must be able~~ to read ASCII text ASN.1 MIB modules such that the management applications may update their MIB views when appropriate.

Annex A

- *Add, as the first line of the annex:*

-- Copyright (c) 1996 by National Electrical Manufacturers Association.

- Modify the words "mgmt", "experimental", and "private" to "nemaMgmt", "nemaExperimental", and "nemaPrivate"

- *Modify the fourth line after Exports--Everything as follows:*

-- NEMA starts at { iso org dod internet private enterprises 1206 } in the global naming tree.

Annex B

- *Rename the Title of the Annex to:*

*TRANSPORTATION MANAGEMENT INFORMATION BASE
Version 2"*

- *Add, as the first line of the annex:*

-- Copyright (c) 1996 by National Electrical Manufacturers Association.

- Modify the text prior to the *IMPORTS* clause to read:

```
TMIB-II DEFINITIONS ::= BEGIN
-- Transportation Management Information Base and Framework - Version 2
```

- Delete all of the references to the *dynObj1* through *dynObj13*.

- Modify the following definitions to the following:

```
Byte ::= INTEGER (-128..127)
```

```
Short ::= INTEGER (-32768.. 32767)
```

```
Long ::= INTEGER (-2147483648..2147483647)
```

- Replace the definition of *EntryStatus* to the following:

```
ConfigEntryStatus ::= INTEGER
  { valid (1),
    underCreation (2),
    invalid (3)
  }
```

- Replace the comment after the *EntryStatus* with the following:

```
-- See Section 4.1.4 for the complete definition of this Type.
```

- Modify the definition of *OwnerString* as follows:

```
OwnerString ::= DISPLAY STRING (size (0..127)) DisplayString (SIZE (0..127))
```

- Change the *SYNTAX* of *dynObjDef* to read:

```
SYNTAX SEQUENCE OF e DynObjEntry
```

- Modify the definition of the following objects as follows:

```
dynObjNumber OBJECT-TYPE
```

```
SYNTAX INTEGER ( 1..13)
```

```
ACCESS read-writeonly
```

```
STATUS mandatory
```

```
DESCRIPTION
```

"The dynamic object number that this entry is to be associated with. ~~The order in which the objects are transmitted for a given dynamic object number is based on the value of the dynObjIndex. Lower values of dynObjIndex are transmitted first (for the same value of dynObjNumber).~~

~~This object may not be modified if the associated dynObjStatus object is equal to valid (1)."~~

```
::= { dynObjEntry 1 }
```

```
dynObjIndex OBJECT-TYPE
```

```
SYNTAX UByteINTEGER (1..255)
```

```
ACCESS read-only
```

```
STATUS mandatory
```

```
DESCRIPTION
```

"An index that uniquely identifies an entry in the dynamic object table. Each entry defines an object that is to be associated with a dynamic object number.

The ~~dynObjIndex and dynObjNumber together~~ determines the order in which objects are transmitted for the associated dynamic object. The lower dynObjIndex numbers are transmitted before larger numbers for entries within the same dynamic object ~~dynObjNumber.~~"

::= { dynObjEntry 2 }

dynObjVariable OBJECT-TYPE

SYNTAX OBJECT IDENTIFIER

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The object identifier of the particular variable to be included in the specified dynamic object number. Care must be taken when defining dynamic objects so that the maximum size of all the objects included in a dynamic object do not exceed the maximum packet size of the communications network.

When set to reference a columnar object, an agent may wish to only validate the prefix portion of the object identifier. The suffix or index portion of an object identifier need not be instantiated or exist at the time the time a dynObjVariable is defined.

This object may not be modified unless if the associated dynObjStatus object is equal to underCreation ~~valid(4).~~"

::= { dynObjEntry 3 }

dynObjOwner OBJECT-TYPE

SYNTAX OwnerString

ACCESS read-write

STATUS ~~mandatory~~ deprecated

DESCRIPTION

"The entity that configured this entry and is therefore using the resources assigned to it.

This object may not be modified if the associated dynObjStatus object is equal to valid(1).

This object has been replaced with dynObjConfigOwner."

::= { dynObjEntry 4 }

dynObjStatus OBJECT-TYPE

SYNTAX EntryStatus

ACCESS read-write

STATUS ~~mandatory~~ deprecated

DESCRIPTION

"The status of this dynamic object definition entry. See description of EntryStatus above for restrictions on accesses.

This object has been replaced with dynObjConfigStatus."

::= { dynObjEntry 5 }

- Under Dynamic Object Data, add the following comment:

-- all objects under this node have been deprecated due to their limited usefulness

- Change the STATUS field from "mandatory" to "deprecated" for all 13 dynObj# objects.
- Replace the END statement with the following:

```
--*****  
--      Dynamic Object Configuration Group  
--*****
```

dynObjConfigTable OBJECT-TYPE

SYNTAX SEQUENCE OF DynObjConfigEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION

“A table consisting of an owner and status for each of the 13 dynamic object definitions.”
::= { dynObjMgmt 3 }

dynObjConfigEntry OBJECT-TYPE

SYNTAX DynObjConfigEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION

“A table consisting of an owner and status for each of the 13 dynamic object definitions.”
INDEX { dynObjNumber }
::= { dynObjConfigTable 1 }

DynObjConfigEntry ::= SEQUENCE {
 dynObjConfigOwner OwnerString,
 dynObjConfigStatus ConfigEntryStatus }

dynObjConfigOwner OBJECT-TYPE

SYNTAX OwnerString
ACCESS read-write
STATUS mandatory
DESCRIPTION

“The entity that configured the associated dynamic object.”
::= { dynObjConfigEntry 1 }

dynObjConfigStatus OBJECT-TYPE

SYNTAX ConfigEntryStatus
ACCESS read-write
STATUS mandatory
DESCRIPTION

“Indicates the state of the associated dynamic object. Depending on the validity checks that are performed on the dynamic object definition, a set request may or may not be honored. See Section 4.1.4 for a complete description.”
::= { dynObjConfigEntry 2 }

END -- TMIB-II DEFINITIONS

