A Joint Standard of AASHTO, ITE, and NEMA

NTCIP 1201 version v03

National Transportation Communications for ITS Protocol

Global Object (GO) Definitions

published in March 2011

A major revision of NTCIP 1201 v02.32

This Adobe® PDF copy of an NTCIP standard is available at no-cost for a limited time through support from the U.S. DOT / Research and Innovative Technology Administration, whose assistance is gratefully acknowledged.

Published by

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

Institute of Transportation Engineers (ITE)
1627 I ("Eye") Street, N.W., Suite 600
Washington, D.C. 20006

National Electrical Manufacturers Association (NEMA)
1300 North 17th Street, Suite 1752
Rosslyn, Virginia 22209-3801

file version 1201 v03.15r rita-2011b

NOTICES

Copyright Notice

Visit www.ntcip.org for other copyright information, for instructions to request reprints of excerpts, and to request reproduction that is not granted below.

PDF File License Agreement

To the extent that these materials are distributed by AASHTO / ITE / NEMA in the form of an Adobe® Portable Document Format (PDF) electronic data file (the “PDF file”), AASHTO / ITE / NEMA authorizes each registered PDF file user to view, download, copy, or print the PDF file available from the authorized Web site, subject to the terms and conditions of this license agreement:

a) you may download one copy of each PDF file for personal, noncommercial, and intraorganizational use only;
b) ownership of the PDF file is not transferred to you; you are licensed to use the PDF file;
c) you may make one more electronic copy of the PDF file, such as to a second hard drive or burn to a CD;
d) you agree not to copy, distribute, or transfer the PDF file from that media to any other electronic media or device;
e) you may print one paper copy of the PDF file;
f) you may make one paper reproduction of the printed copy;
g) any permitted copies of the PDF file must retain the copyright notice, and any other proprietary notices contained in the file;
h) the PDF file license does not include (1) resale of the PDF file or copies, (2) republishing the content in compendiums or anthologies, (3) publishing excerpts in commercial publications or works for hire, (4) editing or modification of the PDF file except those portions as permitted, (5) posting on network servers or distribution by electronic mail or from electronic storage devices, and (6) translation to other languages or conversion to other electronic formats;
i) other use of the PDF file and printed copy requires express, prior written consent.

Data Dictionary and MIB Distribution Permission

To the extent that these materials are distributed by AASHTO / ITE / NEMA in the form of a Data Dictionary (“DD”) or Management Information Base (“MIB”), AASHTO / ITE / NEMA extend the following permission:

You may make or distribute unlimited copies, including derivative works, of the DD or MIB, including copies for commercial distribution, provided that:

a) each copy you make or distribute includes the citation “Derived from NTCIP 0000 [insert the standard number]. Copyright by AASHTO / ITE / NEMA. Used by permission.”;

b) the copies or derivative works are not made part of the standard publications or works offered by other standard developing organizations or publishers or as works-for-hire not associated with commercial hardware or software products intended for field implementation;
c) use of the DD or MIB is restricted in that the syntax fields may only be modified to define: 1) a more restrictive subrange; or 2) a subset of the standard enumerated values; or 3) a set of retired and defined enumerated values for systems supporting multiversion interoperability;
d) the description field may be modified but only to the extent that: 1) the more restrictive subrange is defined; and 2) only those bit values or enumerated values that are supported are listed.

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 / 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 / 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 a given user.

Use of these materials does not constitute an endorsement or affiliation by or between AASHTO, ITE, or NEMA and the user, the user’s company, or the products and services of the user’s company.

If the user is unwilling to accept the foregoing restrictions, he or she should immediately return these materials.

Content and Liability Disclaimer

The information in this publication was considered technically sound by the consensus of persons engaged in the development and approval of the document at the time it was developed. Consensus does not necessarily mean that there is unanimous agreement among every person participating in the development of this document.

AASHTO, ITE, and NEMA standards and guideline publications, of which the document contained herein is one, are developed through a voluntary consensus standards development process. This process brings together volunteers and seeks out the views of persons who have an interest in the topic covered by this publication. While AASHTO, ITE, and NEMA administer the process and establish rules to promote fairness in the development of consensus, they do not write the document and they do not independently test, evaluate, or verify the accuracy or completeness of any information or the soundness of any judgments contained in their standards and guideline publications.

AASHTO, ITE, and NEMA disclaim liability for any personal injury, property, or other damages of any nature whatsoever, whether special, indirect, consequential, or compensatory, directly or indirectly resulting from the publication, use of, application, or reliance on this document. AASHTO, ITE, and NEMA disclaim and make no guaranty or warranty, express or implied, as to the accuracy or completeness of any information published herein, and disclaims and makes no warranty that the information in this document will fulfill any of your particular purposes or needs. AASHTO, ITE, and NEMA do not undertake to guarantee the performance of any individual manufacturer or seller’s products or services by virtue of this standard or guide.

In publishing and making this document available, AASHTO, ITE, and NEMA are not undertaking to render professional or other services for or on behalf of any person or entity, nor are AASHTO, ITE, and NEMA undertaking to perform any duty owed by any person or entity to someone else. Anyone using this document should rely on his or her own independent judgment or, as appropriate, seek the advice of a competent professional in determining the exercise of reasonable care in any given circumstances. Information and other standards on the topic covered by this publication may be available from other sources, which the user may wish to consult for additional views or information not covered by this publication.

AASHTO, ITE, and NEMA have no power, nor do they undertake to police or enforce compliance with the contents of this document. AASHTO, ITE, and NEMA do not certify, test, or inspect products, designs, or installations for safety or health purposes. Any certification or other statement of compliance with any health or safety-related information in this document shall not be attributable to AASHTO, ITE, or NEMA and is solely the responsibility of the certifier or maker of the statement.

Trademark Notice

ACKNOWLEDGEMENTS

NTCIP 1201 v03 was prepared by the NTCIP Base Standards and Profiles Working Group (BSP2 WG), which is a subdivision of the Joint Committee on the NTCIP. The Joint Committee on the NTCIP is organized under a Memorandum of Understanding among the American Association of State Highway and Transportation Officials (AASHTO), the Institute of Transportation Engineers (ITE), and the National Electrical Manufacturers Association (NEMA). The Joint Committee on the NTCIP consists of six representatives from each of the standards organizations, and provides guidance for NTCIP development.

When NTCIP 1201 v03 was prepared, the following individuals were members of BSP2 WG:

- Bob De Roche (Chair)
- Ralph Boaz
- Manny Insignares
- Robert Lopes
- Greg Mizell
- Alex Mousadi
- Bryan Mulligan
- Peter Ragsdale
- Bob Rausch
- Joerg "Nu" Rosenbohm

Other individuals providing input include:

- Lesly Bien-Aime
- Blake Christie
- Jim Mahugh
- Keith Patton
- Jeris White

In addition to the many volunteer efforts, recognition is also given to those organizations that supported the efforts of BSP2 WG by providing comments and funding, including:

- U.S. Department of Transportation, Research and Innovative Technology Administration
- ConSysTec
- Econolite Control Products, Inc.
- Florida Department of Transportation
- Intelligent Devices, Inc.
- Noblis (formerly Mitretek)
- Pillar Consulting
- Quixote Traffic Corp.
- Robert DeRoche Consulting, Inc.
- Siemens ITS
- Signalisation Ver-Mac, Inc.
- Telvent Farradyne Inc. (formerly PB Farradyne)
- TransCore
- Trevilon
- Washington State DOT

FOREWORD

NTCIP 1201 v03 identifies and defines the common object definitions that may be supported by transportation devices that are NTCIP-conformant.

NTCIP 1201 v03 is an NTCIP Data Dictionary Standard. NTCIP Data Dictionary Standards provide definitions of data elements for use within NTCIP systems. NTCIP Data Dictionary Standards are approved by AASHTO, ITE, and NEMA, after recommendation by the NTCIP Joint Committee. The data is defined using the Simple Network Management Protocol (SNMP) object-type format as defined in RFC 1212 and NTCIP 8004 v02 format. This data is typically exchanged using one of the NTCIP 1103 recognized Application Layers (e.g., SNMP).

The following keywords apply to this document: AASHTO, ITE, NEMA, NTCIP, global, data, data dictionary, object.

For more information about NTCIP standards, visit the NTCIP website at www.ntcip.org.

User Comment Instructions

The term “User Comment” includes any type of written inquiry, comment, question, or proposed revision, from an individual person or organization, about any part of this standards publication’s content. A “Request for Interpretation” is also classified as a User Comment. User Comments are solicited at any time. In preparation of this NTCIP standards publication, input of users and other interested parties was sought and evaluated.

All User Comments will be referred to the committee responsible for developing and/or maintaining this standards publication. The committee chairperson, or their designee, may contact the submitter for clarification of the User Comment. When the committee chairperson or designee reports the committee’s consensus opinion related to the User Comment, that opinion will be forwarded to the submitter. The committee chairperson may report that action on the User Comment may be deferred to a future committee meeting and/or a future revision of the standards publication. Previous User Comments and their disposition may be available for reference and information at www.ntcip.org.

A User Comment should be submitted to this address:

NTCIP Coordinator
National Electrical Manufacturers Association
1300 North 17th Street, Suite 1752
Rosslyn, Virginia 22209-3801
e-mail: ntcip@nema.org

A User Comment should be submitted in the following form:

Standards Publication number and version:
Page:
Section, Paragraph, or Clause:
Comment:
Editorial or Substantive?:
Suggested Alternative Language:

Please include your name, organization, and address in your correspondence.

Approvals

NTCIP 1201 v03 was separately balloted and approved by AASHTO, ITE, and NEMA after recommendation by the Joint Committee on the NTCIP. Each organization has approved NTCIP 1201 v03 as the following standard type, as of the date:

> AASHTO—Standard Specification; December 2010
> ITE—Software Standard; December 2010
> NEMA—Standard; August 2010

History

In 1992, the NEMA 3-TS Transportation Management Systems and Associated Control Devices Section began the effort to develop the NTCIP. Under the guidance of the Federal Highway Administration's NTCIP Steering Group, the NEMA effort was expanded to include the development of communications standards for all transportation field devices that could be used in an ITS network.

In September 1996, an agreement was executed among AASHTO, ITE, and NEMA to jointly develop, approve, and maintain the NTCIP standards. In late 1998, the Global Object Working Group was tasked with updating the Global Object Definitions standard. The first meeting of the GO WG was held in January 1999. In March 2005, the Joint Committee voted to merge the Global Object Working Group with the Base Standards and Protocols Working Group.

From 1996 to 1999, a predecessor of NTCIP 1201 v03 was referenced as NEMA TS 3.4. However, to provide an organized numbering scheme for the NTCIP documents, the reference changed to a predecessor of NTCIP 1201 v03. The technical specifications of NTCIP 1201 v03 evolved as noted:

> NEMA TS 3.4-1996 v96.01.7, April 7, 1997. October 1996—Version 1.5 approved by NEMA. April 1997—Version 1.7 published by NEMA with editorial corrections. October 1996—Accepted as a Recommended Standard by the Joint Committee on the NTCIP. Approved by AASHTO in 1997 and approved by ITE in December 1997.
>
> NEMA TS 3.4 Amendment 1 v98.01.07. October 1998—Version 98.01.05 accepted as a Recommended Amendment by the Joint Committee on the NTCIP, and edited v01.07 referred for balloting and approval by NTCIP Standards Bulletin B0032 in May 1999. Approved by AASHTO in October 1999, approved by ITE in January 2001, and approved by NEMA in December 1999. Amendment 1 clarified ambiguities discovered during real-world implementations of this standard.
>
> NTCIP 1201:1996 [assigned version 01.08]. August 1999—Assigned NTCIP 1201 document number in NTCIP Standards Bulletin B0038. August 2000—Joint NTCIP Standards Publication cover used over TS 3.4 contents.
>
> NTCIP 1201:1996 v01.10, December 2001. January 2002—Formatted for printing: incorporated Amendment 1 v07 into the text; updated title page date and version number; modified and reorganized front matter to conform to NTCIP 8002. Most references to TS 3 standard designations were changed to equivalent NTCIP standard numbers.
>
> NTCIP 1201 v02. December 2002—Developed to reflect additional lessons learned, to incorporate better documentation (in the Annex) of some of the logic required to implement the standards, and to add new features requested by the ITS community.
>
> NTCIP 1201 v02.14. September 2001—Accepted by the NTCIP Joint Committee as a User Comment Draft. February 2002—NTCIP Standards Bulletin B0071 distributed NTCIP 1201 v02.16 for review and comment.

NTCIP 1201 v02.24. October 2002—Accepted by the NTCIP Joint Committee as a Recommended Standard. April 2004—NTCIP Standards Bulletin B0092 referred NTCIP 1201 v02.26 for balloting. Approved by AASHTO in October 2004, approved by ITE in March 2005, and approved by NEMA in November 2004.

NTCIP 1201 v02.31. February 2005—Disposed of ballot comments on backward compatibility, object deprecation, and others. In Section 1.3 Terms, added Deprecated and Obsolete definitions.

NTCIP 1201:2005 v02.32. October 2005—Edited document for publication with modified and reorganized front matter.

NTCIP 1201 v02.41+ Amendment 2 v09, November 2006 / NEVER PUBLISHED (see next history item), but items below were integrated into 1201 v03.03. Modified globalDaylightSaving with additional information regarding 'other' and added a set of begin and end daylight saving time (DST) objects to reflect changes enacted by U.S. Congress (Energy Policy Act of 2005) to take effect in 2007. Revised names of AuxIO objects that appeared in v02.31 and added the AuxIO object that appeared in NTCIP 1203 but is now listed as deprecated in 1203. Moved the object definitions previously defined under the globalReport node to NTCIP 1103. Amendment 2 v10. Included these additional changes:

a) In NTCIP 1201 v02.32, the ACCESS of auxIOPort Direction and auxIO2PortDirection should have been listed as read-only as this was incorrect in NTCIP 1203 v01.15.
b) In NTCIP 1201 v02.32, it was agreed the SYNTAX of auxIO2PortDescription should be changed from SIZE (0..50) to (0..255).
c) In NTCIP 1201 v02.32, it was agreed the SYNTAX of auxIO2PortResolution should be changed from SIZE (1..255) to (1..32).
d) In NTCIP 1201 v02.32, the maxAuxIO2TableNumDigitalPorts and maxAuxIO2TableNumDigitalPorts should not have been changed for 0..255 to 1..255 as this was what was shown in NTCIP 1203v01.15.
e) In NTCIP 1201 v02.32, the node registration of the new AuxIO objects shown in the ISO figure should have been {global 7}.

NTCIP 1201 v03.03 November 2007—The Joint Committee accepted v03.03 as a User Comment Draft, after the Joint Committee withdrew acceptance of Amendment 2 v10 because of ballot comments on the daylight saving time mechanisms. The WG proposed major version v03 with updated DST mechanisms, and with additional changes, and for easier version tracking. Major version v03 includes:

a) Revised the DST mechanism by replacing the previously proposed DST objects with new objects.
b) Added the SNMP interface dialogs from NTCIP 1203v02 to become the generic standard for all NTCIP device standards.
c) Deferred the addition of externally-developed test procedures to a future major version.

NTCIP 1201 v03.05 February 2008—Disposed of user comments. Main issues were associated with the new DST table.

NTCIP 1201 v03.08 April 2008—The NTCIP Joint Committee accepted v03.08 as the Recommended Standard. However, starting as early as the June 2006 BSP2 WG meeting, the WG proposed submitting a revised v03.08+ DST solution to the NTCIP Joint Committee as the Recommended Standard. Finally, during October to December 2009 teleconferences, the BSP2 WG resolved issues reported from deployment experience, and agreed on a revised approach to DST.

NTCIP 1201 v03.09e December 2009—Revised to address a DST compatibility issue with other standards. The original change deprecated the globalDaylightSaving object, which did not account for other standards that use this object to indicate whether DST was enabled.

NTCIP 1201 v03.10 March 2010—The NTCIP Joint Committee authorized their ballot action to accept the proposed replacement Recommended Standard, to include substantive revisions to reflect a revised approach to DST start and end dates, primarily in Section 2.4.2 and subsections. Other revisions, both related to DST and to other topics, addressed other issues. Annex B became Annex C, and Annex C became Annex B.

NTCIP 1201 v03.15 March 2011—In May 2010, version v03.13 was accepted as a replacement Recommended Standard. Standards Bulletin B0136 sent v03.13a to the SDOs for ballot and approval. After the December 2010 Joint Approval, NTCIP 1201 v03.15 was edited for publication.

Compatibility of Versions

To distinguish NTCIP 1201 v03 (as published) from previous drafts, NTCIP 1201 v03 also includes NTCIP 1201 v03.15 on each page header. All NTCIP Standards Publications have a major and minor version number for configuration management. The version number syntax is "v00.00a," with the major version number before the period, and the minor version number and edition letter (if any) after the period.

NTCIP 1201 v03 is designated, and should be cited as, NTCIP 1201 v03. Anyone using NTCIP 1201 v03 should seek information about the version number that is of interest to them in any given circumstance. The MIB, the PRL, and the PICS should all reference the version number of the standards publication that was the source of the excerpted material.

Compliant systems based on later, or higher, version numbers MAY NOT be compatible with compliant systems based on earlier, or lower, version numbers. Anyone using NTCIP 1201 v03 should also consult NTCIP 8004 v02 for specific guidelines on compatibility.

< This page intentionally left blank. >

CONTENTS

FIGURES

Page

TABLES

Page

< This page is intentionally left blank. >

Section 1
GENERAL

1.1 SCOPE

The messaging between the Transportation Management Center and field devices is accomplished by using the NTCIP Application Layer services to convey requests to access or modify values stored in a given device; these values are referred to as objects. NTCIP 1201 v03 identifies and defines object definitions that may be supported by multiple device types (e.g., actuated signal controllers and dynamic message signs). In the NTCIP family of standards, objects for a given device type are grouped in a device-type-specific data dictionary standard.

1.2 REFERENCES

Normative references contain provisions that, through reference in this text, constitute provisions of NTCIP 1201 v03. Other references in NTCIP 1201 v03 might provide a complete understanding of the entire protocol and the relations between all parts of the protocol. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent editions of the standard listed.

1.2.1 Normative References

AASHTO / ITE / NEMA NTCIP 1103 v02	*Transportation Management Protocols* published July 2010
AASHTO / ITE / NEMA NTCIP 8004 v02	*Structure and Identification of Management Information (SMI)* published June 2010
AASHTO / ITE / NEMA NTCIP 8005 v01	*Procedures for Creating Management Information Base (MIB) Files* published June 2010

1.2.2 Other References

IAB STD 15	(RFC 1157) *A Simple Network Management Protocol (SNMP)*. M. Schoffstall; M. Fedor; J. Davin; J. Case; 05/10/1990
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; K. McCloghrie; March 1991
IAB STD 17	*(RFC 1213) Management Information Base for Network Management of TCP/IP-based Internets: MIB-II. K. McCloghrie; M. Rose; March 1991*
ISO 14817:2002	*Transport information and control systems—Requirements for an ITS/TICS central Data Registry and ITS/TICS Data Dictionaries*
NEMA TS 2-2003	*Traffic Controller Assemblies with NTCIP Requirements*

1.2.3 Contact Information

1.2.3.1 IAB Documents

For Internet Architecture Board (IAB) documents, contact:

Internet Architecture Board (IAB)
www.rfc-editor.org
www.rfc-editor.org/repositories.html

1.2.3.2 NTCIP Standards

Copies of NTCIP standards may be obtained from:

NTCIP Coordinator
National Electrical Manufacturers Association
1300 N.17th Street, Suite 1752
Rosslyn, Virginia 22209-3801
www.ntcip.org
e-mail: ntcip@nema.org

Draft amendments, which are under discussion by the relevant NTCIP Working Group, and amendments recommended by the NTCIP Joint Committee are available.

1.2.3.3 National Electrical Manufacturers Association (NEMA) Standards

Obtain NEMA standards from:

National Electrical Manufacturers Association
1300 North 17th Street, Suite 1752
Rosslyn, Virginia 22209
www.nema.org

1.3 TERMS

For the purposes of NTCIP 1201 v03, the following terms, definitions, acronyms, and abbreviations apply. Terms not defined here are in accordance with their definitions in NTCIP 8004 v02. Electrical and electronic terms not defined here are used in accordance with their definitions in IEEE Std 100-2000. English words not defined here or in IEEE Std 100-2000 are used in accordance with their definitions in *Webster's New Collegiate Dictionary*.

ASC Actuated Signal Controller

Class An abstraction of any kind of object that may be described.

NOTE—Equivalent to an ISO 14817 Object Class.

Component A central system, field device, etc., that supports NTCIP.

Cyclic Redundancy Check (CRC) Polynomial algorithm performed on a specified range of data resulting in a 16 or 32 bit value

Data Value The value of a data element.

Database Object Any object identified as a ‘database object’ by the relevant NTCIP device-specific standard.

NOTE—For example, in NTCIP 1202 v02, objects identified as ‘P’ or ‘P2’ are database objects.

DST Daylight Saving Time

Deprecated Defined in NTCIP 8004 v02.

Feature A capability of a component.

Interchangeable A condition that exists when two or more items possess such functional and physical characteristics as to be equivalent in performance and durability, and are capable of being exchanged one for the other without alteration of the items themselves, or adjoining items, except for adjustment, and without selection for fit and performance.

NOTE—See National Telecommunications and Information Administration, U.S. Department of Commerce.

Interoperability The ability of two or more systems or components to exchange information and use the information that has been exchanged.

NOTE—See IEEE *Standards Dictionary, Glossary of Terms and Definitions*.

Management Information Base (MIB A collection of objects defined using Abstract Syntax Notation One (ASN.1) that can be accessed via a network management protocol.

Network Virtual Terminal American Standard Code for Information Interchange (NVT-ASCII)—As defined in RFC 854.

Obsolete Defined in NTCIP 8004 v02.

Point-to-MultiPoint Protocol (PMPP) A transportation specific subnetwork layer protocol that enables communication between multiple devices on the same communications line/channel.

PRL Protocol Requirements List

Profile Refers to a set of protocols, each of which operates independently on one of the seven (7) OSI Layers, if this layer is utilized.

NOTE—Different protocols are utilized at the same layer within different profiles.

Retired In the context of a data value, “retired” indicates the value is valid in limited circumstances, but has been replaced by another.

Simple Transportation Management Protocol (STMP) Part of the Transportation Management Protocols of the NTCIP effort.

NOTE—See NTCIP 1103 v02. STMP provides a simple and bandwidth efficient mechanism to communicate with field devices.

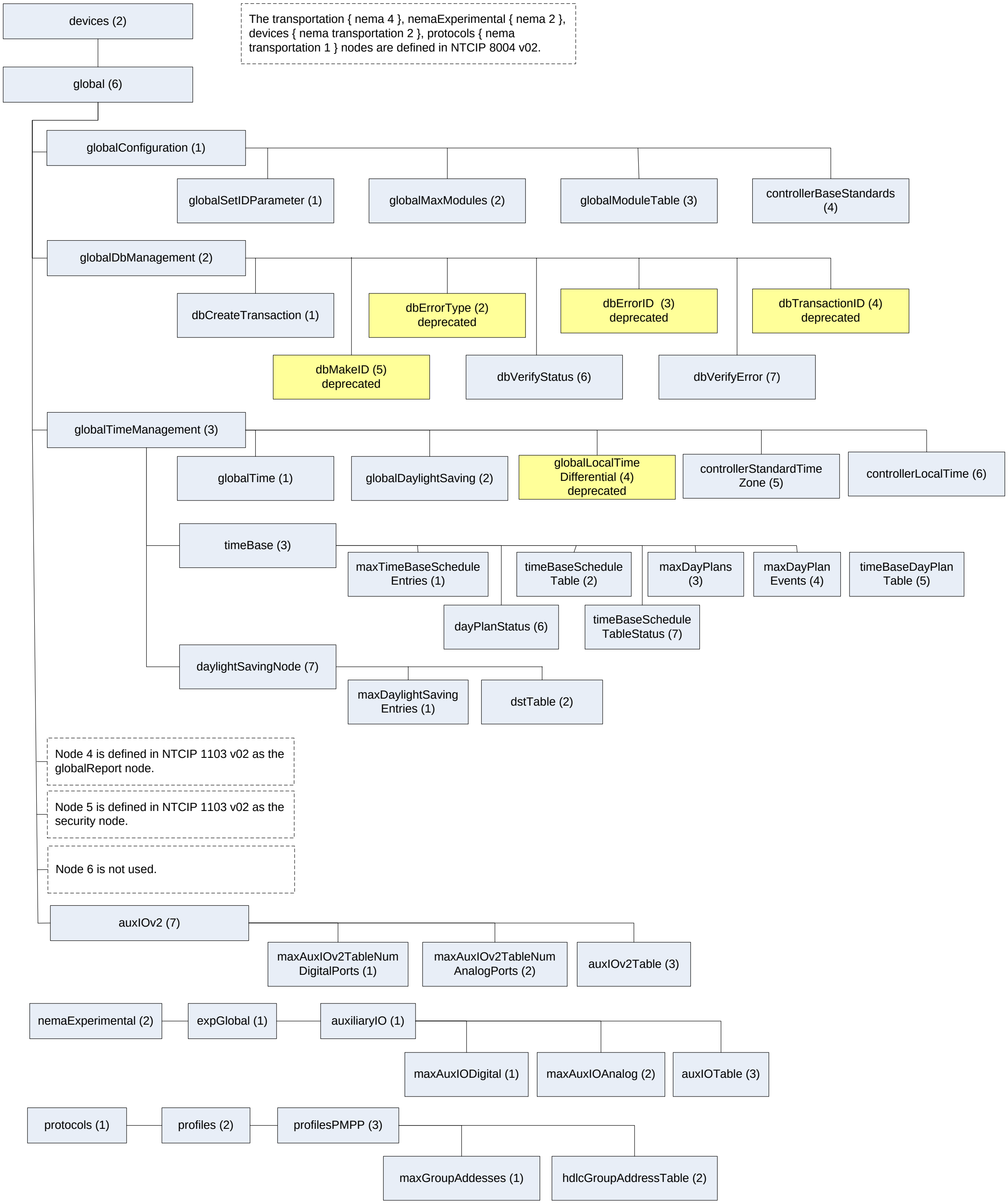
Static Database Object A parameter that does not change other than by a user command.

NOTE—For example, the controllerTimeZone object is a static database object since it only changes value through some sort of user command; however, globalTime is not a static database object since it is constantly incrementing.

UML Unified Modeling Language

1.4 OBJECT TREE

Figure 1 provides an overview of the organization of the data defined in NTCIP 1201 v03.



The transportation { nema 4 }, nemaExperimental { nema 2 }, devices { nema transportation 2 }, protocols { nema transportation 1 } nodes are defined in NTCIP 8004 v02.

Node 4 is defined in NTCIP 1103 v02 as the globalReport node.

Node 5 is defined in NTCIP 1103 v02 as the security node.

Node 6 is not used.

Figure 1 ISO Tree Structure

Section 2
MANAGEMENT INFORMATION BASE (MIB)
[NORMATIVE]

Section 2 defines those objects that are expected to be used by different device types, such as actuated signal controllers (ASC), variable message signs, and ramp meter controllers. The objects are defined in OJBECT-TYPE macro defined in RFC 1212 per the rules defined in NTCIP 8004 v02. The text provided from Section 2.1 through the end of Section 2 (except the headings) constitutes the standard NTCIP1201 v03 MIB.

To convert these object definitions into data concepts, e.g., for the exchange in center-to-center communications, the rules defined in NTCIP 8005 v01 shall apply.

All objects defined in NTCIP 1201 v03 reside under the "global" node of the global naming tree (see Section 1.4). To aid in object management, the "global" node has been subdivided into logical categories, each defined by a node under the "global" node. Individual objects are then located under the appropriate node.

Nodes should not be confused with conformance requirements, which are defined in NTCIP device-specific standards. NTCIP conformance requirements are based on logical groupings of objects that provide specific features that may be desired in a device. While conformance requirements frequently correspond to the nodal structure, a Protocol Requirements List (PRL) may contain objects that are not lexicographically ordered. For example, a schedule entry in a PRL may contain both "global" and "asc" specific objects.

NOTE—NTCIP 1201 v03 uses NTCIP 8004 v02 conventions. It specifies all (non-deprecated/non-obsolete) objects to have a STATUS of "mandatory" according to the conventions in NTCIP 8004 v02; it is the responsibility of any agency specification referring to NTCIP 1201 v03 to specify exactly which objects should be supported under what conditions, using a PRL. Agency specifications referring to NTCIP 1201:1996 (v01) should use the STATUS settings as defined in NTCIP 1201:1996 (v01).

NOTE—NTCIP 1201:1996 is sometimes referenced as NTCIP 1201:1997. In NTCIP 1201 v03, it is referenced as NTCIP 1201 v01 (following a revised NTCIP standards designation convention).

Text preceded by a double hyphen in MIB definitions represents normative text for NTCIP 1201 v03. Class diagrams contained in Annex B are informative.

2.1 NTCIP OBJECTS

```
-- ***************************************************************************
-- Filename:    1201v0315 Main.mib
-- Description: This MIB defines various globally applicable objects
-- Source:      NTCIP 1201 v03.15
-- MIB Revision History:
-- 09/27/04   Created file from Word .Doc file.
--            Changed name of file
--            Removed global report, logicalNameTranslation and communityName
--            nodes per NTCIP 1103v0124
--            Removed state transition diagram form 2.3.1
--            Changed "FROM NTCIP8004-A" to "FROM NTCIP8004-A-2004"
--            Added space before INTEGER in  timeBaseScheduleNumber INTEGER,
--            Changes status of moduleNumber, timeBaseScheduleNumber,
--            dayPlanNumber, dayPlanEventNumber, eventClassNumber,
--            eventConfigID  and their associated Entry to mandatory to
--            eliminate incompatible status errors
--            Checked with -CJ switch to eliminate gratuitous default value
--            warnings on globalTime and eventClassLimit.
--            Restructured IMPORTS around NTCIP8004-A-2004
-- 11/20/06   Added new beginning and end of DST objects. Revised Source
--            information
--            Added header and created standalone MIB without AuxIO objects
-- 07/02/07   Revised new beginning and end of DST objects.
-- 09/18/07   Further revisions of DST objects and updates to this Header.
-- 02/12/08   Addressed user comments. Again, modified UCD-defined DST
--            objects.  Updated header.  Created standalone MIBs for the
--            2 different versions of AuxIO objects.
-- 03/24/08   Addressed JC comments pertaining to the DST objects, minor
--            changes.  Updated header.
-- 03/00/11   Edited for publication. Extracted MIB from v0315 file.
--
--Copyright 2010 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 in whole or in part in any form, translation into other
--languages and display are reserved by the copyright owners 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 for the MIB, Do not copy without written permission of either AASHTO,
--ITE, or NEMA.
--
--                     Joint NEMA, AASHTO, and ITE
--                  NTCIP Management Information Base
--                        DISTRIBUTION NOTICE
--
--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 sub-range 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.
--
--NTCIP is a trademark of AASHTO/ITE/NEMA.
--****************************************************************************

--NTCIP OBJECTS
NTCIP1201-v03 DEFINITIONS ::= BEGIN

-- NTCIP 8004 v02 Header
--

--For the purpose of this section, the following OBJECT IDENTIFIERS are used:
IMPORTS
   OBJECT-TYPE
        FROM RFC-1212

   DisplayString
        FROM RFC1213-MIB
   devices, Protocols, profiles, global
        FROM NTCIP8004v02
   Opaque, Counter, Gauge, null
        FROM RFC1155-SMI;

-- global OBJECT IDENTIFIER ::= { devices 6 } 1.3.6.1.4.1.1206.4.2.6
```

2.2 GLOBAL CONFIGURATION NODE

```
globalConfiguration OBJECT IDENTIFIER ::= { global 1 }
-- This node is an identifier used to group all objects for support of
-- configuration functions that are common to most device types.
-- <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.1
```

2.2.1 Global Set ID Parameter

```
globalSetIDParameter OBJECT-TYPE
   SYNTAX   INTEGER (0..65535)
   ACCESS   read-only
   STATUS   mandatory
   DESCRIPTION
      "<Definition>  Specifies a relatively unique ID (e.g., this could be a
      counter, a check-sum, etc.) for all user-changeable parameters of the
      particular device-type currently implemented in the device.  Often this
      ID is calculated using a CRC algorithm.
      This value shall be calculated when a change of any static database
      object has occurred.  The value reported by this object shall not change
      unless there has been a change in the static data since the last
      request.  If the actual objects, which are to be included to create this
      object value, are not defined in the actual device-level standard such
      as 1202 or 1203, then the general guidance is to include all
      configuration objects that are stored in a type of memory that survives
      power outages.
      A management station can use this object to detect any change in the
      static database objects by monitoring this value after it has
      established a baseline.
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.1.1"
   ::= { globalConfiguration 1}
```

2.2.2 Maximum Modules Parameter

```
globalMaxModules   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   mandatory
   DESCRIPTION
      "<Definition>The number of rows that are listed in the
      globalModuleTable.
      <Unit>module
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.1.2"
   ::= { globalConfiguration 2}
```

2.2.3 Module Table

```
globalModuleTable   OBJECT-TYPE
   SYNTAX   SEQUENCE OF ModuleTableEntry
   ACCESS   not-accessible
   STATUS   mandatory
   DESCRIPTION
      "<Definition>A table containing information regarding manufacturer
      of software and hardware and the associated module models and
      version numbers as well as an indicator if the module is hardware
      or software related. The number of rows in this table shall equal
      the value of the globalMaxModules object.
      <TableType> static
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.1.3"
   ::= { globalConfiguration 3 }

moduleTableEntry OBJECT-TYPE
   SYNTAX   ModuleTableEntry
   ACCESS   not-accessible
   STATUS   mandatory
   DESCRIPTION
      "<Definition>This object defines an entry in the module table.
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.1.3.1"
   INDEX { moduleNumber }
   ::= { globalModuleTable 1 }

ModuleTableEntry ::= SEQUENCE {
   moduleNumber         INTEGER,
   moduleDeviceNode     OBJECT IDENTIFIER,
   moduleMake           OCTET STRING,
   moduleModel          OCTET STRING,
   moduleVersion        OCTET STRING,
   moduleType           INTEGER }
```

2.2.3.1 Module Number Parameter

```
moduleNumber OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   mandatory
   DESCRIPTION
      "<Definition>This object contains the row number (1..255) within
      this table for the associated module.
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.1.3.1.1"
   ::= { moduleTableEntry 1 }
```

2.2.3.2 Module Device Node Parameter

```
moduleDeviceNode   OBJECT-TYPE
   SYNTAX   OBJECT IDENTIFIER
   ACCESS   read-only
   STATUS   mandatory
   DESCRIPTION
      "<Definition>This object contains the device node number of the
      device-type, e.g., an ASC signal controller would have an OID of
      1.3.6.1.4.1.1206.4.2.1.
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.1.3.1.2"
   ::= { moduleTableEntry 2 }
```

2.2.3.3 Module Make Parameter

```
moduleMake   OBJECT-TYPE
   SYNTAX   OCTET STRING
   ACCESS   read-only
   STATUS   mandatory
   DESCRIPTION
      "<Definition>This object specifies the manufacturer of the
      associated module.  A null-string shall be transmitted if this
      object has no entry.
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.1.3.1.3"
   ::= { moduleTableEntry 3 }
```

2.2.3.4 Module Model Parameter

```
moduleModel  OBJECT-TYPE
   SYNTAX   OCTET STRING
   ACCESS   read-only
   STATUS   mandatory
   DESCRIPTION
      "<Definition>This object specifies the model number (hardware) or
      firmware reference (software) of the associated module.  A null-
      string shall be transmitted if this object has no entry.
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.1.3.1.4"
   ::= { moduleTableEntry 4 }
```

2.2.3.5 Module Version Parameter

```
moduleVersion OBJECT-TYPE
   SYNTAX   OCTET STRING
   ACCESS   read-only
   STATUS   mandatory
   DESCRIPTION
      "<Definition>This object specifies the version of the associated
      module.  If the moduleType has a value of software, the value of
      this object shall include the date on which the software was
      released as a string in the form of YYYYMMDD, it shall be followed
      by a space, a hyphen, another space, the lower-case letter ‘v’,
      followed by a version or configuration number.  Preceding zeros
      shall be required for the date.  For example, version 7.03.02 of
      the software released on July 5, 2002 would be presented as
      20020705 – v7.03.02
      A null-string shall be transmitted if this object has no entry.
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.1.3.1.5"
   ::= { moduleTableEntry 5 }
```

2.2.3.6 Module Type Parameter

```
moduleType  OBJECT-TYPE
   SYNTAX   INTEGER {
                  other (1),
                  hardware (2),
                  software (3) }
   ACCESS   read-only
   STATUS   mandatory
   DESCRIPTION
      "<Definition>This object specifies whether the associated module
      is a hardware or software module.
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.1.3.1.6"
   ::= { moduleTableEntry 6 }
```

2.2.4 Base Standards Parameter

```
controllerBaseStandards   OBJECT-TYPE
   SYNTAX   OCTET STRING (SIZE (0..256))
   ACCESS   read-only
   STATUS   mandatory
   DESCRIPTION
      "<Definition>For use in this object, an ASCII string that shall
      identify all of the standard document numbers that define or
      reference MIBs upon which the device is based.  Where applicable,
      profiles shall be referenced rather than the base standards.  The
      version string shall be constructed as follows: The acronym of the
      standards development organization (or other body) that developed
      and approved the standard; a space; the standards document number;
      a colon; and the documents version number as designated by the
      standards development organization (or other body).  Separate
      entries in the list of standards shall be separated by a carriage
      return (0x0d) and line feed (0x0a).
      In the case of NTCIP documents prior to formal approval, the
      version number shall be the version number in the form of lower
      case ‘v’ followed by the major version followed by a period
      followed by the minor revision.  In the case of approved NTCIP
      standards, the publication year shall precede the version number.
      In the case of amended NTCIP standards, the version number shall
      be replaced by the four digit year of publication of the published
      standard followed by the upper case letter ‘A’, followed by the
      amendment number.

      For example, a message sign may have the following value for this
      object:

            NTCIP 1201:v02.19
            NTCIP 1203:1997A1
            NTCIP 2101:2001 v01.19
            NTCIP 2103:v01.13
            NTCIP 2201:v01.14
            NTCIP 2301:2001 v01.08

      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.1.4"
   ::= { globalConfiguration 4 }
```

2.3 GLOBAL DATABASE MANAGEMENT NODE

```
globalDBManagement OBJECT IDENTIFIER ::= { global 2 }
-- <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.2
-- This node is an identifier used to group those objects used to manage a
-- transaction.
-- A transaction is a SET of one or more database parameters that have inter-
-- relationships with other database parameters, as such a SET for any one of
-- these objects must be validated against a set of consistency checks and may
-- potentially require the setting of a large number of objects
-- simultaneously.  Thus, the mode described by these objects allow for such a
-- large database download.
-- Any device standard that allows this feature shall define which objects are
-- database parameters versus status or control objects.
```

2.3.1 Database Creation Transaction

```
dbCreateTransaction   OBJECT-TYPE
SYNTAX      INTEGER {   normal (1),
                        transaction (2),
                        verify (3),
                        done (6)
                        }
ACCESS      read-write
STATUS      mandatory

DESCRIPTION
      "<Definition>This object provides transaction control for device
      configuration.  The transaction mode changes the behavior of the
      agent to force buffering of database objects until all related
      database objects have been modified.  In the normal mode, SET
      operations to any database object shall either be stored in a
      device’s database immediately with no regard to whether other
      changes will be made or be rejected (as defined in the device-
      specific Information Profile).  In the transaction mode, SET
      operations to any database object shall be buffered until a verify
      state performs a consistency check.  When the consistency check
      completes, the device automatically transitions to the done state
      where a normal or transaction command may be issued.
      A database object is a user-provided piece of setup information
      (or it may be defined in an information profile) that is necessary
      for the proper operation of a device.  It is static in nature in
      that the agent would never change it without direction from the
      management station.  For example, a parameter that defines a
      default mode of operation would be a database object.  A parameter
      that indicates the current state of the device would not be a
      database object.
      The states and commands are defined as:
      NORMAL:  SET operations behave as normal SETs and shall have an
      immediate effect on the value of any database objects used by the
      device if none of the objects contained in the operation require
      the use of the transaction mode (as defined in the device-specific
      Information Profile).  A SET operation containing any database
      object that requires the use of transaction mode shall result in a
      genErr.  This is the default state of this object.
      The only command that may be written to dbCreateTransaction while
      in this state is TRANSACTION.  Any other values written to this
      object in this state shall result in an error response of
      ‘badValue’.
      TRANSACTION: A SET operation of one or more database objects that
      use the same community name as used in the request for the
      TRANSACTION state are buffered by the agent device for later
      consistency checks and a normal response is returned.  A SET
      operation of one or more database objects using different
```

community names shall result in a genErr with the index set to zero. A SET operation without a community name field (e.g., an STMP operation) shall be buffered by the agent device for later consistency checks and a normal response is returned. Standard SYNTAX checking shall take place at the time of the SET operation. A transaction may consist of multiple SET operations over multiple frames.

A SET operation for one or more non-database objects shall be processed as normal even if it uses another community name, except for this (i.e., the dbCreateTransaction) object.

A SET operation containing both database and non-database objects shall be processed in full according to these two rules. Thus, if it contains the same community name as used in the request for the TRANSACTION state, the non-database objects shall be stored immediately while the database objects shall be buffered. If it uses a different community name, the entire request will be rejected and a genErr with an index of zero shall be returned.

GET operations on any object shall return the values of the data stored in the controller and shall ignore any values contained in the buffer.

Any valid community name may read this (dbCreateTransaction) object when in this state, but only the community name used to command the object to the transaction mode and the administrator community name can set this object. A set from any other community name shall result in a genErr with an index of zero. The only commands that can be written to dbCreateTransaction while in this state are VERIFY and NORMAL. A VERIFY command will change the state to VERIFY. If a NORMAL command is received, all buffered data is discarded and the state is returned to NORMAL. Any other values written to this object when in this state shall result in an error response of ‘badValue’.

VERIFY: Specific database objects are checked for consistency. When consistency checks are complete the device will automatically advance to the DONE state.

The state of dbCreateTransaction cannot be changed when in the VERIFY state. Any values written to this object in this state shall result in an error response of ‘badValue’.

The consistency check analyzes certain critical objects 'in context' and treats them as an interrelated whole rather than separate non-related data items. The consistency check rules are not defined in NTCIP 1201 v03, since these are device and implementation specific. Where applicable, the consistency check rules are defined in application specific object definition standards. A specific implementation may add additional checks beyond those defined in NTCIP standards.

A SET operation containing any database objects while in the VERIFY state shall result in a genErr with the index set to zero.

DONE: This state is entered automatically once consistency checks have completed in the VERIFY mode. The value of dbVerifyStatus and dbVerifyError indicate whether the consistency check found any errors.

A SET operation containing any database objects while in the DONE state shall result in a genErr with the index set to zero.

Any valid community name may read this (dbCreateTransaction) object when in this state, but only the community name used to command the object to the transaction mode and the administrator community name can set this object. A set from any other community

name shall result in a genErr with an index of zero. The only commands that can be written to dbCreateTransaction while in this state are NORMAL and TRANSACTION. Any other values written to this object in this state shall result in an error response of ‘badValue’.
If a NORMAL command is issued and dbVerifyStatus indicates doneWithNoError, the buffered data is transferred to the device memory and the state is returned to NORMAL. If a NORMAL command is issued and dbVerifyStatus indicates something other than doneWithNoError then the buffered data is discarded and the state is returned to NORMAL.
If a TRANSACTION command is issued, regardless of dbVerifyStatus, no action takes place (the buffered data is not changed) and the TRANSACTION state is re-entered.

		COMMANDED STATE (9)			
		transaction	*verify*	*normal*	*done*
CURRENT STATE	normal	transaction (1)	normal (2)	normal (2)	normal (2)
	transaction	transaction (2)	verify (3)	normal (4)	transaction (2)
	verify (7)	verify (2)	verify(2)	verify (2)	verify (2)
	done (8)	transaction (5)	done(2)	normal (6)	done (2)

Operational procedures and error responses:

(1) Once a copy of all database objects is placed in a buffer, the state is changed to transaction and error response indicates noError. If the operation fails, the state remains the same and error response indicates genErr.
(2) No action takes place, the state remains the same, but response indicates badValue.
(3) The state is changed to verify, a consistency check is started, and response indicates noError. Once the consistency check is completed, the state automatically changes to done.
(4) The buffered copy of all database objects is discarded, the state is changed to normal, and response indicates noError.
(5) The buffered copy of all database objects is not changed or reloaded, the state is changed to transaction, and response indicates noError.
(6) If dbVerifyStatus indicates doneWithNoError, then the copy of all database objects is transferred to memory, the state is changed to normal and response indicates noError. If dbVerifyStatus indicates doneWithError then the buffered data is discarded, the state is changed to NORMAL, and response indicates noError.
(7) The state automatically changes to done when the consistency check completes.
(8) dbVerifyStatus and dbVerifyError are only valid in this state.

```
            (9)  All SET operations on this (dbCreateTransaction) parameter
            shall be made using a protocol that uses a community name, or
            equivalent field (e.g., SNMP).
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.2.1"
DEFVAL      {normal}
::= { globalDBManagement 1 }
```

2.3.2 Database Error Type Parameter

-- This object has been deprecated. See Annex D.1.8 for more information.

```
dbErrorType OBJECT-TYPE
SYNTAX      INTEGER { tooBig (1),
               noSuchName (2),
               badValue (3),
               readOnly (4),
               genError (5),
               updateError (6),
               noError (7)}
ACCESS      read-only
STATUS      deprecated
DESCRIPTION
"This object returns the current error status of the transaction.  The value
of this
object is only valid when the dbCreateTransaction object is in the Done or
Error state.
<Object Identifier> 1.3.6.1.4.1.1206.4.2.6.2.2"
::= { globalDBManagement 2 }
```

2.3.3 Database Error ID Parameter

-- This object has been deprecated. See Annex D.1.8 for more information.

```
dbErrorID  OBJECT-TYPE
SYNTAX        OBJECT IDENTIFIER
ACCESS        read-only
STATUS        deprecated
DESCRIPTION
"This object contains the object identifier of the first object in the
transaction buffer that caused an error while dbCreateTransaction object was
in the Verifying or Updating state.  The value of this object is only valid
when the dbCreateTransaction object is in the Error state.  It is undefined
when the dbCreateTransaction object is in other states.
<Object Identifier> 1.3.6.1.4.1.1206.4.2.6.2.3"
::= { globalDBManagement 3 }
```

2.3.4 Database Transaction ID Parameter

-- This object has been deprecated. See Annex D.1.8 for more information.

```
dbTransactionID  OBJECT-TYPE
SYNTAX      INTEGER (0..255)
ACCESS      read-write
STATUS      deprecated
DESCRIPTION
"This object contains the transaction ID value that is to be contained in all
SET operation writes while the dbCreateTransaction object is not in the Normal
state.  During transaction operations every SET command shall begin with a
write to this object with the current value of this object.  If a SET
operation is performed without writing to this object, or with a value that
does not match the current value, then an error response of ‘genError’ shall
```

```
be returned.  This mechanism is used to determine that the same management
station that started the transaction is performing the SET operations that are
being buffered or modifying the state of dbCreateTransaction.
<Object Identifier> 1.3.6.1.4.1.1206.4.2.6.2.4"
::= { globalDBManagement 4 }
```

2.3.5 Database Make ID Parameter

-- This object has been deprecated. See Annex D.1.8 for more information.

```
dbMakeID OBJECT-TYPE
SYNTAX      INTEGER (0..255)
ACCESS      read-only
STATUS      deprecated
DESCRIPTION
"This object is used to create unique transaction ID’s for management stations
to use when starting transactions using the dbCreateTransaction object.  This
object will be incremented by one every time it is read, so that different
values will be returned for each read.  Management stations wishing to start a
transaction should first read the dbCreateTransaction object to verify that it
is in the Normal state.  If so then the management shall GET dbMakeID to
obtain a transaction ID to use, then SET dbCreateTransaction to startCmd and
dbTransactionID to the value just received.  If the response to the SET
operation is ‘noError’ then the management station has started a transaction.
If the response to the SET operation is ‘genError’ then the management station
should read the dbCreateTransaction and dbTransactionID objects to ensure that
the error was not due to a communications retry.  If the dbCreateTransaction
is in the Transaction state, and the dbTransactionID is the same value
returned by the read of this object, then the management station is the owner
of the transaction.  If the dbTransactionID does not match the value
originally returned by this object, then the management station is not the
owner of the transaction and must wait until the dbCreateTransaction object
returns to the Normal state before attempting to start the transaction.
<Object Identifier> 1.3.6.1.4.1.1206.4.2.6.2.5"
::= { globalDBManagement  5 }
```

2.3.6 Database Verify Status Parameter

```
dbVerifyStatus   OBJECT-TYPE
SYNTAX      INTEGER {   notDone (1),
                        doneWithError (2),
                        doneWithNoError (3) }
ACCESS      read-only
STATUS      mandatory
DESCRIPTION
      "<Definition>This object indicates the current status of verify
      (consistency checking) processing. The value of this object is
      only meaningful when the dbCreateTransaction object is in the
      Verify or Done state.
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.2.6"
::= { globalDBManagement 6 }
```

2.3.7 Database Verify Error Parameter

```
dbVerifyError   OBJECT-TYPE
SYNTAX      OCTET STRING (SIZE (0..255))
ACCESS      read-only
STATUS      mandatory
DESCRIPTION
```

```
      "<Definition>This object contains a textual description of or a
      reference to an error that was found by the verify (consistency
      checking) processing.  The value of this object is only meaningful
      when the dbCreateTransaction object is in the Done state and the
      dbVerifyStatus object is in the doneWithError state.
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.2.7"
::= { globalDBManagement 7 }
```

2.4 GLOBAL TIME MANAGEMENT NODE

```
globalTimeManagement   OBJECT IDENTIFIER
::= { global 3 }
--<Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3

-- This node is an identifier used to organize all objects for support of
-- time-related functions that are common to most device types.
```

2.4.1 Global Time Parameter

```
globalTime  OBJECT-TYPE
SYNTAX      Counter
ACCESS      read-write
STATUS      mandatory
DESCRIPTION
      "<Definition>The number of seconds since the epoch of 00:00:00
      (midnight) January 1, 1970 UTC (a.k.a. Zulu or GMT).
      <Unit>second
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.1"
DEFVAL      { 0 }
::= { globalTimeManagement 1}
```

2.4.2 Global Daylight Saving Parameter

```
-- This object has been modified with additional information on the operation
-- of 'enableUSDST' and the inclusion of one new value:
-- “enableDaylightSavingNode”.
-- See dstBeginMonth object for additional information.

globalDaylightSaving   OBJECT-TYPE
SYNTAX   INTEGER { other (1),
                   disableDST (2),
                   enableUSDST (3),
                   enableEuropeDST (4),
                   enableAustraliaDST (5),
                   enableTasmaniaDST (6),
                   enableEgyptDST (7),
                   enableNamibiaDST (8),
                   enableIraqDST (9),
                   enableMangoliaDST (10),
                   enableIranDST (11),
                   enableFijiDST (12),
                   enableNewZealandDST (13),
                   enableTongaDST (14),
                   enableCubaDST (15),
                   enableBrazilDST (16),
                   enableChileDST (17),
                   enableFalklandsDST (18),
                   enableParaguayDST (19),
                   enableDaylightSavingNode (20) }
ACCESS  read-write
STATUS  mandatory
```

```
DESCRIPTION
     "<Definition>This object specifies whether the daylight saving time (DST)
     is enabled, disabled or some other form of DST is active.
     other - DST adjustments by a mechanism not defined within this standard.
     disableDST -  DST clock adjustments shall NOT occur.
     enableUSDST - DST shall begin the first Sunday in April and shall end the
        last Sunday of October.  All changes of time occur at 2:00 AM. (This
        is the pre-2007 DST settings for the USA.)
     enableEuropeDST - DST shall start the last Sunday of March at 2:00 AM
        and ends the last Sunday of October at 3:00 AM.
     enableAustraliaDST - DST shall start the last Sunday in October at 2:00
        AM and ends the last Sunday in March at 2:00 AM.
     enableTasmaniaDST - DST shall start the first Sunday in October at 2 a.m.
        and ends the last Sunday in March at 3 a.m.
     enableEgyptDST –   DST shall start the last Friday in April and end the
        last Thursday in September.
     enableNamibiaDST – DST shall start the first Sunday in September and end
        the first Sunday in April.
     enableIraqDST –    DST shall start on April 1 and end on October 1.
     enableMongoliaDST – DST shall start the last Sunday in March and end the
        last Sunday in September.
     enableIranDST – DST shall start the first day of Farvardin and end the
        first day of Mehr
     enableFijiDST – DST shall start the first Sunday in November and end
        the last Sunday in February.
     enableNewZealandDST – DST shall start the first Sunday in October and end
        the first Sunday on or after March 5th.
     enableTongaDST – DST shall start the first Saturday in October and end
        the first Saturday on or after April 15th.
     enableCubaDST – DST shall start April 1st and end last Sunday in October.
     enableBrazilDST – DST shall start the first Sunday in October and end the
        last Sunday in February.
     enableChileDST – DST shall start the first Sunday on or after October 9th
        and end the first Sunday on or after March 9th.
     enableFalklandsDST – DST shall start the first Sunday on or after
        September 8th and end the first Sunday on or after April 8th.
     enableParaguayDST – DST shall start the first Sunday in October and end
        the last Saturday in February.
     enableDaylightSavingNode – DST operation is controlled by objects located
        under the daylightSavingNode.
<Informative> This object is maintained for backward compatibility and it is
envisioned that only the following values are supported with all other values
being ‘retired’:
    - other (1),
    - disableDST (2),
    - enableDaylightSavingNode (20)

NOTE:  Users should ensure that the values of globalDaylightSaving and the
entries in the new DST Table are coordinated.  The globalDaylightSaving object
is intended to be used to enable and disable DST and should not be set to the
value ‘20’, enableDaylightSavingNode until after the dstTable entries have
been fully configured.  Further, the globalDaylightSaving object
supersedes the settings in the DST Table.

<Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.2"
REFERENCE
"NEMA TS 2  Clause 3.8.2;
http://fatty.law.cornell.edu/uscode/15/260a.html;
http://webexhibits.org/daylightsaving/g.html "
DEFVAL      { enableDaylightSavingNode }
```

```
::= { globalTimeManagement 2 }
```

2.4.3 TimeBase Event Scheduler Node

```
timebase OBJECT IDENTIFIER ::= { globalTimeManagement 3 }
-- <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.3
-- This node is an identifier used to organize the main objects for event
-- scheduling.  Device type-specific objects (tables) pointed to are defined
-- within the appropriate MIB.
```

2.4.3.1 Maximum Number of Time Base Schedule Entries Parameter

```
maxTimeBaseScheduleEntries   OBJECT-TYPE
SYNTAX      INTEGER (1..65535)
ACCESS      read-only
STATUS      mandatory
DESCRIPTION
      "<Definition>The value of this object specifies the maximum number
      of different entries supported by the device as shown by the
      number of rows in the timeBaseScheduleTable.
      <Unit>TimeBaseScheduleEntry
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.3.1"
::= { timebase 1 }
```

2.4.3.2 Time Base Schedule Table

```
timeBaseScheduleTable   OBJECT-TYPE
SYNTAX      SEQUENCE OF TimeBaseScheduleEntry
ACCESS      not-accessible
STATUS      mandatory
DESCRIPTION
      "<Definition>A table containing the time base schedule parameters
      for the device.  The number of rows in this table shall be equal
      to the maxTimeBaseScheduleEntries object.  The table references
      the appropriate day plan for the device.  The plan is determined
      by comparing the current month (MONTH), day of week (DOW) and date
      of month (DOM) to the appropriate fields. The settings for MONTH,
      DOW and DOM are connected with a logical AND. To determine which
      timebased event to select, determine the event which has the most
      specific date specified.  Select the more specific event based on
      their MONTH settings; if the same, select the most specific DOM;
      if that is still the same, select the most specific DOW; if still
      the same, the first occurrence within the time base event table
      shall be selected. ’More specific’ means the least number of bits
      set within an object.  All entries in Time Base Schedule Table are
      expressed in local time and date.  A row in the table may be
      deactivated by setting the Month, Day, Date, or DayPlan parameters
      to zero (0)
      <TableType> static
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.3.2"
::= { timebase 2 }

timeBaseScheduleEntry   OBJECT-TYPE
SYNTAX      TimeBaseScheduleEntry
ACCESS      not-accessible
STATUS      mandatory
DESCRIPTION
      "<Definition>Event Parameters for the time based schedule
      programming of the device.
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.3.2.1"
INDEX   { timeBaseScheduleNumber }
```

```
::= { timeBaseScheduleTable 1 }

TimeBaseScheduleEntry ::= SEQUENCE {
            timeBaseScheduleNumber     INTEGER,
            timeBaseScheduleMonth      INTEGER,
            timeBaseScheduleDay        INTEGER,
            timeBaseScheduleDate       INTEGER,
            timeBaseScheduleDayPlan    INTEGER }
```

2.4.3.2.1 Time Base Schedule Number Parameter

```
timeBaseScheduleNumber   OBJECT-TYPE
SYNTAX      INTEGER (1..65535 )
ACCESS      read-only
STATUS      mandatory
DESCRIPTION
      "<Definition>The time base schedule number for objects in this
      row.  The value of this object shall not exceed the value of the
      maxTimeBaseScheduleEntries object.  The activation of a scheduled
      entry shall occur whenever allowed by all other objects within
      this table.
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.3.2.1.1"
::= { timeBaseScheduleEntry 1 }
```

2.4.3.2.2 Time Base Schedule Month of Year Parameter

```
timeBaseScheduleMonth   OBJECT-TYPE
SYNTAX      INTEGER (0..65535)
ACCESS      read-write
STATUS      mandatory
DESCRIPTION
      "<Definition>The Month(s) Of the Year that the schedule entry
      shall be allowed.  Each bit represents a specific month.  If the
      bit is set to one (1), then the scheduled entry shall be allowed
      during the associated month.  If the bit is set to zero (0), then
      the scheduled entry shall not be allowed during the associated
      month.  The bits are defined as:
       Bit      Month of Year
       0        Reserved
       1        January
       2        February
       3        March
       4        April
       5        May
       6        June
       7        July
       8        August
       9        September
       10       October
       11       November
       12       December
       13 - 15  Reserved
      Thus, a value of six (6) would indicate that the entry would only
      be allowed during the months of January and February.  A value of
      zero (0) shall indicate that this row has been disabled.
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.3.2.1.2"
::= { timeBaseScheduleEntry 2 }
```

2.4.3.2.3 Time Base Schedule Day of Week Parameter

```
timeBaseScheduleDay   OBJECT-TYPE
SYNTAX      INTEGER (0..255)
ACCESS      read-write
STATUS      mandatory
DESCRIPTION
      "<Definition>The Day(s) Of Week that the schedule entry shall be
      allowed. Each bit represents a specific day of the week.  If the
      bit is set to one (1), then the scheduled entry shall be allowed
      during the associated DOW.  If the bit is set to zero (0), then
      the scheduled entry shall not be allowed during the associated
      DOW.  The bits are defined as:
        Bit     Day of Week
        0       Reserved ('Holiday', not defined by this standard)
        1       Sunday
        2       Monday
        3       Tuesday
        4       Wednesday
        5       Thursday
        6       Friday
        7       Saturday
      Thus, a value of six (6) would indicate that the entry would only
      be allowed on Sundays and Mondays.  A value of zero (0) shall
      indicate that this row has been disabled.
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.3.2.1.3"
::= { timeBaseScheduleEntry 3 }
```

2.4.3.2.4 Time Base Schedule Date Parameter

```
timeBaseScheduleDate   OBJECT-TYPE
SYNTAX      INTEGER (0..4294967295)
ACCESS      read-write
STATUS      mandatory
DESCRIPTION
      "<Definition>The Day(s) Of a Month that the schedule entry shall
      be allowed.  Each bit represents a specific date of the month.  If
      the bit is set to one (1), then the scheduled entry shall be
      allowed during the associated date.  If the bit is set to zero
      (0), then the scheduled entry shall not be allowed during the
      associated date.  The bits are defined as:
        Bit     Day Number
        0       Reserved
        1       Day 1
        2       Day 2
        ||
        31      Day 31
      Thus, a value of six (6) would indicate that the entry would only
      be allowed on the first and second of the allowed months.  A value
      of zero (0) shall indicate that this row has been disabled.
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.3.2.1.4"
::= { timeBaseScheduleEntry 4 }
```

2.4.3.2.5 Time Base Schedule Day Plan Parameter

```
timeBaseScheduleDayPlan   OBJECT-TYPE
SYNTAX      INTEGER (0..255)
ACCESS      read-write
STATUS      mandatory
DESCRIPTION
```

```
      "<Definition>This object specifies what Plan number shall be
      associated with this timeBaseScheduleDayPlan object.  The value of
      this object cannot exceed the value of the maxDayPlans object.  A
      value of zero (0) shall indicate that this row has been disabled.
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.3.2.1.5"
::= { timeBaseScheduleEntry 5 }
```

2.4.4 Day Plan Parameters

2.4.4.1 Maximum Number of Day Plans—Parameter

```
maxDayPlans   OBJECT-TYPE
SYNTAX      INTEGER (1..255)
ACCESS      read-only
STATUS      mandatory
DESCRIPTION
      "<Definition>The value of this object specifies the maximum, fixed
      number of different timebased Day Plans supported by the device.
      The value of this object represents the number of day plans
      (primary key into the table) available in the
      timeBaseDayPlanTable.
      <Unit>DayPlan
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.3.3"
::= { timebase 3 }
```

2.4.4.2 Maximum Number of Day Plan Events—Parameter

```
maxDayPlanEvents   OBJECT-TYPE
SYNTAX      INTEGER (1..255)
ACCESS      read-only
STATUS      mandatory
DESCRIPTION
      "<Definition>The value of this object specifies the fixed number
      of different timebased Day Plan Events within each Day Plan
      supported by the device.  The value of this object represents the
      number of rows (secondary key into the table) available within
      each of the day plans that are available in the
      timeBaseDayPlanTable.  All day plans shall have the same number of
      day plan events available for use.
      <Unit>DayPlanEvent
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.3.4"
::= { timebase 4 }
```

2.4.4.3 Day Plan Table

```
timeBaseDayPlanTable OBJECT-TYPE
SYNTAX      SEQUENCE OF TimeBaseDayPlanEntry
ACCESS      not-accessible
STATUS      mandatory
DESCRIPTION
      "<Definition>A table containing day plan numbers, the times when
      to implement them and the associated actions. The number of rows
      in this table shall be equal to the product of the maxDayPlans
      object and the maxDayPlanEvents object.  The dayPlanNumbers within
      this table shall begin with day plan number 1 and increment by one
      to the maxDayPlans.  The dayPlanEventNumbers within this table
      shall begin with day plan event number 1 and increment by one to
      the maxDayPlanEvents.

      This table is always used in association with device-type specific
      objects specifying device-type specific actions such as activating
      a message on a VMS sign or initiating a pattern for a signal
      controller.  A device MIB that defines an action table should
      define the relative priority of the action table as compared to
      the priority of system and other commands.  The device-type
      specific action is only initiated when (1) the specific DayPlan
      has been activated, (2) the scheduler has sufficient priority to
      override the current operation of the device, and (3) at the
      indicated time.

      After a power recovery, or after a change to any object that
      affects controlerLocalTime, the operational mode called for by the
      scheduler shall be per the last event that would have been called
      for by the currently defined schedule; the logic searches for all
      events that may have occurred for at least the previous 24 hours.
      <TableType> static
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.3.5"
::= { timebase 5 }

timeBaseDayPlanEntry   OBJECT-TYPE
SYNTAX      TimeBaseDayPlanEntry
ACCESS      not-accessible
STATUS      mandatory
DESCRIPTION
      "<Definition>Day plan parameters for the time based schedule
      programming of a device.
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.3.5.1"
INDEX { dayPlanNumber, dayPlanEventNumber}
::= { timeBaseDayPlanTable 1 }

TimeBaseDayPlanEntry ::= SEQUENCE {
             dayPlanNumber          INTEGER,
             dayPlanEventNumber     INTEGER,
             dayPlanHour            INTEGER,
             dayPlanMinute          INTEGER,
             dayPlanActionNumberOID OBJECT IDENTIFIER }
```

2.4.4.3.1 Day Plan Number

```
dayPlanNumber   OBJECT-TYPE
SYNTAX      INTEGER (1..255)
ACCESS      read-only
STATUS      mandatory
DESCRIPTION
      "<Definition>This object specifies the day plan number for objects
      in this row. The value shall not exceed the value of the
      maxDayPlans object.  Day plan numbers are used in the TimeBase
      Event Table to specify day plan numbers to be implemented on
      specific days of the year or as part of the week plans.
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.3.5.1.1"
::= { timeBaseDayPlanEntry 1 }
```

2.4.4.3.2 Day Plan Event Number

```
dayPlanEventNumber   OBJECT-TYPE
SYNTAX      INTEGER (1..255)
ACCESS      read-only
STATUS      mandatory
DESCRIPTION
      "<Definition>This object identifies day plan event number(s) to be
      scheduled on a specific day plan number.  Several different events
      can be scheduled to take place during a day, and each of these
      events is one entry or row within a specified day plan number.
      The total number of events for one day plan shall not exceed the
      value of the maxDayPlanEvents object. If multiple non-conflicting
      events are scheduled to occur at the same time, they shall be
      logically executed in order of their dayPlanEventNumber with the
      lowest number occurring first.  An implementation shall omit lower
      number actions that are in conflict with higher number actions at
      the same time.
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.3.5.1.2"
::= { timeBaseDayPlanEntry 2 }
```

2.4.4.3.3 Day Plan Hour Parameter

```
dayPlanHour   OBJECT-TYPE
SYNTAX      INTEGER (0..23)
ACCESS      read-write
STATUS      mandatory
DESCRIPTION
      "<Definition>The Hour of day, as measured by the
      controllerLocalTime object, that the associated event shall become
      active.
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.3.5.1.3"
DEFVAL      {0}
::= { timeBaseDayPlanEntry 3 }
```

2.4.4.3.4 Day Plan Minute Parameter

```
dayPlanMinute   OBJECT-TYPE
SYNTAX      INTEGER (0..59)
ACCESS      read-write
STATUS      mandatory
DESCRIPTION
      "<Definition>The Minute of the hour (defined in the dayPlanHour),
      as measured by the controllerLocalTime object, that the associated
      event shall become active.
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.3.5.1.4"
DEFVAL      {0}
::= { timeBaseDayPlanEntry 4 }
```

2.4.4.3.5 Day Plan Action Number OID Parameter

```
dayPlanActionNumberOID   OBJECT-TYPE
SYNTAX      OBJECT IDENTIFIER
ACCESS      read-write
STATUS      mandatory
DESCRIPTION
      "<Definition>This object provides a reference to the device-type
      specific action that shall be executed.  The object shall
      reference the action by its associated object identifier,
      including its instance (i.e., the full OID of the scalar or
      columnar object).  Only objects whose description field explicitly
      states that they may be called by the action table may be
      referenced.  If a management system attempts to set this value to
      any other object identifier, the device shall respond with a
      genErr.
      Any object allowing the action table to reference it shall define
      precisely what action takes place when it is activated, and
      whether the action is transitionary or continuous until
      deactivated.  The object shall also define what, if any,
      restrictions may be placed on other operations the device may be
      able to perform.
      If the action to be performed is defined by a row of a table, one
      of the index columns should be identified as the explicit object
      that is referenced.
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.3.5.1.5"
DEFVAL      {null}
::= { timeBaseDayPlanEntry 5 }
```

2.4.4.4 Day Plan Status Parameter

```
dayPlanStatus   OBJECT-TYPE
SYNTAX      INTEGER (0..255)
ACCESS      read-only
STATUS      mandatory
DESCRIPTION
      "<Definition>This object indicates the current value of the active
      dayPlanNumber-object.  A value of zero (0) indicates that there is
      no dayPlanNumber that is currently active.
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.3.6"
::= { timebase 6 }
```

2.4.4.5 Schedule Status Parameter

```
timeBaseScheduleTableStatus OBJECT-TYPE
   SYNTAX   INTEGER (0..65535)
   ACCESS   read-only
   STATUS   mandatory
   DESCRIPTION
  "<Definition>This object indicates the number of the TimeBaseSchedule which
  is currently selected by the scheduling logic; the device may or may not be
  using the selected schedule.  The value of zero (0) indicates that there is
  no timeBaseScheduleNumber that is currently selected.
  <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.3.7"
::={timebase 7}
```

2.4.5 Global Local Time Differential Parameter

```
-- This object has been deprecated.  See Annexes D.2.1 and D.2.2 for more
information.
globalLocalTimeDifferential   OBJECT-TYPE
SYNTAX      INTEGER (-43200..43200)
ACCESS      read-write
STATUS      deprecated
DESCRIPTION
      "Indicates the number of seconds offset between local time and
      GMT.  Positive values indicate local times in the Eastern
      Hemisphere up to the International Date Line and negative values
      indicate local times in the Western Hemisphere back to the
      International Date Line.  If one of the daylight saving times is
      activated, this value will change automatically at the referenced
      time. For example, Central Standard Time (CST) is -21600 and
      Central Daylight Time (CDT) is -18000.
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.4"
::= { globalTimeManagement 4 }
```

2.4.6 Standard Time Zone Parameter

```
controllerStandardTimeZone   OBJECT-TYPE
SYNTAX      INTEGER (-43200..43200)
ACCESS      read-write
STATUS      mandatory
DESCRIPTION
```

```
      "<Definition> Indicates the number of seconds offset between local
      Standard Time and GMT.  Positive values indicate local times in
      the Eastern Hemisphere up to the International Date Line and
      negative values indicate local times in the Western Hemisphere
      back to the International Date Line.  This value does not change
      in response to a DST event.
      <Unit>second
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.5"
DEFVAL      {0}
::= { globalTimeManagement 5 }
```

2.4.7 Local Time Parameter

```
controllerLocalTime   OBJECT-TYPE
SYNTAX      Counter
ACCESS      read-only
STATUS      mandatory
DESCRIPTION
      "<Definition> The current local time expressed in seconds since
      00:00:00 (midnight) January 1, 1970 of the same time offset.  This
      value changes by 3600 seconds in response to a DST event.
      <Unit>second
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.6"
::= { globalTimeManagement 6 }
```

2.4.8 Daylight Saving Time (DST) Node

```
daylightSavingNode   OBJECT IDENTIFIER
::= { globalTimeManagement 7 }
--<Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.7
-- This node is an identifier used to organize all objects for support of
-- defining DST.  This function is common to most device
-- types.  The objects under this node only affect device operation when
-- globalDaylightSaving = enableDaylightSavingNode (20). See Annex A.2.2 for
-- examples.
```

2.4.8.1 Maximum Daylight Saving Time (DST) Table Entries Parameter

```
maxDaylightSavingEntries   OBJECT-TYPE
SYNTAX  INTEGER (1..100)
ACCESS  read-only
STATUS  mandatory
DESCRIPTION
   "<Definition> The maximum number of entries (begin and end date pairs) that
     the DST Table can contain within the device.

     As of July 2007, devices used within the United States only require
     1 entry when using the generic begin and end date method.

     <informative>It is expected that, for devices using the absolute date
     method, the device would need to support at least 1 entry per year
     programmed.
For multi-step DST transitions, a minimum of 2 rows are required (see Annex
A.2.1 Figure 6).
More than one row may be required if absolute date method (see Section
2.4.8.2.2) is used for more than one year, or if more than one time change is
implemented in a given year.

     <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.7.1"
::= { daylightSavingNode 1 }
```

2.4.8.2 Daylight Saving Time (DST) Table Parameter

```
dstTable   OBJECT-TYPE
   SYNTAX   SEQUENCE OF DstEntry
   ACCESS   not-accessible
   STATUS   mandatory
   DESCRIPTION
      "<Definition> A table containing DST Begin and End
       dates.  The table is useful for agencies with multiple daylight saving
       time incremental steps per year.  The number of rows in this table is
       equal to the maxDaylightSavingEntries object.
       <TableType> static
       <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.7.2"
::= { daylightSavingNode 2 }

dstEntry   OBJECT-TYPE
   SYNTAX   DstEntry
   ACCESS   not-accessible
   STATUS   mandatory
   DESCRIPTION
      "<Definition> The DST Begin and End dates parameters.
       <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.7.2.1"
   INDEX   { dstEntryNumber }
::= { dstTable 1 }

DstEntry ::= SEQUENCE {
   dstEntryNumber                           INTEGER,
   dstBeginMonth                            INTEGER,
   dstBeginOccurrences                      INTEGER,
   dstBeginDayOfWeek                        INTEGER,
   dstBeginDayOfMonth                       INTEGER,
   dstBeginSecondsToTransition              INTEGER,
   dstEndMonth                              INTEGER,
   dstEndOccurrences                        INTEGER,
   dstEndDayOfWeek                          INTEGER,
   dstEndDayOfMonth                         INTEGER,
   dstEndSecondsToTransition                INTEGER,
   dstSecondsToAdjust                       INTEGER }
```

2.4.8.2.1 Daylight Saving Time (DST) Entry Number Parameter

```
dstEntryNumber   OBJECT-TYPE
   SYNTAX   INTEGER (1..100)
   ACCESS   read-only
   STATUS   mandatory
   DESCRIPTION
      "<Definition> The entry number for the DST objects
       in this row.  This value shall not exceed the
       maxDaylightSavingEntries object value.
……………..
       <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.7.2.1.1
       <Unit> dstEntry"
::= { dstEntry 1 }
```

2.4.8.2.2 Daylight Saving Time (DST) Beginning Month Parameter

```
dstBeginMonth   OBJECT-TYPE
SYNTAX  INTEGER { january (1),
                  february (2),
                  march (3),
                  april (4),
                  may (5),
                  june (6),
                  july (7),
                  august (8),
                  september (9),
                  october (10),
                  november (11),
                  december (12),
                  absolute (13),
                  disabled (14)}

ACCESS  read-write
STATUS  mandatory
DESCRIPTION
     "<Definition> The month during which daylight saving time (DST) begins.
An entry of 'absolute' means that dstBeginSecondsToTransition defines an
absolute time to begin DST relative to midnight January 1, 1970.  In this
case, any value indicated in the dstEndMonth, dstBeginOccurences,
dstBeginDayOfWeek, dstBeginDayOfMonth, dstEndOccurances, dstEndDayOfWeek, and
dstEndDayOfMonth objects are irrelevant, and the dstEndSecondsToTransition
object defines an absolute time to end DST relative to midnight January 1,
1970.
     If the daylightSavingNode is enabled (i.e. globalDaylightSaving =
     enableDaylightSavingNode), and the value of this object is disabled(14),
     then the values in the remaining objects in this row of the
     dstTable are irrelevant and therefore ignored by the device.

     <Unit>
     <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.7.2.1.2"
     DEFVAL { March }
::= { dstEntry 2 }
```

2.4.8.2.3 Daylight Saving Time (DST) Beginning Occurrence Parameter

```
dstBeginOccurrences   OBJECT-TYPE
SYNTAX  INTEGER { first (1),
                  second (2),
                  third (3),
                  fourth (4),
                  last (5),
                  secondLast (6),
                  thirdLast (7),
                  fourthLast (8),
                  specificDayOfMonth (9) }
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
     "<Definition>For values of 1-4, the number of occurrences of the
      specific day of week that shall occur on or after
      dstBeginDayOfMonth until the daylight saving transition shall take
      place.

      For values of 5-8, the number of occurrences of the specific day of
      week that shall occur on or before dstBeginDayOfMonth until the daylight
      saving transition shall take place.

      For value = 9, dstBeginDayOfMonth defines the specific day of the month
      that the DST transition occurs regardless of value in
      dstBeginDayOfWeek object.
      NOTE: To specify the last occurrence of a specified day of the
      month, simply specify the last occurrence of the specified day of the
      week on or before the last day of the month (e.g., 31).
     <Unit>
     <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.7.2.1.3"
     DEFVAL { second }
::= { dstEntry 3 }
```

2.4.8.2.4 Daylight Saving Time (DST) Beginning Day of Week Parameter

```
dstBeginDayOfWeek   OBJECT-TYPE
SYNTAX  INTEGER { sunday (1),
                  monday (2),
                  tuesday (3),
                  wednesday (4),
                  thursday (5),
                  friday (6),
                  saturday (7) }
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
     "<Definition> The Day of the week on which daylight saving time
(DST)
      begins.  This object shall only apply if       dstBeginOccurrences = 1-
8.
     <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.7.2.1.4"
      DEFVAL { sunday }
::= { dstEntry 4 }
```

2.4.8.2.5 Daylight Saving Time (DST) Beginning Day of Month Parameter

```
dstBeginDayOfMonth      OBJECT-TYPE
SYNTAX  INTEGER (1..31)
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
     "<Definition> If dstBeginOccurrences = 1-8: The day of
      the month from which to begin counting occurrences of a specific
      day of the week (forward for values 1-4, and backwards for values 5-8).

      If dstBeginOccurrences = 9: The specific day of the month
      on which the transition occurs.
     <Unit> day of month
     <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.7.2.1.5"
     DEFVAL { 1 }
::= { dstEntry 5 }
```

2.4.8.2.6 Daylight Saving Time (DST) Beginning Seconds to Transition Parameter

```
dstBeginSecondsToTransition    OBJECT-TYPE
SYNTAX  INTEGER (0..4294967295)
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
     "<Definition> If dstBeginMonth = absolute, then this object
      defines when DST begins based on the seconds from midnight
      January 1, 1970 (UTC/GMT).

      If dstBeginMonth = 1-12 (January to December), then this object defines
      the time when DST begins in seconds past midnight relative to local time
      (see the controllerLocalTime object).
      NOTE:  a set of parameters that causes a day
      transition that crosses the midnight boundary may result in
      unexpected behavior.
     <Unit>seconds
     <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.7.2.1.6"
     DEFVAL { 7200 }
::= { dstEntry 6 }
```

2.4.8.2.7 Daylight Saving Time (DST) Ending Month Parameter

```
dstEndMonth   OBJECT-TYPE
SYNTAX  INTEGER { january (1),
                  february (2),
                  march (3),
                  april (4),
                  may (5),
                  june (6),
                  july (7),
                  august (8),
                  september (9),
                  october (10),
                  november (11),
                  december (12) }
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
     "<Definition> The month during which daylight saving time (DST) ends.
      If the value of dstBeginMonth object = 'absolute' or 'disabled',
      then the agent shall ignore the value of this object.
      Otherwise, the value of this object is valid.

     <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.7.2.1.7"
     DEFVAL { november }
::= { dstEntry 7 }
```

2.4.8.2.8 Daylight Saving Time (DST) Ending Occurrences Parameter

```
dstEndOccurrences   OBJECT-TYPE
SYNTAX  INTEGER { first (1),
                  second (2),
                  third (3),
                  fourth (4),
                  last (5),
                  secondLast (6),
                  thirdLast (7),
                  fourthLast (8),
                  specificDayOfMonth (9) }
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
     "<Definition>For values of 1-4, the number of occurrences of the
       specific day of week that shall occur on or after
       dstEndDayOfMonth until the daylight saving transition shall take place.

       For values of 5-8, the number of occurrences of the specific day
       of week that shall occur on or before dstEndDayOfMonth until the
       daylight saving transition shall take place.

       For value = 9, dstEndDayOfMonth defines the specific day of the month
       that the DST transition occurs regardless of value in
```

```
        dstEndDayOfWeek object.

        NOTE: To specify the last occurrence of a specified day of
        the month, simply specify the last occurrence of the specified day
        of the week on or before the last day of the month (e.g. 31).
     <Unit>
     <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.7.2.1.8"
     DEFVAL { first }
::= { dstEntry 8 }
```

2.4.8.2.9 Daylight Saving Time (DST) Ending Day of Week Parameter

```
dstEndDayOfWeek   OBJECT-TYPE
SYNTAX  INTEGER { sunday (1),
                  monday (2),
                  tuesday (3),
                  wednesday (4),
                  thursday (5),
                  friday (6),
                  saturday (7) }
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
     "<Definition> The Day of the week on which daylight saving time
(DST)
     ends.  This object shall only apply if dstEndOccurrences
     = 1-8.
     <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.7.2.1.9"
     DEFVAL { sunday }
::= { dstEntry 9 }
```

2.4.8.2.10 Daylight Saving Time (DST) Ending Day of Month Parameter

```
dstEndDayOfMonth  OBJECT-TYPE
SYNTAX  INTEGER (1..31)
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
     "<Definition> If dstEndOccurrences = 1-8: The day of the month from
      which to begin counting occurrences of a specific day of the week
      (forward for values 1-4, and backwards for values 5-8).

      If dstEndOccurrences = 9: The specific day of the month on which the
      transition occurs.
     <Unit> day of month
     <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.7.2.1.10"
     DEFVAL { 1 }
::= { dstEntry 10 }
```

2.4.8.2.11 Daylight Saving Time (DST) Ending Seconds to Transition Parameter

```
dstEndSecondsToTransition    OBJECT-TYPE
SYNTAX  INTEGER (0..4294967295)
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
     "<Definition> If dstBeginMonth = absolute, then this object defines
      when DST ends based on the seconds from midnight January 1, 1970
     (UTC/GMT).
      If dstBeginMonth = 1-12 (January to December), then this
      object defines the time when DST ends in seconds past midnight relative
      to local time (see the controllerLocalTime object).

      NOTE: a set of parameters that causes a day
      transition that crosses the midnight boundary may result in
      unexpected behavior.
     <Unit> seconds
     <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.7.2.1.11"
     DEFVAL { 7200 }
::= { dstEntry 11 }
--NOTE:  This object allows what may be considered an exception, in that it
--is possible and allowed to configure an adjustment backward past midnight.
```

2.4.8.2.12 Daylight Saving Time (DST) Seconds to Adjust Parameter

```
dstSecondsToAdjust    OBJECT-TYPE
SYNTAX  INTEGER (0..21600)
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
     "<Definition> This is the absolute offset in seconds that will be added
       to the local time reference point to determine the local time when DST
       is in effect as specified by this row entry.  Values of this object in
       adjacent rows, even if they overlap, are not cumulative. That is, the
       row with the latest dstBegin time, which has not terminated due to
       passing the dstEnd time, shall determine the setting of the local TOD
       clock; the dstSecondsToAdjust for the latest dstBegin governs the Local
       TOD clock settings.
       The maximum offset to adjust is 21600 seconds, an equivalent of
       6 hours.
     <Unit> seconds
     <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.3.7.2.1.12"
     DEFVAL { 3600 }
::= { dstEntry 12 }
```

2.5 REPORT PARAMETER NODE

-- NOTE—The objects originally under this node have been moved to NTCIP 1103 v02.

2.6 STMP OBJECT NODE

-- NOTE—The objects originally under this node have been moved to NTCIP 1103 v02.

2.7 PMPP OBJECT NODE

-- NOTE—PMPP objects may move to a future version of NTCIP 2101 v01 at some future point.

```
profilesPMPP OBJECT IDENTIFIER
::= { profiles 2 }
-- <Object Identifier> 1.3.6.1.4.1.1206.4.1.2.3


-- This node is an identifier used to group all objects for support of the
-- PMPP function that are common to all device types.  The objects under this
-- node are placed under the Protocols\Profiles\PMPP subtree within the NEMA
-- node, but they have been listed here due to the lack of a separate document
   that lists these objects.
```

2.7.1 Maximum HDLC Group Address Parameter

```
maxGroupAddresses OBJECT-TYPE
SYNTAX      INTEGER (1..255)
ACCESS      read-only
STATUS      mandatory
DESCRIPTION
      "<Definition>The maximum number of group addresses this device
      supports. This object indicates the maximum number of rows in the
      hdlcGroupAddressTable.
      <Unit>address
      <Object Identifier> 1.3.6.1.4.1.1206.4.1.2.3.1"
::= {profilesPMPP 1 }
```

2.7.2 HDLC Group Address Table

```
hdlcGroupAddressTable OBJECT-TYPE
SYNTAX      SEQUENCE OF HdlcGroupAddressEntry
ACCESS      not-accessible
STATUS      mandatory
DESCRIPTION
      "<Definition> A table containing group addresses at which a device
      may receive frames.
      <TableType> static
      <Object Identifier> 1.3.6.1.4.1.1206.4.1.2.3.2"
::= { profilesPMPP 2 }

hdlcGroupAddressEntry   OBJECT-TYPE
SYNTAX      HdlcGroupAddressEntry
ACCESS      not-accessible
STATUS      mandatory
DESCRIPTION
      "<Definition> An entry in the group address table that contains a
      device’s data link layer group address at which it will accept
      frames.
      <Object Identifier> 1.3.6.1.4.1.1206.4.1.2.3.2.1"
INDEX { hdlcGroupAddressIndex }
::= { hdlcGroupAddressTable 1 }

HdlcGroupAddressEntry ::= SEQUENCE {
   hdlcGroupAddressIndex            INTEGER,
   hdlcGroupAddress                 INTEGER, -- deprecated
   hdlcGroupAddressNumber           INTEGER }
```

2.7.2.1 HDLC Group Address Index Parameter

```
hdlcGroupAddressIndex   OBJECT-TYPE
SYNTAX      INTEGER (1..255)
ACCESS      read-only
STATUS      mandatory
DESCRIPTION
      "<Definition>The index number for the group address in this row.
      <Object Identifier> 1.3.6.1.4.1.1206.4.1.2.3.2.1.1"
::= { hdlcGroupAddressEntry 1}
```

2.7.2.2 HDLC Group Address Parameter

-- This object has been deprecated. See Annex D.2.11 for more information.

```
hdlcGroupAddress  OBJECT-TYPE
SYNTAX      INTEGER
ACCESS      read-write
```

```
STATUS       deprecated
DESCRIPTION
      "A group address for the data link layer. For PMPP, the syntax is
      an 8 or 16 bit entry with the second low order bit set to a one
      indicating that this is a group address.
      <Object Identifier> 1.3.6.1.4.1.1206.4.1.2.3.2.1.2"
REFERENCE
"NEMA TS 3.3 Clause 3.3.3.1"
::= { hdlcGroupAddressEntry 2}
```

2.7.2.3 HDLC Group Address Number Parameter

```
hdlcGroupAddressNumber  OBJECT-TYPE
SYNTAX      INTEGER (0..62)
ACCESS      read-write
STATUS      mandatory
DESCRIPTION
      "<Definition>A group address number prior to any encoding for the
      data link layer. The address of 63 is reserved for the all
      stations address. The value of zero (0) shall disable this row of
      the table.

      NOTE that in PMPP all group addresses are encoded in one byte.
      <Object Identifier> 1.3.6.1.4.1.1206.4.1.2.3.2.1.3"
REFERENCE
"NTCIP 2101"
DEFVAL  { 0 }
::= { hdlcGroupAddressEntry 3}
```

2.8 SECURITY NODE

```
-- NOTE: The objects under this heading have been moved to NTCIP 1103 v02.

END -- NTCIP1201-2008 DEFINITIONS
```

2.9 NEW AUXILIARY I/O OBJECTS IN NTCIP 1201 V03

```
-- ***************************************************************************
-- Filename:    1201v0315 AuxIO2.mib
-- Description: This MIB represents the data elements that first appeared in
--              NTCIP 1201 v02.31.  In moving the objects from NTCIP 1203 to
--              NTCIP 1201 v02.31, the objects were registered under a new
--              node but were not given new names. For backward compatibility,
--              this MIB retains the registration used in NTCIP 1201 v02.31
--              but changes the names of the objects.  In the context of
--              implementations that supported NTCIP 1201 v02.31, there are no
--              changes except for the object names.
-- Source:      NTCIP 1201 v03.09
-- MIB Revision History:
-- 07/12/06     Created this file from NTCIP 1201 v01
--              Changed formatting so that spacing appears correctly in the
--              text format of the MIB
--              Added but commented registration names and IMPORTS if one uses
--              the new NTCIP 8004 v02 SMI structure
--              Added Header and DESCRIPTION subfields per NTCIP 8004 v02
--              Changed all object to deprecated
-- 10/02/06     Changed all STATUSes to mandatory to eliminate checking errors
--              Revised object names
--              Organized objects as standalone MIB
-- 09/18/07     Updated this header
-- 03/00/11     Edited for publication; updated this header.
```

```
--
--Copyright 2010 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 in whole or in part in any form, translation into other
--languages and display are reserved by the copyright owners 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 for the MIB, Do not copy without written permission of either AASHTO,
--ITE, or NEMA.
--
--                      Joint NEMA, AASHTO, and ITE
--                   NTCIP Management Information Base
--                         DISTRIBUTION NOTICE
--
--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 sub-range 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.
--
--NTCIP is a trademark of AASHTO/ITE/NEMA.
--*************************************************************************

AuxIOv2-1201 DEFINITIONS ::= BEGIN

IMPORTS
   OBJECT-TYPE
       FROM RFC-1212
   DisplayString
       FROM RFC1213-MIB
   global
       FROM NTCIP8004 v02;

auxIOv2  OBJECT IDENTIFIER ::= { global 7}
-- <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.7
-- This node is an identifier used to group all objects supporting auxiliary
-- I/O functions
--
-- NOTE:  The auxiliary I/O management objects listed in NTCIP 1201 v03 define
a
-- mechanism for the support of unspecified I/O for an NTCIP device.
-- The agency  or device specifications define the intended operation
-- of these ports.
--
-- NOTE: NTCIP 1201 v01 defined functionality of these
-- objects under the experimental node.  For the purposes of
-- backward compatibility, the names of these  objects, originally defined in
-- NTCIP 1201 v02, have been modified to eliminate any conflicts/
-- confusion with the objects defined under the experimental node.  For those
-- agents that may support these objects and those originally defined under
-- the experimental node (see Section 2.10), the object definitions aretreated
-- as aliases such that a write to an object in one group acts as write
-- to the corresponding object in the other group.  As aliases, a read of an
-- object in this group is equivalent to a read of the corresponding object in
-- the auxIO group.
--
-- NOTE:
-- Early NTCIP deployments included the Aux I/O objects defined in NTCIP
-- 1203v01 located under an experimental node.  These objects were moved to a
-- permanent node with the release of NTCIP 1201 v02 and given new
-- names.  This can create confusion and backward compatibility issues.
-- As noted in the object definition, both sets of objects refer to the
-- same functions within the device; hence, both sets of objects cause the
-- same device action or provide the same device status.
-- Agency specifications that do NOT require support for the Aux I/O
-- objects under the experimental node should explicitly exclude the support
-- for these experimental objects (which have been deprecated) for
-- backward compatibility.  Support of the Aux I/O objects under the
-- permanent node identified in NTCIP 1201 v03 may be optional
-- or mandatory depending on agency or device specifications.
--
--Use the PRL to exclude support of NTCIP 1201 v01-defined aux I/O objects.
--
-- The relationship between mandatory and optional support of NTCIP v01
(experimental)
-- and NTCIP v02 objects is unique to the Aux I/O objects.
--
```

2.9.1 Maximum Number of Digital Auxiliary I/Os Parameter

```
maxAuxIOv2TableNumDigitalPorts OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   mandatory
   DESCRIPTION
      "<Definition> The number of rows contained in the 'auxIOv2Table' with
       the auxIOv2PortType set to 'digital'.
       <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.7.1"
```

```
::= { auxIOv2 1}
```

2.9.2 Maximum Number of Analog Auxiliary I/Os Parameter

```
maxAuxIOv2TableNumAnalogPorts OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   mandatory
   DESCRIPTION
      "<Definition>The number of rows contained in the 'auxIOv2Table’
       with the auxIOv2PortType set to 'analog’.
      <Unit>port
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.7.2"
::= {auxIOv2 2}
```

2.9.3 Auxiliary I/O Table Parameter

```
auxIOv2Table   OBJECT-TYPE
   SYNTAX   SEQUENCE OF AuxIOv2Entry
   ACCESS   not-accessible
   STATUS   mandatory
   DESCRIPTION
      "<Definition>A table providing the means to access any non-
       mission-critical or non-safety-related auxiliary I/O of the
       controller, including reading inputs and setting outputs.
       The number of rows in this table equals the sum of the values
       of the ‘maxAuxIOv2TableNumDigitalPorts’ and
       ‘maxAuxIOv2TableNumAnalogPorts’ objects.
       This table shall not be used to control or monitor any safety
       related equipment.  The electrical levels used by the ports are
       not standardized by auxIOv2Table objects; such information should
       be contained in the hardware manual.
      <TableType> static
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.7.3"
::= { auxIOv2 3}

auxIOv2Entry OBJECT-TYPE
   SYNTAX   AuxIOv2Entry
   ACCESS   not-accessible
   STATUS   mandatory
   DESCRIPTION
      "<Definition>Parameters of the auxiliary I/O table.
      <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.7.3.1"
INDEX {auxIOv2PortType, auxIOv2PortNumber}
::={auxIOv2Table 1}

AuxIOv2Entry ::= SEQUENCE {
   auxIOv2PortType               INTEGER,
   auxIOv2PortNumber             INTEGER,
   auxIOv2PortDescription        DisplayString,
   auxIOv2PortResolution         INTEGER,
   auxIOv2PortValue              INTEGER,
   auxIOv2PortDirection          INTEGER,
   auxIOv2PortLastCommandedState INTEGER
   }
```

2.9.3.1 Auxiliary Port Type Parameter

```
auxIOv2PortType OBJECT-TYPE
   SYNTAX   INTEGER{
                   other (1),
                    analog (2),
                    digital (3)
                   }
   ACCESS   read-only
   STATUS   mandatory
   DESCRIPTION
      "<Definition>Indicates the type of auxiliary I/O, which can be
       analog or digital.
      <Information> From NTCIP 1201 v01 to NTCIP v02 of these objects, it was
      determined that ports are either digital, analog, or other.
       <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.7.3.1.1"
::= {auxIOv2Entry 1}
```

2.9.3.2 Auxiliary Port Number Parameter

```
auxIOv2PortNumber OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   mandatory
   DESCRIPTION
      "<Definition>Indicates the port number for the associated port
       type.  Port numbers are used sequentially from one to max for
       each port type.  There can be a port 1 for analog port and port 1
       for digital port.
       <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.7.3.1.2"
::= {auxIOv2Entry 2}
```

2.9.3.3 Auxiliary Description Parameter

```
auxIOv2PortDescription  OBJECT-TYPE
   SYNTAX   DisplayString (SIZE (0..255))
   ACCESS   read-write
   STATUS   mandatory
   DESCRIPTION
      "<Definition>Informational text field describing the device at the
       associated auxiliary I/O.
       <Informative> In NTCIP 1203 v01, the SYNTAX SIZE was listed
       as (0..50). In NTCIP 1201 v02 and NTCIP 1201 v03, this was
changed to
       0..255).
       <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.7.3.1.3"
::= {auxIOv2Entry 3}
```

2.9.3.4 Auxiliary Resolution Parameter

```
auxIOv2PortResolution  OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   mandatory
   DESCRIPTION
      "<Definition>Defines number of bits used for the IO-port (e.g.
       width of digital, resolution of analog). Thus, this feature
       allows the digital monitoring (via NTCIP) of an analog port on
       the agent.
       <Informative> In NTCIP 1203 v01, ACCESS was listed as read-write;
       however, in NTCIP 1201 v03, ACCESS changed to read-only .  This changed
       because resolution is fixed by the hardware implementation and cannot
       be changed by the management station.
       The SYNTAX also changed from NTCIP 1201 v02 to NTCIP 1201 v03; it is
       now as it was originally under the experimental node defined in
       NTCIP 1203v01.  This changed to address backward compatibility and the
       ‘aliasing’ between the version 1 objects (see Section 2.10) and the
       Version 02 objects.
       <Unit>bit
       <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.7.3.1.4"
::= {auxIOv2Entry 4}
```

2.9.3.5 Auxiliary Value Parameter

```
auxIOv2PortValue  OBJECT-TYPE
   SYNTAX   INTEGER (0..4294967295)
   ACCESS   read-write
   STATUS   mandatory
   DESCRIPTION
      "<Definition>For input or bidirectional ports, this contains the
       current value of the input.  For output ports, this is the last
       commanded value of the port.  A genError shall be generated, if
       this object is set and the port is an input. The actual value
       exchanged shall not exceed [2^(auxIOv2PortResolution) – 1]; any
       SET operation to a value in excess of this number shall result in
       a genErr, and any GET response in excess of this value shall be
       considered erroneous.
       <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.7.3.1.5"
::= {auxIOv2Entry 5}
```

2.9.3.6 Auxiliary Port Direction Parameter

```
auxIOv2PortDirection OBJECT-TYPE
   SYNTAX   INTEGER {
                      output (1),
                      input (2),
                      bidirectional (3)}
   ACCESS   read-only
   STATUS   mandatory
   DESCRIPTION
      "<Definition>Indicates whether state of this port can be set (output),
       read (input) or both (bidirectional).
       <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.7.3.1.6"
::= {auxIOv2Entry 6}
```

2.9.3.7 Auxiliary Port Last Commanded State Parameter

```
auxIOv2PortLastCommandedState OBJECT-TYPE
   SYNTAX   INTEGER (0..4294967295)
   ACCESS   read-only
   STATUS   mandatory
   DESCRIPTION
      "<Definition>For bi-directional ports, this object indicates
       the last state to which the auxIOv2PortValue object was set.  For
       output ports, this value shall always be equal to the
       auxIOv2PortValue object.  For input ports, this value shall
       always be zero (0).
       <Object Identifier> 1.3.6.1.4.1.1206.4.2.6.7.3.1.7"
::= {auxIOv2Entry 7}

END -- AuxIOv2-1201 DEFINITIONS
```

2.10 OLD AUXILIARY I/O OBJECTS FROM NTCIP 1203:1996 (NTCIP 1201 V01)

```
-- ***************************************************************************
-- Filename:    1201v0315 AuxIO1.mib
-- Description: This MIB represents the data elements that appeared in
--              NTCIP 1203:1997.  In moving the objects to NTCIP 1201 v01 and
--              registering them under the global node, new definitions had to
--              be created and therefore the original definitions had to be
--              deprecated.  They have been moved here for purposes of
--              maintaining backward compatibility.  In the context of
--              implementation that supports these objects, there is no
--              difference between what appeared in NTCIP 1203:1997 and what
--              appears here.
-- Source:      NTCIP 1201 v03.09
-- MIB Revision History:
-- 07/12/06     Created this file from original standard
--              Change formatting so that spacing appears correctly in the
--              text format of the MIB
--              Added but commented registration names and IMPORTS if one uses
--              the new NTCIP 8004 v01 SMI structure
--              Added Header and DESCRIPTION subfields per NTCIP 8004 v01
--              Changed all object to deprecated
-- 11/20/06     Revised filename and source information
-- 09/18/07     Updated this header
-- 03/00/11     Edited for publication; updated this header.
--
--Copyright 2010 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 in whole or in part in any form, translation into other
--languages and display are reserved by the copyright owners 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 for the MIB, Do not copy without written permission of either AASHTO,
--ITE, or NEMA.
--
--                     Joint NEMA, AASHTO, and ITE
--                  NTCIP Management Information Base
--                        DISTRIBUTION NOTICE
--
--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 sub-range 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.
--
--NTCIP is a trademark of AASHTO/ITE/NEMA.
--****************************************************************************

AuxIOv1-1203 DEFINITIONS ::= BEGIN

IMPORTS
   OBJECT-TYPE
      FROM RFC-1212
   DisplayString
      FROM RFC1213-MIB
--   experimental
--      FROM NEMA_SMI
   nemaExperimental
      FROM NTCIP8004 v01;
-- Replace the previous 2 uncommented lines above with the commented lines
-- when using NTCIP 8004 v01 SMI

--   exp-global OBJECT IDENTIFIER ::= {experimental 1}
   expGlobal OBJECT IDENTIFIER ::= {nemaExperimental 2}
-- Replace the previous uncommented line above with the commented line when
-- using NTCIP 8004 v01.37 SMI

--   auxiliaryIO OBJECT IDENTIFIER ::= { exp-global 1}
   auxiliaryIO  OBJECT IDENTIFIER ::= { expGlobal 1}
-- Replace the previous uncommented line above with the commented line when
-- using NTCIP 8004 v01.37 SMI
--
-- NOTE:  The auxiliary I/O management objects listed herein define a
-- mechanism for the support of unspecified I/O for an NTCIP device.
-- The agency or device specifications should define the intended operation
-- of these ports.
--
-- NOTE: These objects are still logically located under the nemaExperimental
-- node and use their originally defined textual names and OIDs.  For the
-- purposes of backward compatibility, the object STATUS has been changed to
-- deprecated. For those agents that may support these objects and the new
-- objects under the global node (see Section 2.9), the object definitions
-- shall be treated as aliases in that a write to an object in one group acts
-- as write to the corresponding object in the other group. As aliases, a read
-- of an object in one group also acts as read of the corresponding
-- object in the other group.
-- NOTE:
-- Early NTCIP deployments included the Aux I/O objects defined in NTCIP
-- 1203 v01 located under an experimental node.  These objects were moved to a
-- permanent node with the release of NTCIP 1201 v02 and given new
-- names.  This can create confusion and backward compatibility issues.
-- As noted in the object definition, both sets of objects refer to the
-- same functions within the device; hence, both sets of objects cause the
-- same device action or provide the same device status.
-- Agency specifications which do NOT require support for the Aux I/O
-- objects under the experimental node should exclude the support
-- for these experimental objects (which have been deprecated) to ensure
-- backward compatibility.  Support of the Aux I/O objects under the
-- permanent node identified in NTCIP 1201 v03 may be optional
-- or mandatory depending on the agency- or project specification.
--
--Use the PRL to exclude support of NTCIP v01-defined aux I/O objects.
--
-- The relationship between mandatory and optional support of NTCIP 1201 v01
-- (experimental) and NTCIP 1201 v02 objects is unique to the Aux I/O objects.
--
```

2.10.1 Maximum Number of Digital Auxiliary I/Os Parameter

```
maxAuxIODigital OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   deprecated
   DESCRIPTION
      "<Definition> The number of rows contained in the 'auxIOTable' with the
       auxIOPortType set to 'digital'.
       <Object Identifier> 1.3.6.1.4.1.1206.2.1.1.1 "
::= {auxiliaryIO 1}
```

2.10.2 Maximum Number of Analog Auxiliary I/Os Parameter

```
maxAuxIOAnalog OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   deprecated
   DESCRIPTION
      "<Definition>The number of rows contained in the 'auxIOTable' with the
       auxIOPortType set to 'analog'.
       <Object Identifier> 1.3.6.1.4.1.1206.2.1.1.2 "
::= {auxiliaryIO 2}
```

2.10.3 Auxiliary I/O Table Parameter

```
auxIOTable  OBJECT-TYPE
   SYNTAX   SEQUENCE OF AuxIOEntry
   ACCESS   not-accessible
   STATUS   deprecated
   DESCRIPTION
      "<Definition> A table providing the means to access the auxiliary I/O of
       the Controller, including reading inputs and setting outputs.
       A maximum of 255 auxiliary I/Os may be defined for all, digital,
       analog or other types of ports.
       <TableType> static
       <Object Identifier> 1.3.6.1.4.1.1206.2.1.1.3 "
::= { auxiliaryIO 3}

auxIOEntry OBJECT-TYPE
   SYNTAX   AuxIOEntry
   ACCESS   not-accessible
   STATUS   deprecated
   DESCRIPTION
      "<Definition> Parameters of the auxiliary I/O table.
       <Object Identifier> 1.3.6.1.4.1.1206.2.1.1.3.1 "
   INDEX {auxIOPortType, auxIOPortNumber}
::={auxIOTable 1}
```

```
AuxIOEntry ::= SEQUENCE {
   auxIOPortType        INTEGER,
   auxIOPortNumber      INTEGER,
   auxIODescription     DisplayString,
   auxIOResolution      INTEGER,
   auxIOValue           INTEGER,
   auxIOPortDirection   INTEGER
   }
```

2.10.3.1 Auxiliary Port Type Parameter

```
auxIOPortType OBJECT-TYPE
   SYNTAX   INTEGER{
                   other (1),
                   analog (2),
                   digital (3)
                   }
   ACCESS   read-only
   STATUS   deprecated
   DESCRIPTION
      "<Definition> Indicates the type of auxiliary I/O, which may be analog,
       digital or other.
       <Object Identifier>  1.3.6.1.4.1.1206.2.1.1.3.1.1 "
::= {auxIOEntry 1}
```

2.10.3.2 Auxiliary Port Number Parameter

```
auxIOPortNumber OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   deprecated
   DESCRIPTION
      "<Definition> Indicates the port number for the associated port type.
       Port numbers are used sequentially from one to max for each port
       type.  There can be a port 1 for analog port and port 1 for digital
       port.
       <Object Identifier> 1.3.6.1.4.1.1206.2.1.1.3.1.2"
::= {auxIOEntry 2}
```

2.10.3.3 Auxiliary Description Parameter

```
auxIODescription OBJECT-TYPE
   SYNTAX   DisplayString (SIZE (0..50))
   ACCESS   read-write
   STATUS   deprecated
   DESCRIPTION
      "<Definition> Informational text field describing the device at the
       associated auxiliary I/O
       <Informative> In NTCIP 1203 v01, the SYNTAX SIZE was listed
       as (0..50).  In all versions of NTCIP 1201 v02, auxIO2Description (this
       object's alias) was changed to (0..255).  This does not present a
       backward compatibility issue if a NTCIP 1201 v02 management station
       limits the size of the DisplayString to 50 characters.
       <Object Identifier> 1.3.6.1.4.1.1206.2.1.1.3.1.3 "
::= {auxIOEntry 3}
```

2.10.3.4 Auxiliary Resolution Parameter

```
auxIOResolution OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   deprecated
   DESCRIPTION
      "<Definition> Defines number of bits used for the IO-port (e.g. width of
       digital, resolution of analog).
       <Informative> In NTCIP 1203 v01, the ACCESS was listed as read-
       write. Resolution is fixed by the hardware implementation and cannot be
       changed by the management station.
       <Unit>bit
       <Object Identifier> 1.3.6.1.4.1.1206.2.1.1.3.1.4 "
::= {auxIOEntry 4}
```

2.10.3.5 Auxiliary Value Parameter

```
auxIOValue OBJECT-TYPE
   SYNTAX   INTEGER (0..4294967295)
   ACCESS   read-write
   STATUS   deprecated
   DESCRIPTION
      "<Definition> For input or bidirectional ports, this contains the
       current value of the input.  For output ports, this is the last
       commanded value of the port.  A genError shall be generated, if this
       object is set and the port is an input.
       <Object Identifier> 1.3.6.1.4.1.1206.2.1.1.3.1.5 "
::= {auxIOEntry 5}
```

2.10.3.6 Auxiliary Port Direction Parameter

```
auxIOPortDirection OBJECT-TYPE
   SYNTAX   INTEGER {
                   output (1),
                   input (2),
                   bidirectional (3)}
   ACCESS   read-only
   STATUS   deprecated
   DESCRIPTION
      "<Definition> Indicates whether state of this port can be set (output),
       read (input) or both (bidirectional).
       <Informative > The ACCESS has been changed from what originally
       appeared in NTCIP 1203 v01 because it was an error.
       <Object Identifier> 1.3.6.1.4.1.1206.2.1.1.3.1.6 "
::= {auxIOEntry 6}

END  -- AuxIOv1-1203 DEFINITIONS
```

Section 3
CONFORMANCE

NOTE—The conformance requirements previously included in NTCIP 1201 v01 have been removed. NTCIP 1201 v03 only defines the data that may be useful for a given device; any requirements for supporting a specific piece of data is defined in NTCIP device-specific standards.

Annex A
CONCEPT OF OPERATIONS
[NORMATIVE]

Annex A provides examples of how a management station may interface with a device complying with NTCIP 1201 v03. Any device claiming conformance to the NTCIP 1201 v03 features depicted (download transaction, set time, and configure schedule) shall support the exchanges as shown. However, the flexible design of the NTCIP protocols allows a large number of other possibilities, and these figures do not limit any other requirements of NTCIP standards. These diagrams promote a common understanding of how systems may be designed to increase the likelihood of interchangeability in deployed systems.

Three use cases are shown in Figure 2.

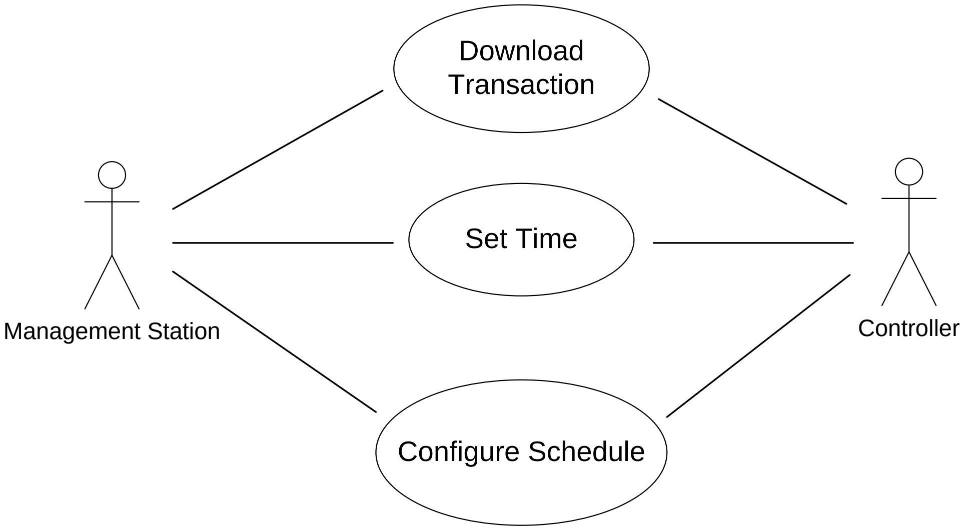


Figure 2 Global Use Cases

A.1 DOWNLOAD TRANSACTION USE CASE

The first use case is for a download transaction. The intent of this use case is that a management station has a need to download several inter-related parameters to the controller. Because the parameters are inter-related, the parameters shall be set simultaneously for the controller to validate the set operation (e.g., the download may consist of a set of parameters, whose sum shall equal the sum of another set of parameters. and the management station wishes to change the sum for both sets).

The parameters that require the use of the transaction mode are device-specific. Some devices may not require support of the transaction feature, while other devices may require SET operations on any database object to be within the transaction mode.

When used, the feature allows a device to buffer a series of set operations on database parameters and to implement all operations simultaneously to properly perform controller consistency checks.

The normal, fault-free process is shown in Figure 3.

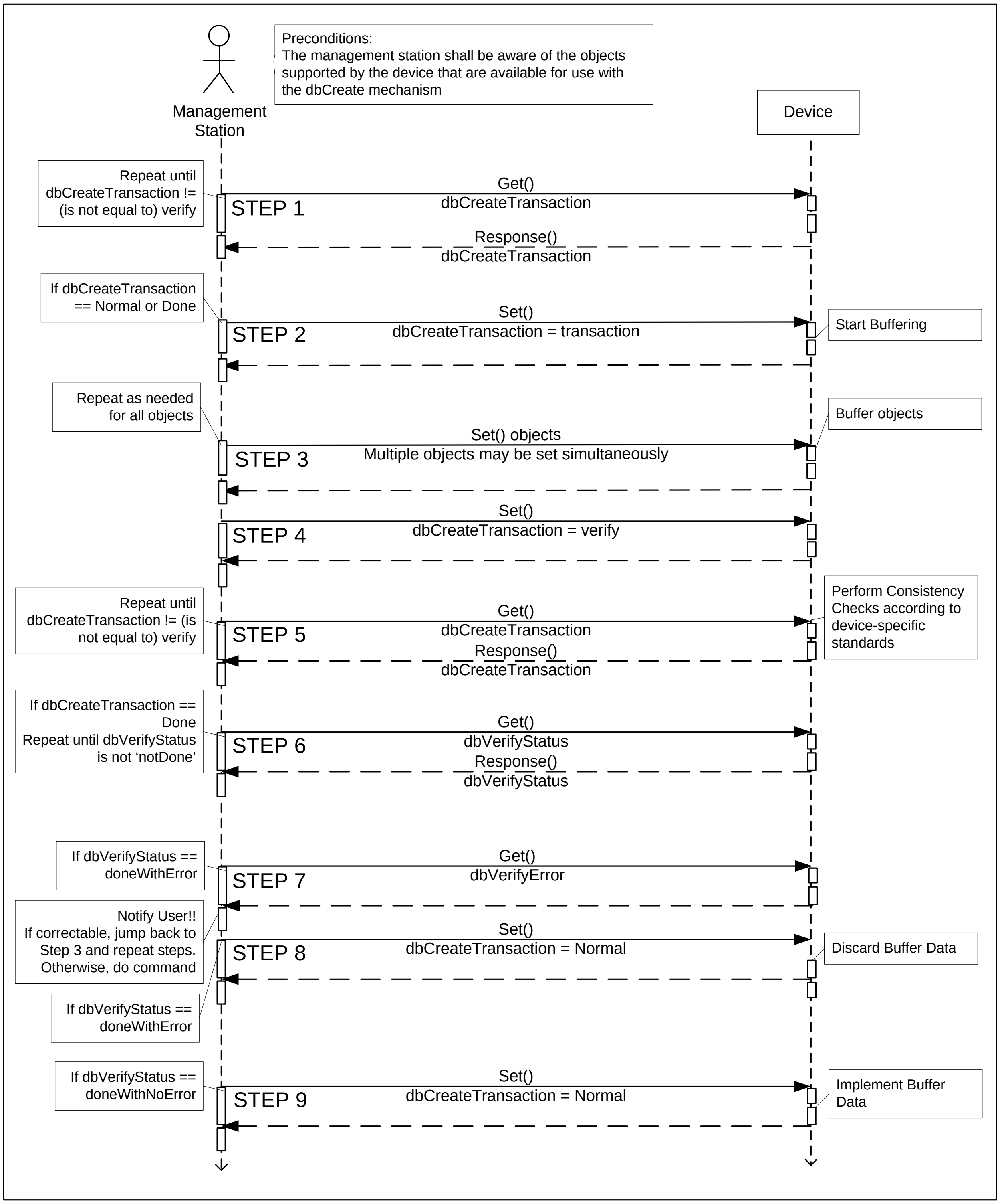

Preconditions:
The management station shall be aware of the objects supported by the device that are available for use with the dbCreate mechanism
Management Station
Device
Repeat until dbCreateTransaction != (is not equal to) verify
STEP 1
Get() dbCreateTransaction
Response() dbCreateTransaction
If dbCreateTransaction == Normal or Done
STEP 2
Set() dbCreateTransaction = transaction
Start Buffering
Repeat as needed for all objects
STEP 3
Set() objects Multiple objects may be set simultaneously
Buffer objects
STEP 4
Set() dbCreateTransaction = verify
Repeat until dbCreateTransaction != (is not equal to) verify
STEP 5
Get() dbCreateTransaction
Response() dbCreateTransaction
Perform Consistency Checks according to device-specific standards
If dbCreateTransaction == Done Repeat until dbVerifyStatus is not 'notDone'
STEP 6
Get() dbVerifyStatus
Response() dbVerifyStatus
If dbVerifyStatus == doneWithError
STEP 7
Get() dbVerifyError
Notify User!! If correctable, jump back to Step 3 and repeat steps. Otherwise, do command
STEP 8
Set() dbCreateTransaction = Normal
Discard Buffer Data
If dbVerifyStatus == doneWithError
If dbVerifyStatus == doneWithNoError
STEP 9
Set() dbCreateTransaction = Normal
Implement Buffer Data

Figure 3 Download Transaction Process Sequence Diagram

Within this mode, the controller operates as a state machine as described in the definition of dbCreateTransaction (see Section 2.3.1). Figure 4 supplements this definition and is a state diagram that provides a formal Unified Modeling Language (UML) representation of the state machine. See Annex B.1 for a Controller Class Diagram in UML notation.

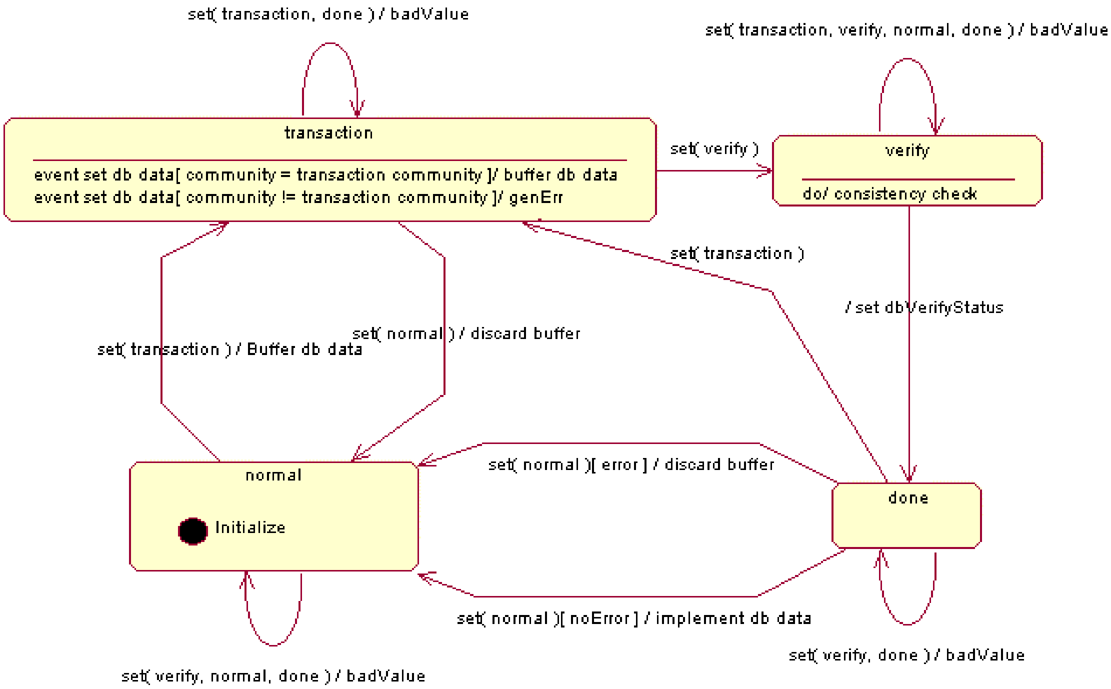

set(transaction, done) / badValue
set(transaction, verify, normal, done) / badValue
transaction
event set db data[community = transaction community]/ buffer db data
event set db data[community != transaction community]/ genErr
set(verify)
verify
do/ consistency check
set(transaction)
/ set dbVerifyStatus
set(normal) / discard buffer
set(transaction) / Buffer db data
normal
Initialize
set(normal)[error] / discard buffer
done
set(normal)[noError] / implement db data
set(verify, done) / badValue
set(verify, normal, done) / badValue

Figure 4 Controller State Diagram

A.2 SET TIME

The second use case is to set the time in a controller. Four key parameters affect the local time stored in the controller:

a) globalTime (which is time in Coordinated Universal Time (UTC))
b) globalDaylightSaving
c) begin and end DST objects
d) controllerStandardTimeZone (which is the offset between local Standard Time and UTC)

Each of these parameters is independent from one another and, thus, a controller shall allow a management station to set any or all of these parameters in any order using one or more set operations and may additionally combine these parameters in any fashion with other parameters.

When setting any one of these values, the indicated object shall be set to the indicated value and the value of controllerLocalTime shall be updated by the controller to reflect this new value, but none of the other time objects shall be affected.

A.2.1 Examples—Operation of the Daylight Saving Time (DST) Mechanism

Figures 5 through 7 (and, arguably, Figure 8) and Table 1 further document how DST is to be implemented.

NOTE—The examples shown are unusual programming examples to demonstrate the flexibility of the DST functionality.

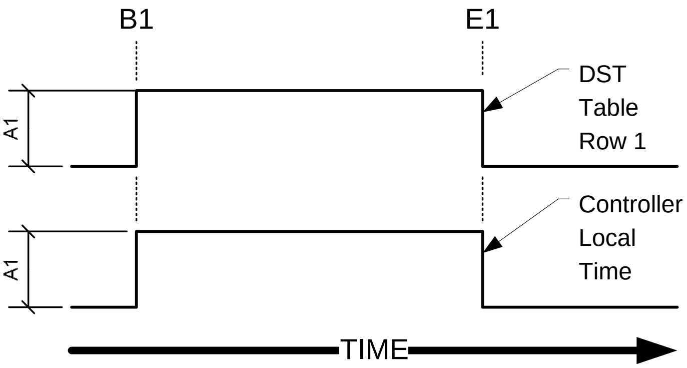


Figure 5 Example 1—DST Configuration (Current DST Usage)

Figure 5 shows that the DST Table contains 1 row. This row defines the date and time at which DST starts and ends. Row 1 includes a beginning date/time at B1 and an ending date/time at E1 with an adjustment time of A1 (assume 60 minutes). This example shows the current operation of DST in the US and most countries in the world.

The lower portion of Figure 5 shows what happens to the local controller time, which is adjusted by the adjustment time A1.

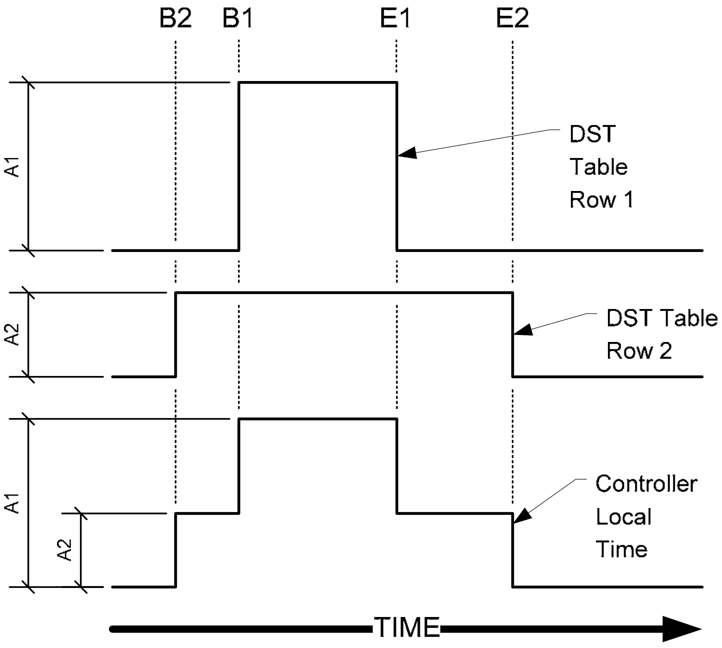


Figure 6 Example 2—Multi-Step DST Configuration (Proposed DST Usage)

Figure 6 shows that the DST Table contains 2 rows. Each row defines different dates/times at which DST starts and ends. Row 1 includes a beginning date/time at B1 and an ending date/time at E1 with an adjustment time of A1 (assume 60 minutes). Row 2 includes a beginning date/time at B2 and an ending date/time at E2 with an adjustment time of A2 (assume 30 minutes). This example shows a proposed operation of DST, where DST is phased in and phased out.

The bottom portion of Figure 6 shows what happens when each of these 2 rows are activated, i.e., that the

the actual DST in effect is increased by 30 minutes when Row 2 is activated, and goes back down to 30 minutes once the end date/time of Row 2 has been reached.

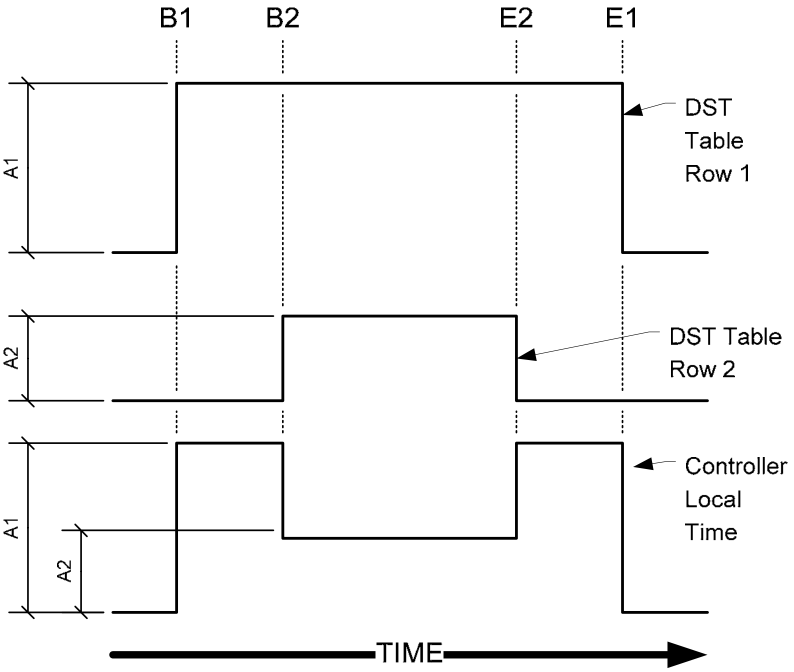


Figure 7 Example 3—DST Configuration (Non-Realistic)

Figure 7 shows that the DST Table contains 2 rows. Each row defines different dates/times at which DST starts and ends. Row 1 includes a beginning date/time at B1 and an ending date/time at E1 with an adjustment time of A1 (assume 60 minutes). Row 2 includes a beginning date/time at B2 and an ending date/time at E2 with an adjustment time of A2 (assume 30 minutes). (This example is not a real-life example in the US).

Lastly, Figure 7 shows on the bottom what happens when each of these 2 rows are activated, i.e., that the actual DST in effect is reduced to 30 minutes when Row 2 is activated, and goes back up to 60 minutes once the end date/time of Row 2 has been reached.

	Before B1	**B1**	**B2**	**E2**	**E1**	**After E1**
Controller Local Time	TOD	TOD+A1	TOD+A2	TOD+A1	TOD	TOD
Event Description	No DST Table Row Active	DST Table Row 1 becomes Latest Active Event	DST Table Row 2 becomes Latest Active Event	DST Table Row 2 event Ends DST Table Row 1 becomes Latest Active Event	DST Table event Row 1 Ends	No DST Table Row Active
TOD is Global Time with corrections for Controller Time Zone without DST adjustments						

Table 1 DST Configuration Figure Explanation

Table 1 shows what actions take place when events B1, B2, E1 and E2 are reached.

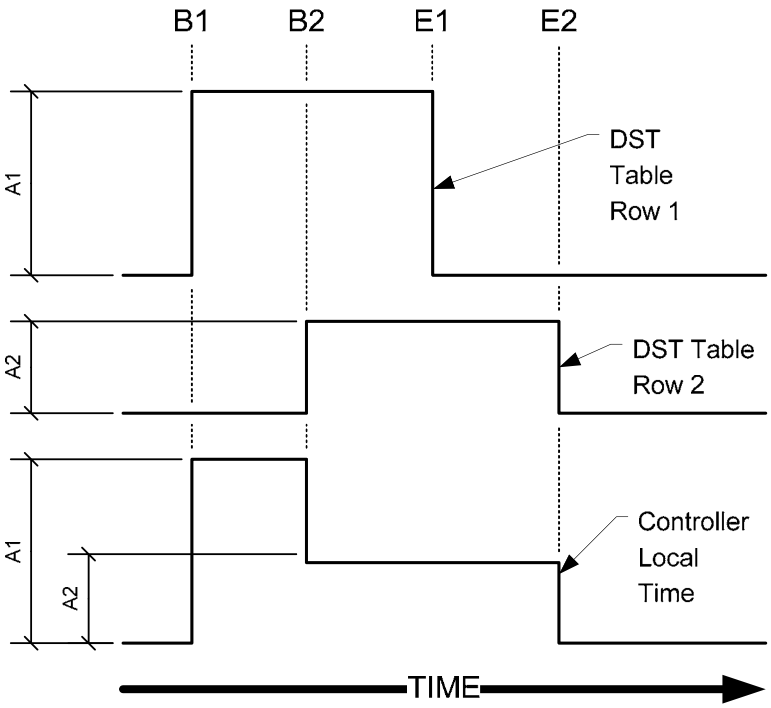


Figure 8 Example 4—DST Configuration (Non-Realistic)

Figure 8 shows that the DST Table contains 2 rows. Each row defines different dates/times at which DST starts and ends. Row 1 includes a beginning date/time at B1 and an ending date/time at E1 with an adjustment time of A1 (assume 60 minutes). Row 2 includes a beginning date/time at B2 and an ending date/time at E2 with an adjustment time of A2 (assume 30 minutes). (Note that this example is not a real-life example in the US).

Lastly, Figure 8 shows on the bottom what happens when each of these 2 rows are activated, i.e., that the actual DST in effect is reduced to 30 minutes when Row 2 is activated and terminates altogether once the end date/time of Row 2 has been reached.

In summary, the row with the latest *dstBegin* time, which has not terminated because of passing the *dstEnd* time, determines the setting of the local TOD clock; the *dstSecondsToAdjust* for the latest *dstBegin* governs the Local TOD clock settings.

A.2.2 Example 1—Changing Global Time

Original Values:
globalTime: 1023278400 (12:00 noon 5 June 2002)
globalDaylightSaving: disableDST
controllerStandardTimeZone: -21600
controllerLocalTime: 1023256800 (6:00 AM 5 June 2002)

Set Operation
globalTime: 1023282000 (1:00 PM 5 June 2002)

Updated Values:
globalTime: 1023282000 (1:00 PM 5 June 2002)

globalDaylightSaving: disableDST
controllerStandardTimeZone: -21600 (-6 hours)
controllerLocalTime: 1023260400 (7:00 AM 5 June 2002)

A.2.3 Example 2—Changing Daylight Saving Time (DST)

Original Values:
globalTime: 1023278400 (12:00 noon 5 June 2002)
globalDaylightSaving: disableDST
controllerStandardTimeZone: -21600
controllerLocalTime: 1023256800 (6:00 AM 5 June 2002)

Set Operation
dstBeginMonth: march (3)
dstBeginOccurrences: second (2)
dstBeginDayOfWeek: sunday (1)
dstBeginDayOfMonth: 1 (meaning the first day of the month)
dstBeginSecondsToTransition: 7200 (meaning 2 hours after midnight in seconds)
dstEndDayOfMonth: november (11)
dstEndOccurrences: first (1)
dstEndDayOfWeek: sunday (1)
dstEndDayOfMonth: 1 (meaning the first day of the month)
dstEndSecondsToTransition: 7200 (meaning 2 hours after midnight in seconds)
dstSecondsToAdjust: 3600 (meaning the offset between local time and DST is 1 hour at the beginning and the end of DST)
globalDaylightSaving: enableDaylightSavingNode (20)

Updated Values:
globalTime: 1023278400 (12:00 noon 5 June 2002)
globalDaylightSaving: enableDaylightSavingNode
dstBeginMonth: march
dstBeginOccurrences: second
dstBeginDayOfWeek: sunday
dstBeginDayOfMonth: 1
dstBeginSecondsToTransition: 7200
dstEndDSTMonth: november
dstEndDSTOccurrences: first
dstEndDSTDayOfWeek: sunday
dstEndDSTDayOfMonth: 1
dstEndDSTSecondsToTransition: 7200
dstSecondsToAdjust: 3600
controllerStandardTimeZone: -21600 (-6 hours)
controllerLocalTime: 1023260400 (7:00 AM 5 June 2002)

A.2.4 Example 3—Changing Time Zone

Original Values:
globalTime: 1023278400 (12:00 noon 5 June 2002)
globalDaylightSaving: disableDST
controllerStandardTimeZone: -21600 9 (-6 hours)
controllerLocalTime: 1023256800 (6:00 AM 5 June 2002)

Set Operation
ControllerStandardTimeZone: -18000

Updated Values:
globalTime: 1023278400 (12:00 noon 5 June 2002)

globalDaylightSaving: disableDST
controllerStandardTimeZone: -18000 (-5 hours)
controllerLocalTime: 1023260400 (7:00 AM 5 June 2002)

A.2.5 Example 4—Changing Multiple Parameters

Original Values:
globalTime: 1023278400 (12:00 noon 5 June 2002)
globalDaylightSaving: disabled
controllerStandardTimeZone: -21600
controllerLocalTime: 1023256800 (6:00 AM 5 June 2002)

Set Operation
globalTime: 1023282000 (1:00 PM 5 June 2002)
dstBeginMonth: march
dstBeginOccurrences: second
dstBeginDayOfWeek: sunday
dstBeginDayOfMonth: 1
dstBeginSecondsToTransition: 7200
dstEndMonth: november
dstEndOccurrences: first
dstEndDayOfWeek: sunday
dstEndDayOfMonth: 1
dstEndSecondsToTransition: 7200
dstSecondsToAdjust: 3600
controllerStandardTimeZone: -18000
globalDaylightSaving: enableDaylightSavingNode (20)

Updated Values:
globalTime: 1023282000 (1:00 PM 5 June 2002)
globalDaylightSaving: enableDaylightSavingNode
dstBeginMonth: march
dstBeginOccurrences: second
dstBeginDayOfWeek: sunday
dstBeginDayOfMonth: 1
dstBeginSecondsToTransition: 7200
dstEndMonth: november
dstEndOccurrences: first
dstEndDayOfWeek: sunday
dstEndDayOfMonth: 1
dstEndSecondsToTransition: 7200
dstSecondsToAdjust: 3600
controllerStandardTimeZone: -18000 (-5 hours)
controllerLocalTime: 1023267600 (9:00 AM 5 June 2002)

A.3 CONFIGURE SCHEDULER

One approach for configuring the scheduler is to use dbCreateTransaction to download the schedule parameters to the field device in any combination. Upon completion of the download, the downloaded information is 'verified' and stored in a device's database, if correct.

Other approaches for configuring the scheduler are defined in device-specific standards such as NTCIP 1202, 1203, and 1204 (if deployed).

NOTE—Deployment experience has shown that implementations operate differently in terms of execution of an action within a schedule. Some deployments implement a modification immediately, while others wait until the next occurrence of the schedule entry. NTCIP 1201 v03 does not define how a field device implements the operations.

The operational action according to the schedule shall be in effect within one (1) minute of setting the 'globalTime' parameter. See Section 2.4.4.3 for further information.

Annex B
CLASS DIAGRAMS
[INFORMATIVE]

Annex B provides an overview of NTCIP 1201 v03 data defined using UML Class Diagrams. The information presented in Annex B is defined elsewhere in NTCIP 1201 v03; however, these figures depict key characteristics of data definitions in a concise manner and are provided as a useful reference tool. The diagrams conform to the modeling conventions defined by ISO 14817, and were used to develop the ISO 14817 conforming Descriptive Names as shown within each object definition in Section 2. The ObjectClassTerm of the descriptive name is indicated by the name of each box within the figures and the propertyTerm is shown as an item within the box. These Descriptive Names are also used by the online ITS Data Registry as the primary name of each data concept.

NOTE—While the discussion in Annex B indicates that virtually every feature is optional, to claim conformance with various NTCIP device-specific standards, support for many of these features may be mandatory.

B.1 CONTROLLER CLASS DIAGRAM

Figure 9 depicts the components of data stored within a controller in UML notation.

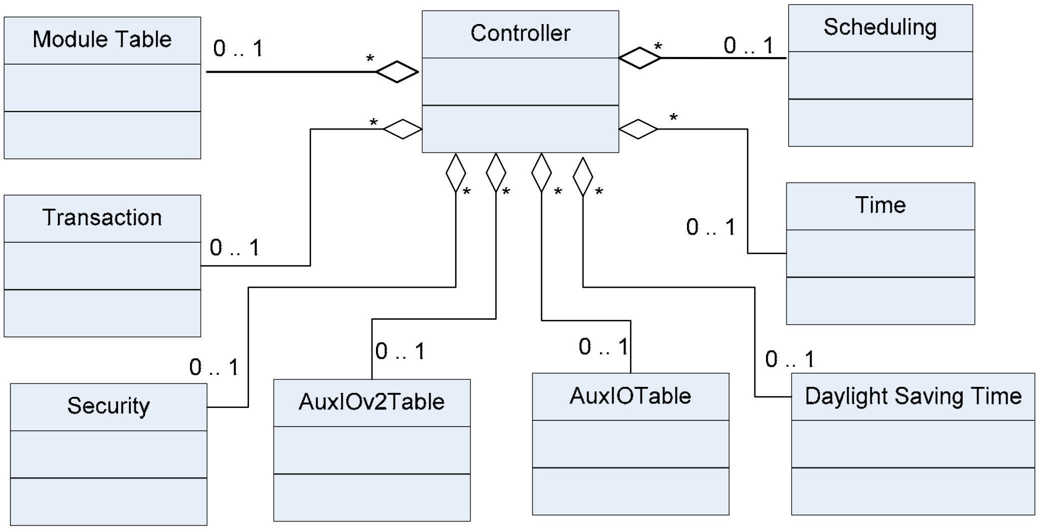


Figure 9 Controller Class Diagram

Figure 9 indicates that a Controller may or may not contain the following major components:

a) A module table
b) A transaction service
c) A definition of time
d) A definition of daylight saving time (DST)
e) A scheduling service

f) A definition of security objects
g) An auxiliary input/output table (NTCIP 1201 v01, NTCIP 1201 v02, or NTCIP 1201 v03 tables)

NOTE—Two different versions of the Aux I/O object definitions are currently defined. The first was defined under an experimental node of the global tree and was originally contained in NTCIP 1203 v01. Because of the applicability of the Aux I/O objects to more than dynamic message signs (DMS), these objects were moved to NTCIP 1201 v02, and moved from the experimental node to a permanent node under the global tree. To differentiate these two sets of objects, the objects associated with Aux I/O in NTCIP 1201 v02 have had ‘v2’ added in the object name.

The details of each class are defined through the Management Information Base provided in Section 2. More detailed class diagrams for each feature are provided in Annex C.

NOTE—There is another component of data that is currently defined in this standard. This component of data group contains the object definitions defined under the PMPP Object Node. However, this node and all its object definitions will ultimately be moved to NTCIP 2101. Therefore, this information is not included in the diagram above.

B.2 CONFIGURATION INFORMATION

Figure 10 depicts the configuration data stored by a controller.

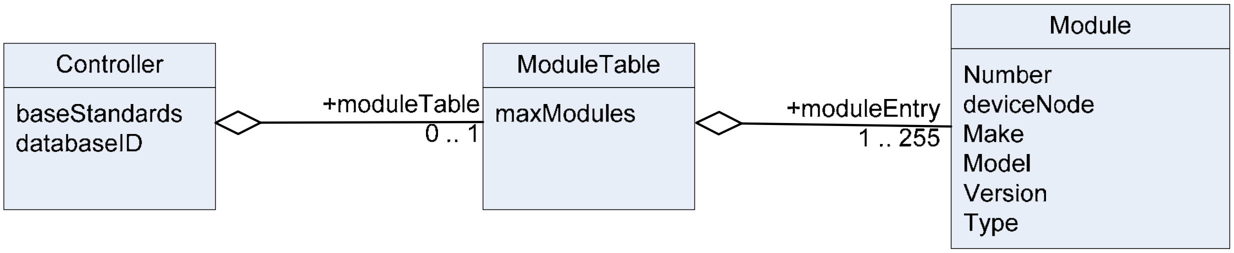


Figure 10 Class Diagram of the Configuration Data

Figure 10 indicates a controller may have a database identifier and zero or one module tables. If there is a module table, then the controller may additionally support an object defining the maximum number of modules supported within the table, which may be between one and 255, as indicated by the link to the Module class. For each module, the controller may support a variety of information, including:

a) The module number;
b) The device node to which the module relates;
c) The make of the module;
d) The model of the module;
e) The version of the module; and
f) The type of module.

B.3 TRANSACTION INFORMATION

Figure 11 depicts the transaction state data stored by a controller.

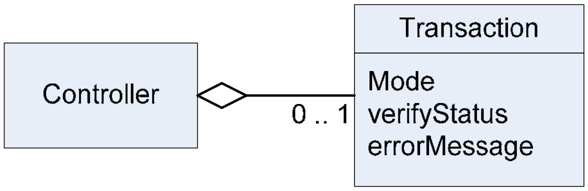


Figure 11 Class Diagram of the Transaction Service

A controller may support a transaction feature (see Figure 11). The following information characterizes the feature:

a) A mode;
b) A status; and
c) An error code.

B.4 TIME AND DAYLIGHT SAVING TIME (DST) INFORMATION

Figure 12 depicts the time-related and DST-related data stored by a controller.

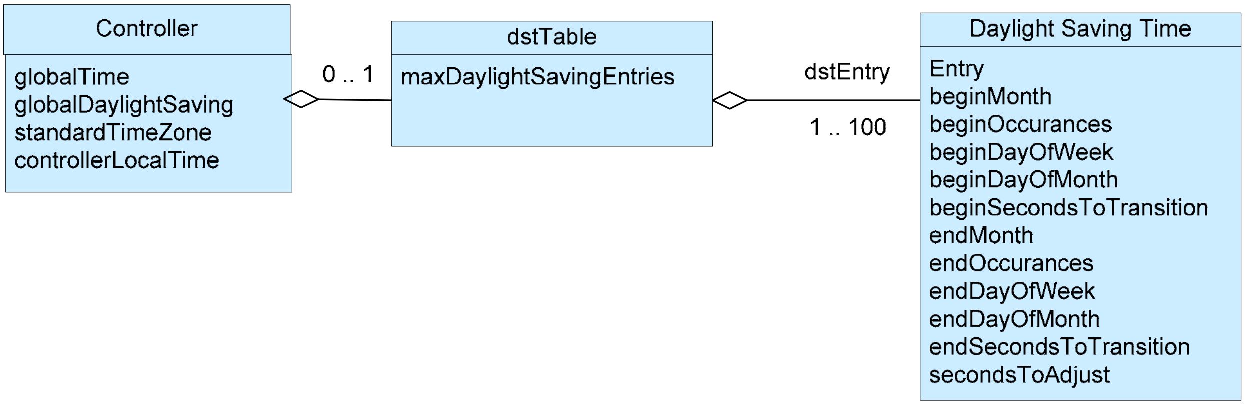


Figure 12 Class Diagram of Time/DST Information

Figure 12 indicates a controller may store time-related information, including:

a) The current time in UTC
b) An indication of the daylight saving mode
c) An indication of the time zone when in standard time
d) An indication of the local time, which includes and accounts for DST

The controller may also support a DST Table. If this is supported, it is characterized by the maximum number of entries that it may contain, which is required to be at least one and may be no greater than 100. For each entry, the following information may be stored:

a) A DST number
b) A begin DST month indicating in which month the DST may begin
c) A begin DST occurrences parameter indicating the number of occurrences of the specific day of week required to have occurred within the selected month before DST begins [NOTE—“beginOccurances” (sic) is misspelled in the figure, but is spelled correctly in the MIB object definitions.]

d) A begin DST day of week indicating on which day of week the DST may begin
e) A begin DST day of month indicating on which day of month the DST may begin

f) A begin DST Seconds to Transition parameter indicating after how many seconds after midnight of a particular day the DST may begin
g) An end DST month indicating in which month the DST may begin
h) An end DST occurrences parameter indicating number of occurrences of the specific day of week required to have occurred within the selected month before DST ends
i) An end DST day of week indicating on which day of week the DST may end
j) An end DST day of month indicating on which day of month the DST may end
k) An end DST Seconds to Transition parameter indicating after how many seconds after midnight of a particular day the DST may end
l) A seconds to adjust parameter indicating by how many seconds the DST time is offset from the local reference time when DST as defined by this entry is in effect

B.5 GENERIC SCHEDULE INFORMATION

Figure 13 depicts the generic schedule-related data stored by a controller.

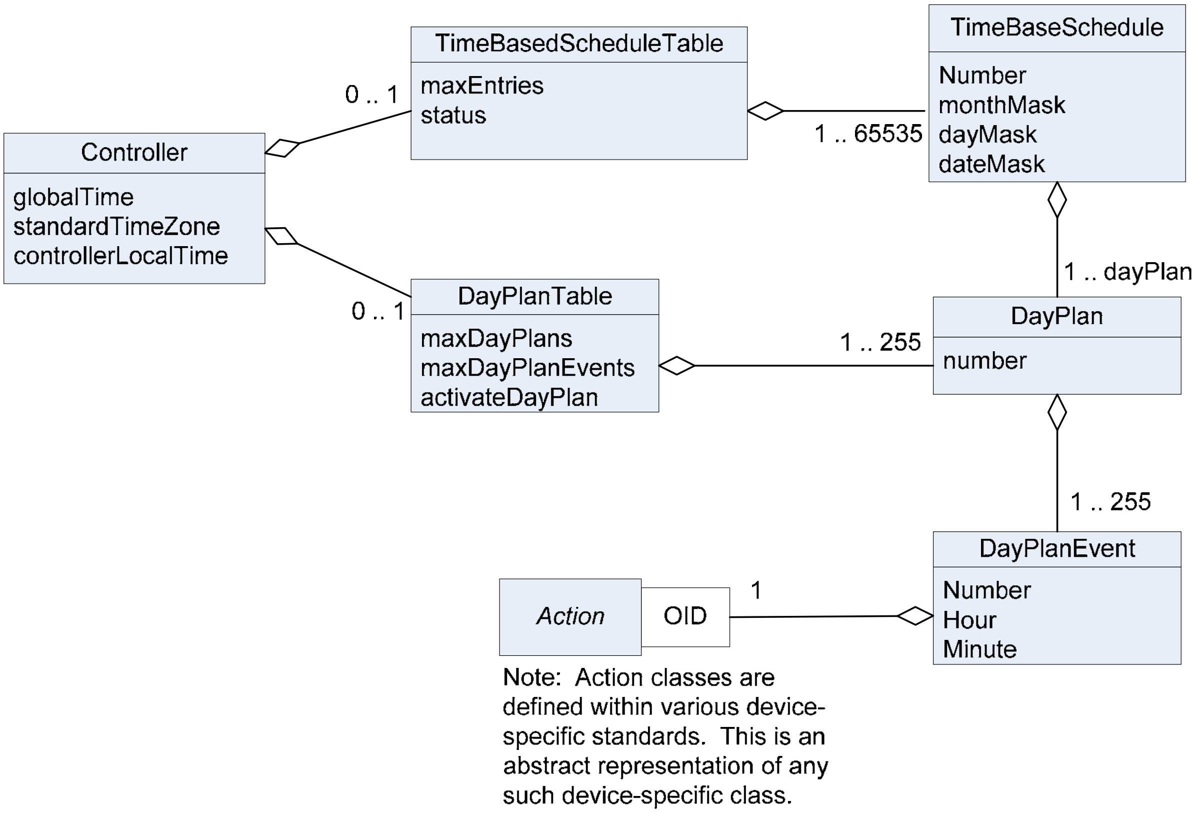


Figure 13 Class Diagram of Generic Schedule-Related Information

Figure 13 indicates a controller may store schedule information, including:

a) The current time in UTC
b) An indication of the time zone when in standard time
c) An indication of the local time, which includes and accounts for DST

The controller may also support a timebase schedule table. If supported, it is characterized by the maximum number of entries that it may contain, which is required to be at least one and may be no greater than 65535, and a status. For each entry, the following information may be stored:

a) A schedule number
b) A month mask indicating which months the schedule may be active
c) A day mask indicating which days of the week the schedule may be valid
d) A date mask indicating which dates of the month the schedule may be active
e) A link to a day plan record

To have a link to a day plan, the day plan is also required to be supported, which in turn requires that its container class and the day plan table are also required to be supported. The day plan table is characterized by:

a) The maximum number of day plans that may be stored, which must be between one and 255
b) The maximum number of events that may occur during a day, which must be between one and 255
c) An indication of the day plan that is currently active

The day plan itself only consists of the day plan number and a link to between one and 255 day plan events. Each day plan event is described by:

a) A number;
b) The hour during which the event occurs;
c) The minute during which the event occurs;
d) The status of the action; and
e) A link to the specific action to be performed

The specific action to be performed is defined elsewhere because of the device-specific nature of actions.

B.6 AUXILIARY INPUT/OUTPUT INFORMATION

Figure 14 depicts the auxiliary input/output data stored by a controller. Two diagrams are shown, one depicting the methods and object definitions defined in NTCIP 1201 v02 and the methods and objects defined in NTCIP 1201 v03.

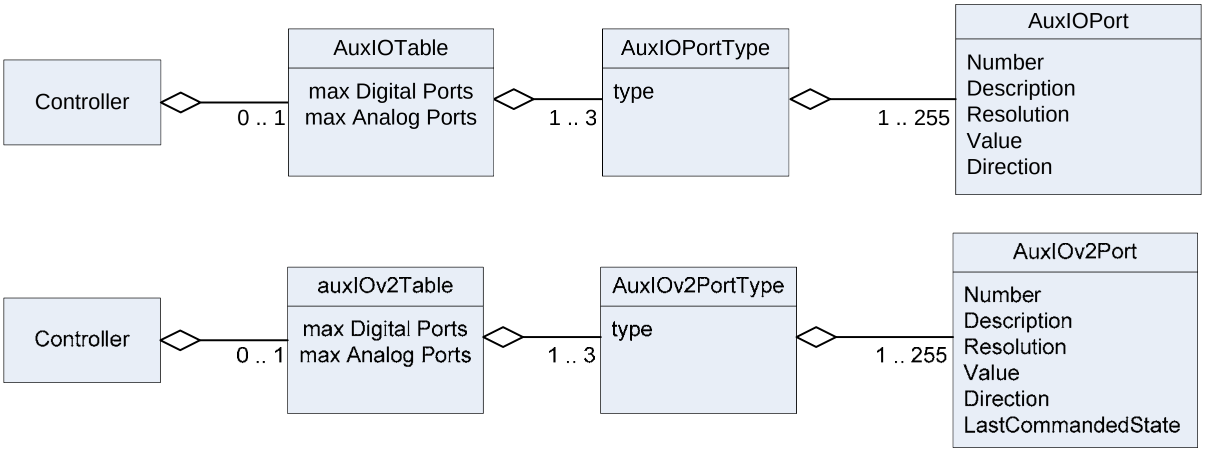


Figure 14 Class Diagrams for Auxiliary Input/Output Services (NTCIP 1201 v02 & NTCIP 1201 v03)

Figure 14 indicates a controller may support an auxiliary input/output table (AuxIO2 and/or AuxIO). If either is supported, it is characterized by the maximum number of digital and analog ports supported by the device. Each port type is allocated to its own sub-table in the AuxIOPortType table, which contains multiple entries, one for each port, where each port is characterized by:

a) A number
b) A description
c) A resolution of the data supported by the port

d) A value
e) A direction
f) The last commanded state (only in NTCIP 1201 v02)

Annex C
TEST PROCEDURES
[NORMATIVE]

NOTE—It is anticipated that a future version of NTCIP 1201 may include test procedures, and Annex C serves as a placeholder.

Annex D
SUMMARY OF CHANGES
[INFORMATIVE]

To the extent reasonable, the NTCIP community attempts to minimize the number of changes to an NTCIP standard to minimize interoperability issues among different versions of a single NTCIP standard. However, on occasion, issues are identified with existing NTCIP standards that necessitate a change. When rectifying such issues, NTCIP standards attempt to minimize the impact on existing implementations. Annex D explains the issue identified resulting in significant revisions, provides a description of the revision made, and a brief analysis of the impact of each revision on existing implementations.

NOTE—NTCIP 1201:2005 is referenced as NTCIP 1201 v02. NTCIP 1201:1996 (also referred to as NTCIP 1201:1997) is referenced as NTCIP 1201 v01.

D.1 REVISIONS FROM NTCIP 1201 V01 TO NTCIP 1201 V02

D.1.1 Updated Object Tree

Because of the various other revisions in NTCIP 1201 v02, the (ISO) Object Tree contained in Section 1.4 was updated to properly reflect content. Interoperability issues are not anticipated.

D.1.2 Updated to Conform with NTCIP 8004 v02

Data stored in field devices are often retrieved by a central system and then may be exchanged with other centers as a part of regional communications. These center-to-center communications use protocols other than SNMP and require the data to be defined according to either IEEE 1489 (or its recently approved update known as ISO 14817). NTCIP v01 standards did not define data in this format, creating ambiguities for center-to-center implementations.

To ensure that a single definition for all NTCIP data, regardless of its use context (e.g., center-to-center vs. center-to-field), NTCIP defined an enhanced MIB format, as defined in NTCIP 8004 v02, to be used for all new and updated NTCIP standards.

Interoperability issues are not anticipated because of the additions that this update creates (e.g., the <DEFINITION> tags, etc.).

D.1.3 Updated Name of the MIB

Changes to a MIB can affect the way other MIBs import data. Thus, when a MIB imports data from another MIB, it should be able to unambiguously reference the specific version of the MIB that it wants to import. Therefore, every update to an NTCIP standard results in an update to the name of the MIB according to the rules in NTCIP 8004 v02.

Additionally, because there are now two versions of the auxiliary input/output object definitions, two additional MIBs were created, each of which contains one of the AuxIO definitions.

The update to the MIB name should prevent ambiguity as to which version of a MIB may be referenced from another MIB.

D.1.4 Added Default Value Statements

Interoperability problems can arise when different controllers initialize differently. As a result, this standard has standardized the default initialization value of several configuration and control parameters.

This NTCIP 1201 v02 revision may result in some NTCIP 1201 v01 devices performing slightly differently than NTCIP 1201 v02 devices. However, this should *reduce* interoperability issues overall. Current implementations operate differently from one another, and any central system is customized to handle this uniqueness for each manufacturer. By defining the default value, this customization can be avoided in the future.

D.1.5 [Section Deleted]

NOTE—Annex D.5, renumbered as Annex D.1.5 in NTCIP 1201 v03, was not included in NTCIP 1201 v02.

D.1.6 Enhanced Module Version Definition

The module table provides basic information about the make, model, and version of the controller. However, NTCIP 1201 v01 provided a generic format for the version that did not adequately allow for proper configuration management of software. NTCIP 1201 v02 defined a detailed format for the presentation of version information.

While some NTCIP 1201 v01 devices may not conform, this revision should not present any real interoperability issues between NTCIP 1201 v01 and NTCIP 1201 v02 devices.

D.1.7 Added an Object to Identify Supported Standards

Several integrators have expressed concerns over the ability to be able to quickly determine to which NTCIP standards and which versions of NTCIP standards a device claims conformance. By being able to query the device to determine which standards it supports, a central system can quickly determine how to manage the device. Therefore, an object providing this information in a standard format was added.

Interoperability issues are not anticipated. A central system can readily identify any NTCIP v01 device, since the device returns a noSuchName error.

D.1.8 Corrected the Database Transaction Feature

The transaction mode process was modified by NTCIP 1201:1996 Amendment 1. Implementations discovered that the original process did not provide for the desired operation in the presence of multiple management stations (e.g., a central and a local laptop). Specifically, there were issues with the second management station overriding the first operation to issue a control command. The solution deprecated dbErrorID, dbTransactionID, and dbMakeID; revised the definition of dbCreateTransaction; and created two new objects labeled dbVerifyStatus and dbVerifyError.

This revision resolves pre-existing interoperability issues. While NTCIP 1201 v01 implementations require modification to conform to NTCIP 1201 v02, the NTCIP 1201 v01 feature did not work as intended.

NOTE—For the dbVerifyStatus object, to align NTCIP 1201 v02 with other NTCIP and several Internet standards, the object definitions in NTCIP 1201 v01 that had enumerated values starting with a value of (0) have been changed in NTCIP 1201 v02 to start with a value of (1). The changed definition in NTCIP 1201 v02 makes this object incompatible with NTCIP 1201 v01.

D.2 REVISIONS FROM NTCIP 1201 V02 TO NTCIP 1201 V03

D.2.1 Added Support for Additional Daylight Saving Modes

The *globalDaylightSaving* object was deprecated after NTCIP 1201 v02; therefore, the following paragraphs are no longer relevant. See Annex D.2.2 for more information.

Several parties located outside of the U.S. are now deploying NTCIP for various devices and have pointed out that the NTCIP should support all of the various daylight saving plans. Thus, these have been added to the daylight saving object.

This addition is fully backwards compatible and should not cause any interoperability problems. It will have no effect on systems in the US; NTCIP 1201 v01 systems outside of the US have not had a way to offer support of other daylight saving modes in a standard way, but with the NTCIP 1201 v02 enhancement, this feature is offered.

D.2.2 Added New Objects to Address US Daylight Saving Time (DST) Modifications

The globalDaylightSaving object was deprecated and a table allowing definition of the beginning and ending of DST was added (see dstBeginMonth). The definition of these parameters in a table allows definition of one or more sets of DST. Additionally, a new enumerated value was added to globalDaylightSaving object to support cases where a device supports both mechanisms and backward compatibility is required.

These changes were due to new rules enacted by the U. S. Congress to take effect in 2007. Since the DST rules have changed about every 10 years, the new set of DST objects allows users to define the start and end of DST.

D.2.3 Added a Schedule Status Object

Some agencies wanted to monitor the logic of the timebase schedule more closely and, as a result, a status object is added to the timebase schedule table. This extra feature is anticipated to be backward compatible.

D.2.4 Clarified Definitions of Day Plan Objects

Various questions were raised about the precise meaning of the object definitions for the day plan table. NTCIP 1201 v03 clarifies these definitions. However, the clarifications reflect actual implementations and interoperability issues are not anticipated.

D.2.5 Corrected Problems with the Local Time Logic

A problem was discovered with the time differential logic in that if the globalTime was set during the one-hour fall-back period of the DST logic; there was an ambiguity as to what time was intended. Manufacturers overcame this ambiguity via unique implementations, many of which created interoperability issues with other manufacturers. Several options were considered to correct this flawed logic, but all resulted in some level of interoperability issues. The best solution was to produce the cleanest design, which required deprecating the global time differential object (globalLocalTimeDifferential) and adding new objects for local time (controllerLocalTime) and time zone (controllerStandardTimeZone).

NTCIP 1201 v02 corrects an existing interoperability problem. Minor compatibility issues between the variety of NTCIP 1201 v01 interpretations and NTCIP 1201 v02 were noted, but an alternative solution that adequately corrected these issues without presenting new issues was not identified. By deprecating objects and creating new objects, any central system can quickly discover (by receiving a noSuchName error) when the wrong version is used.

D.2.6 Clarified Definitions Related to the Event Log

The exact definitions used for objects in the event log yielded a variety of detailed comments. Several clarifications resulted, but in all cases, these merely clarified the text and explained how manufacturers had implemented the features. Interoperability issues are not anticipated.

D.2.7 Reordered Sections for the Event Log

The order of the sections related to the event log proved confusing to some users, and sections were re-ordered. However, the OBJECT IDENTIFIERs for the objects have not changed. As such, this is an editorial change; interoperability issues are not anticipated.

After the above changes were made, the report node and event related objects were moved to NTCIP 1103 v02.

D.2.8 Added Support for Another Mode to Event Log and Moved to NTCIP 1103 v02

Based on requests from implementers, a new mode was added for the event log configuration table (andedWithValue) and better explanations of the definitions of each mode were provided.

This is an editorial clarification; interoperability issues are not anticipated.

NOTE—After the above changes were made, the report node and even-related objects were moved to NTCIP 1103 v02.

D.2.9 Added Error Value to the Event Configuration Status

Based on requests from implementers, an error code to the status object of the event configuration table was added to ensure that the controller is not programmed to repeatedly check an invalid condition. Logic was also added to the object that requires a consistency check whenever the configuration of the row changes.

Minor backward compatibility challenges may be anticipated; however, issues associated with some manufacturers’ use of the 'other' code to mean error are avoided.

After the above revisions were made, the report node and event related objects were moved to NTCIP 1103 v02.

D.2.10 Corrected Syntax of Event Log Size Object

NTCIP 1201 v01 indicated that the lower bound of the event log size was zero; however, if the size was zero, there would be no table and this object would not be supported. Thus, to avoid this contradiction, the lower limit was re-defined to be one.

Interoperability issues are not anticipated, since any implementation supporting this feature is expected to have a value greater than one.

After the above revisions were made, the report node and event related objects were moved to NTCIP 1103 v02.

D.2.11 Replaced the Group Address Object

NTCIP 1201 v01 had an object defining the PMPP group address to which the device belonged; however, the meaning of the value had been interpreted in two different ways. One group held that the value was supposed to be the group address number that was encoded in the PMPP address field. The other group held that the value was the encoded PMPP address field. Because of this conflict and resulting non-interoperability in deployed systems, the existing object (hdlcGroupAddress) was deprecated, and a new object was defined (hdlcGroupAddressNumber) to resolve the issue.

While replacing the existing object presents a minor interoperability issue, the solution does provide an unambiguous definition of the object, and any central system can readily identify NTCIP 1201 v01 implementations since these do not support the NTCIP 1201 v02 object.

D.2.12 Added Generic Auxiliary I/O Objects

The development of the NTCIP DMS standard (NTCIP 1203 v01) identified the need to support auxiliary I/O ports. This need was later identified by other NTCIP device-specific standards development groups, including the ESS WG. As a result, the auxiliary objects defined in NTCIP 1203 were moved to NTCIP 1201 v02. An effort to just refine, enhance, and rename the object definitions was anticipated to lead to interoperability issues. Therefore, the original object definitions moved from NTCIP 1203 v01 to NTCIP 1201 v01 were retained (see Section 2.10) but deprecated, and it is anticipated that these will be removed from a successor of NTCIP 1201 v03. The renamed and redefined object definitions are now also defined (see Section 2.9).

The provision of both sets of Auxiliary I/O object definitions, minimizes interoperability issues with either NTCIP 1201 v01 or NTCIP v02 implementations.

D.2.13 Removed Conformance Statements

Deployments using the NTCIP v01 (‘version 01’) standards highlighted problems with using conformance groups in agency specifications. Issues also arose as NTCIP standards were updated to include systems engineering information (such as user needs and requirements), since user needs and requirements for each feature were typically not defined in a clear fashion within the NTCIP v01 standard.

Subsequent NTCIP standards follow an outline that is based on a systems engineering process (SEP). As a result, conformance groups were eliminated and replaced with a Protocol Requirements List (PRL). In the case of NTCIP 1201 v03, the PRL is located in the subject device standard that references NTCIP 1201 v03. The PRL, combined with the Requirements Traceability Matrix (RTM), now defines the conformance requirements rather than the conformance groups and conformance statement used in NTCIP v01 standards.

Interoperability issues are not anticipated.

D.2.14 Added a Concept of Operations

The intended operation of some features was unclear to NTCiP 1201 v01 implementers. To address this issue, Annex A is included to explain how various features were intended to operate.

Although this is normative text, these are intended to clarify the text that already existed in the version 01 standard and are therefore not expected to produce any interoperability problems.

D.2.15 Prepared Communication Objects Moved to NTCIP 1103 v02

The security objects and the event log objects moved to NTCIP 1103 v02, because they relate more to application layer issues than to the end-application.

D.2.16 Deleted Annex B to Document Deprecated Objects

NTCIP 1201 v02 contained an Annex B that documented all objects that were deprecated between NTCIP 1201 v01 and NTCIP 1201 v02. Annex B provided future developers with an understanding of objects that may exist in or are used by legacy equipment. However, Annex D now serves this purpose.

Including this information may assist in newer systems being able to communicate with NTCIP 1201 v01 devices.

D.2.17 Added Class Diagrams

Many users of NTCIP 1201 v01 found it difficult to understand the context of the various objects defined in the Management Information Base (MIB). While various object definitions provided the detailed definition of each object, it was difficult to readily understand, at a high-level, how the data worked together.

Annex B provides high-level graphical images that depict the various relationships among all data defined in NTCIP 1201 v03, including the rules on multiplicity (i.e., how many of one object might exist for a given instance of another object). While all of this information was (and still is) recorded within the textual definition of the objects, providing high-level graphical depictions of these relationships facilitate this understanding.

Including this information may assist in clarification for users, without (of itself) contributing to backward compatibility issues.

D.2.18 Added Generic SNMP Interface Definitions

NTCIP 1203 v02 contained the definitions for a generic Simple Network Management Protocol (SNMP) interface. This definition continues to appear in NTCIP 1201 v03 Annex E because these operations are generic to all field devices.

Annex E
SNMP INTERFACE
[NORMATIVE]

The device shall conform to the requirements for the Simple Network Management Protocol (SNMP) as defined in NTCIP 1103 v02. Annexes E.1 through E.4 provide a description of the key services offered by SNMP assuming no errors; precise rules and procedures are defined in NTCIP 1103 v02. Annex E.5 extends the requirements of NTCIP 1103 v02 by providing additional requirements that supplement, but do not replace any requirements of NTCIP 1103 v02.

To promote interoperability and to reflect marketplace realities, each NTCIP device-specific standard identifies which of the NTCIP 1103 v02-defined protocols (Simple Network Management Protocol [SNMP], Simple Transportation Management Protocol [STMP] and the Simple Fixed Message Protocol [SFMP]) the device is required to support. For example, the NTCIP dynamic message sign (DMS) standard (NTCIP 1203 v01) requires support only for SNMP, and support of STMP and SFMP is in fact discouraged because "these have not been widely implemented in DMS and thus would likely result in decreased interoperability, limited competition, and increased costs for testing, integration, and maintenance."

E.1 GENERIC SNMP GET INTERFACE

SNMP defines a generic process by which a management station can retrieve data from a device. This process consists of a Get request (GET) and a GetResponse as depicted in Figure 15. Both the Get request and the GetResponse messages contain a list of objects as defined by the varBindingList structure (see Annex E.4). The GetResponse varBindingList shall consist of the same objects identifications, as found in the Get varBindingList, plus their object values.

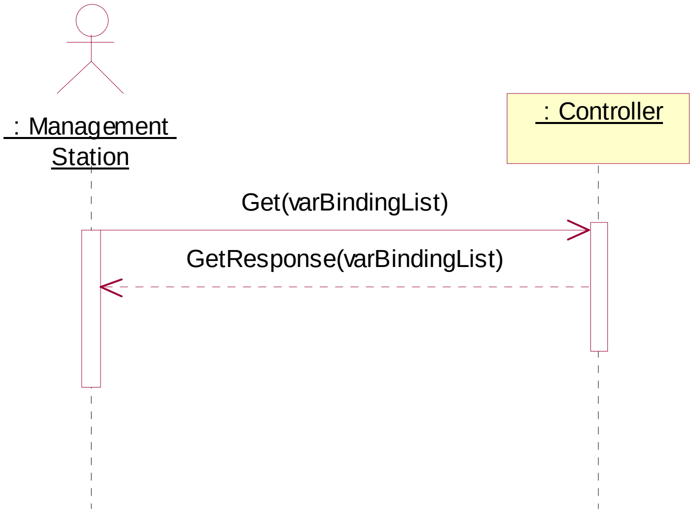


Figure 15 SNMP Get Interface

E.2 GENERIC SNMP GET-NEXT INTERFACE

SNMP defines a process by which a management station can explore data within a device. This process consists of a GetNext request and a GetResponse as depicted in Figure 16. Both the GetNext request and the GetResponse messages contain a list of objects as defined by the varBindingList structure (see

Annex E.4). The GetResponse varBindingList shall consist of the same objects identifications, as found in the Set varBindingList, plus their object values.

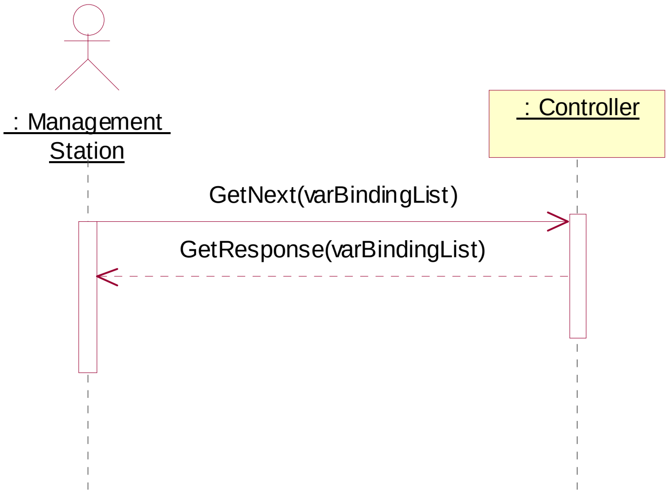


Figure 16 SNMP GetNext Interface

E.3 GENERIC SNMP SET INTERFACE

SNMP defines a generic process by which a management station can send data to a device. This process consists of a Set request and a GetResponse as depicted in Figure 17. Both the Set request and the GetResponse messages contain a list of objects as defined by the varBindingList structure (see Annex E.4).

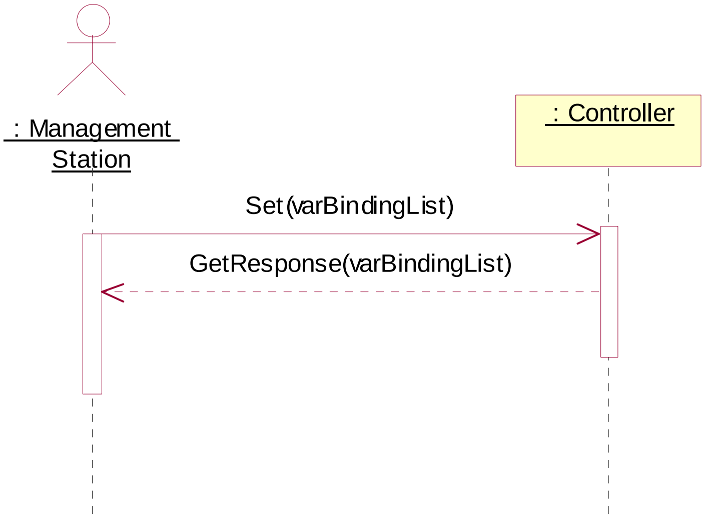


Figure 17 SNMP Set Interface

NOTE—The response message issued to an SNMP Set request is the same message structure as used to respond to an SNMP Get request. The SNMP standard calls this response message a GetResponse, but it is in fact a response to either a GET or a SET.

E.4 VARIABLE BINDING LIST STRUCTURE

The requests and responses for the Get, Get Next and Set operations all use the varBindingList structure. NTCIP 1103 v02 defines this structure as containing zero or more varBindings, where each varBinding is defined to consist of an object name (as indicated by an Object Identifier (OID)) and the associated object value. This relationship is depicted in Figure 18.

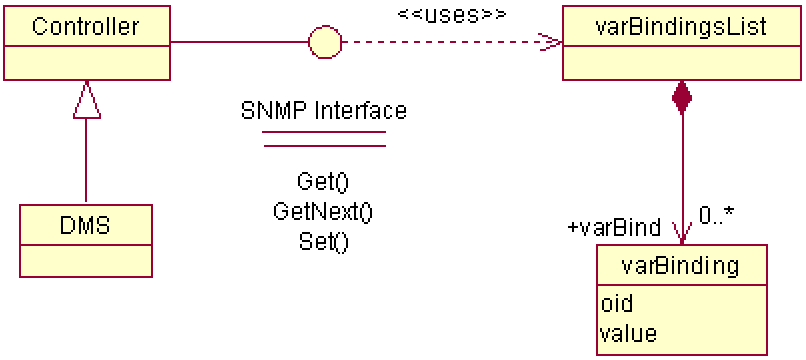


Figure 18 SNMP Interface—View of Participating Classes

E.5 ADDITIONAL REQUIREMENTS

E.5.1 Grouping of Objects in a Request

The device shall allow the management station to perform a single Get, GetNext, or Set operation on any combination of supported objects with the objects listed in any order within the message, unless otherwise restricted by this standard.

The device shall not associate any semantics to the ordering of objects within the varBindingsList. As required by RFC 1157, Section 4.1.5, each object shall be affected "as if simultaneously set with respect to all other assignments specified in the same message."

E.5.2 Support of Get

The device shall allow the management station to perform the Get operation on any supported object for which support for the Get Operation is indicated in Annex E.4.

E.5.3 Support of GetNext

The device shall allow the management station to perform the GetNext operation on any OBJECT IDENTIFIER.

E.5.4 Support of Set

The device shall allow the management station to perform the Set operation on any supported object for which support for the Set Operation is indicated in Annex E.4.

E.5.5 Performance

The device shall process the Get, GetNext, or Set request in accordance with all of the rules of NTCIP 1103 v02 Section 3.2.4 (Response Time), including updating the value in the database and initiating the transmission of the appropriate response.

NOTE—If a user desires a shorter response time, the user specifies this in the agency specification.

§