A Joint Standard of AASHTO, ITE, and NEMA

NTCIP 1211 version v02

National Transportation Communications for ITS Protocol

Object Definitions for Signal Control and Prioritization (SCP)

Published in September 2014

This Adobe® PDF copy of an NTCIP standard is available at no-cost for a limited time through support from the U.S. DOT ITS Joint Program Office, 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 900
Rosslyn, Virginia 22209-3801

file version 1211 v02.24j

NOTICES

Copyright Notice

PDF File License Agreement

Data Dictionary and MIB Distribution 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.

PRL and RTM Distribution Permission

To the extent that these materials are distributed by AASHTO / ITE / NEMA in the form of a Profile Requirements List ("PRL") or a Requirements Traceability Matrix ("RTM"), AASHTO / ITE / NEMA extend the following permission:

a) you may make or distribute unlimited copies, including derivative works of the PRL (then known as a Profile Implementation Conformance Statement ("PICS")) or the RTM, provided that each copy you make or distribute contains the citation "Based on NTCIP 0000 [insert the standard number] PRL or RTM. Used by permission. Original text © AASHTO / ITE / NEMA."
b) you may only modify the PRL or the RTM by adding: 1) text in the Project Requirements column, which is the only column that may be modified to show a product's implementation or the project-specific requirements; and/or 2) additional table columns or table rows that are clearly labeled as ADDITIONAL for project-unique or vendor-unique features; and
c) if the PRL or RTM excerpt is made from an unapproved draft, add to the citation "PRL (or RTM) excerpted from a draft standard containing preliminary information that is subject to change."

This limited permission does not include reuse in works offered by other standards developing organizations or publishers, and does not include reuse in works-for-hire, compendiums, or electronic storage devices that are not associated with procurement documents, or commercial hardware, or commercial software products intended for field installation.

A PICS is a Profile Requirements List that is completed to indicate the features that are supported in an implementation. Visit www.ntcip.org for information on electronic copies of the MIBs, PRLs, and RTMs.

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

NTCIP is a trademark of AASHTO / ITE / NEMA. All other marks mentioned in this standard are the trademarks of their respective owners.

FOREWORD

NTCIP 1211 v02, this standard, defines the management information base for Signal Control and Prioritization (SCP) Systems. It defines individual parameters that represent the configuration, status, and control information that is unique to an SCP. NTCIP 1211 v02 defines a set of objects for use in controlling traffic signal systems in priority applications.

NTCIP 1211 v02 provides definitions of the management information related to Signal Control and Prioritization (SCP). SCP management information includes individual parameters that represent the configuration, status, and control information of such a device. It also includes data frames and message sets of these parameters and others from different standards to address the operational information exchanges between the devices in a baseline system configuration.

NTCIP 1211 v02 defines requirements that are applicable to an NTCIP environment that involves the control of traffic signal controllers. While the term Signal Control and Prioritization implies a logical implementation, the data concepts may be applicable to a physical device. By definition, SCP operates in the context of a "system" that includes a priority requester, such as a transit vehicle, traffic signal controllers, and the management centers that configure and monitor the traffic signal controllers. It, therefore, imposes functional and communications requirements on all of these "system" components. The following keywords apply to NTCIP 1211 v02: NTCIP, traffic signal control, transit, emergency responder, priority control, prioritization, signal priority.

It also defines specific groupings of these parameters and others to address the operational configuration, monitoring, and control of the device/entity in a baseline system configuration. NTCIP 1211 v02 is an NTCIP Device Data Dictionary Standard. NTCIP Device Data Dictionaries Standards formally express management information in terms of objects (data elements, data frames, and messages) for use within NTCIP systems.

NTCIP 1211 v02 uses only metric units. For more information about NTCIP standards, visit the NTCIP Web site at http://www.ntcip.org.

There are normative and informative annexes in NTCIP 1211 v02:

a) Annex A is normative and contains a Requirements Traceability Matrix (RTM) that traces requirements to the dialogs and data objects used to fulfill it.
b) Annex B is informative and contains the object tree showing the major nodes of the SCP object structure within the Global object tree.
c) Annex C is normative and (in a future version) is intended to contain the test procedures associated with the user needs, functional requirements, dialogs, and objects defined in NTCIP 1211 v02.
d) Annex D is informative and contains each of the major revisions and changes provided for between NTCIP 1211 v02, and its predecessor, NTCIP 1211 v01.
e) Annex E is informative and documents user requests that were not included in NTCIP 1211 v02.
f) Annex F is informative and provides a tutorial on SCP and provides examples on how a typical SCP system may work.
g) Annex G is normative and contains the definitions for the Generic SNMP interface including the definitions to perform GET, SET, GET NEXT commands. These definitions may to be moved to a different NTCIP standard at a future date, because this content is applicable to all device-specific NTCIP standards.
h) Annex H is normative and serves as a placeholder for systems engineering descriptions that may be moved to a different NTCIP standard at a future date, because this content is applicable to all device-specific NTCIP standards.

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 NTCIP 1211 v02 content. A “Request for Interpretation” of NTCIP 1211 v02 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 are referred to the committee responsible for NTCIP 1211 v02. 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 is forwarded to the submitter. The committee chairperson may report that action on the User Comment may be deferred to a future committee meeting, 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:

NTCIP Coordinator
National Electrical Manufacturers Association
1300 North 17th Street, Suite 900
Rosslyn, Virginia 22209-3801
e-mail: ntcip@nema.org

A User Comment should be submitted as follows:

Standard 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 1211 v02 was separately balloted and approved by AASHTO, ITE, and NEMA after recommendation by the Joint Committee on the NTCIP. Each organization has approved NTCIP 1211 v02 as the following standard type, as of the date:

AASHTO – Standard Specification; August 2014
ITE – Software Standard; September 2014
NEMA – Standard; September 2014

History

In 1992, the NEMA 3-TS Transportation Management Systems and Associated Control Devices Section began development of the NTCIP. The Transportation Section’s purpose was in response to user needs to include standardized systems communication in NEMA TS 2, *Traffic Controller Assemblies*. 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 Intelligent Transportation Systems (ITS) network.

In September 1996, an agreement was executed among AASHTO, ITE, and NEMA to jointly develop, approve, and maintain the NTCIP standards. The Joint Committee on the NTCIP formed a working group to develop a common set of management information related to controlling the roadside devices that act in supervisory capacity. The SCP WG's first meeting was in November 1999.

The NTCIP 1211 v02 Project was initiated June 2008.

NTCIP 1211 v02.09 May 2014: Issued User Comment Draft with responses due May 12, 2014.

NTCIP 1211 v02.11 June 2014: NTCIP SCP WG accepts draft NTCIP 1211 v02.21, and forwards to NTCIP Joint Committee for consideration in June 2014.

NTCIP 1211 v02.23 June 2014: NTCIP Joint Committee accepts and issues to AASHTO, ITE and NEMA for Ballotting.

NTCIP 1211 v02.24 September 2014: Prepared standard for publication.

Compatibility of Versions

To distinguish NTCIP 1211 v02 (as published) from previous drafts, NTCIP 1211 v02 also includes NTCIP 1211 v02.24 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 1211 v02 is designated, and should be cited as, NTCIP 1211 v02. Anyone using NTCIP 1211 v02 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.

CONTENTS

FIGURES

TABLES

< This page intentionally left blank. >

Section 1
General [Informative]

1.1 Scope

The scope of NTCIP 1211 v02 is the definition of management information and the operations related to Signal Control and Prioritization (SCP). It covers the management of preferential treatment (priority) of different classes of vehicles (such as transit, emergency service, other priority vehicles, which might include commercial fleet vehicles, etc.) and the implementation of special coordination operation within a Traffic Signal Controller. To affect a priority request, its scope includes the content of messages to be exchanged between a prioritizing entity and a controller, the sequence in which messages are exchanged, and associated functions within a controller related to SCP. The scope includes the configuration and monitoring of a prioritizing entity and those aspects of coordination that relate to SCP. The process of generating priority requests is not defined by NTCIP 1211 v02.

NTCIP 1211 v02 deals, primarily, with granting priority while still maintaining coordination with adjacent intersections. The functionality expressed here is intended to work in conjunction with the coordination object definitions and functions as defined in NTCIP 1202. The coordination objects and functions can also be invoked when an intersection is operating in a non-coordinated mode.

Granting absolute priority irrespective of coordination is considered pre-emption and is covered in NTCIP 1202:2005. The process of generating a priority request is also not covered here. For transit vehicles, this process is defined in the American Public Transportation Association's (APTA) Transit Communications Interface Profiles (TCIP) standard.

The remainder of NTCIP 1211 v02 provides the following main sections, each building on the previous section(s):

a) Concept of Operations – This section provides a description of user needs (needs for features and needs related to the operational environment) applicable to SCP.
b) Functional Requirements – This section defines the functional requirements that address the user needs identified in the Concept of Operations. It includes a Protocol Requirements List (PRL) Table that defines conformance requirements thereby allowing users to select the desired options for a particular project.
c) Dialogs – This section describes how functional requirements that require more complex implementations are fulfilled. The dialogs define the standardized procedure for a management station to manage a device and define the responses expected by the device.
d) Management Information Base (MIB) – This section defines the data exchanged during communications (an update of NTCIP 1211 v01 Sections 3 and 4)
e) Requirements Traceability Matrix – This annex provides a table that associates each requirement to a dialog, an interface, and its associated list of objects.
f) Test Procedures – This annex is a placeholder for future test procedures, which ensure that a requirement is met and is validated.

The first two of these sections are presented at a high level and areof interest to most readers of NTCIP 1211 v02; the later sections entail more detailed design issues that are of interest to implementers, integrators, and testers.

Additional annexes provide information on certain topics such as changes between NTCIP 1211 v01 and NTCIP 1211 v02, and the Generic SNMP Interface description.

1.2 References

1.2.1 Normative References

The following standards contain provisions, which, through references in this text, constitute provisions of NTCIP 1211 v02. By reference herein, these standards are adopted, in whole or in part as indicated, in this publication. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on NTCIP 1211 v02 are encouraged to investigate the possibility of applying the most recent editions of the standards listed.

Identifier	Title
AASHTO / ITE / NEMA NTCIP 1103 v02	*Transportation Management Protocols (TMP)* published July 2010
AASHTO / ITE / NEMA NTCIP 1201 v03	*Global Object (GO) Definitions* published March 2011
AASHTO / ITE / NEMA NTCIP 1202:2005	*Object Definitions for Actuated Traffic Signal Controller (ASC) Units – version 02* published November 2005
AASHTO / ITE / NEMA NTCIP 2301 v02	*Simple Transportation Management Framework (STMF) Application Profile (AP) (AP-STMF)* published July 2010
AASHTO / ITE / NEMA NTCIP 8004 v02	*Structure and Identification of Management Information (SMI)* published June 2010
RFC1213	*Management Information Base for Network Management of TCP/IP-based Internets* Published March 1991

1.2.2 Other References

The following documents and standards may provide the reader with a more complete understanding of the entire protocol and the relations between all parts of the protocol. However, these documents do not contain direct provisions that are required by NTCIP 1211 v02. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on NTCIP 1211 v02 are encouraged to investigate the possibility of applying the most recent editions of the standard listed.

Identifier	Title
U.S. National ITS Architecture, Version 7.0	National ITS Architecture, FHWA
AASHTO / ITE / NEMA NTCIP 8007 v01	Testing and Conformity Assessment Documentation within NTCIP Standards Publications
APTA-TCIP-S-01 4.0	Transit Communications Interface Profiles (TCIP)
OMG Unified Modeling Language Specification, Version 1.5	OMG Unified Modeling Language Specification, Object Management Group, 2003
ISO 14817:2002	Transport information and control systems – Requirements for an ITS/TICS central Data Registry and ITS/TICS Data Dictionaries
IEEE Std. 610.12-1990	IEEE Standard Glossary of Software Engineering Terminology

1.2.3 Contact Information

1.2.3.1 IEEE Standards

IEEE Standards may be obtained from IEEE publications as follows:

Institute of Electrical and Electronics Engineers, Inc.
445 Hoes Lane, Piscataway, NJ 08854
standards.ieee.org/

1.2.3.2 ISO/IEC Standards

Members of the International Organization for Standardization (ISO) maintain registers of currently valid ISO/IEC (International Electrotechnical Commission) International Standards. For the USA, the member of ISO is ANSI, which is listed above.

1.2.3.3 National ITS Architecture

The U.S. National ITS Architecture may be viewed on-line at http://itsarch.iteris.com/itsarch/.

1.2.3.4 NTCIP Standards

Copies of NTCIP standards may be obtained from:

NTCIP Coordinator
National Electrical Manufacturers Association
1300 N.17th Street, Suite 900
Rosslyn, Virginia 22209-3806
www.ntcip.org
e-mail: ntcip@nema.org

Note: Visit www.ntcip.org for information on availability of electronic copies of the PRL and RTM.

1.2.3.5 Internet Architecture Board (IAB) Documents

Electronic copies of IAB (RFC) documents may be obtained from:

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

1.3 General Statements

<In the opinion of the responsible NTCIP working group, Section 1.3 does not apply in the context of NTCIP 1211 v02.>

1.4 Terms

For the purposes of NTCIP 1211 v02, the following terms and definitions apply. For terms not defined here, English words are used in accordance with their definitions in the latest edition of Webster's New Collegiate Dictionary. Electrical and electronic terms not defined here or in Webster's New Collegiate Dictionary are used in accordance with their definitions in IEEE Std 100-2000.

Term	Definition
Coordination	The control (operation) of (traffic signal) controller units in a manner to provide a relationship between specific green indications at adjacent intersections in accordance with a time schedule to permit continuous operation (movement) of groups of vehicles along the street at a planned speed. Note: The parenthetical expressions are added to the definition as it appears in NTCIP 1202:2005.

Term	Definition
Coordinator	A logical device or program/routine that provides coordination. An integral part of a Traffic Signal Controller.
Data Element	Per NTCIP 8004 v02 and as defined in ISO 14817, a data element is some single unit of information of interest (such as a fact, proposition, observation, etc.) about some (entity) class of interest (e.g., a person, place, process, property, concept, association, state, event). A data element is considered indivisible in a particular context. Note: A data element is represented by an object class, a property of the represented object class and a value domain. In the context of NTCIP 1211 v02, the SNMP Object Type Macro is used to define the entity type, property, and an explicit value domain term.
Data Frame	Per NTCIP 8004 v02 and as defined in ISO 14817, a data frame is a grouping of data elements primarily for the purpose of referring to the group with a single name, and thereby efficiently reusing groups of data elements that commonly appear together (as an ASN.1 SEQUENCE, SEQUENCE OF, SET, SET OF or CHOICE) in a message specification. Note: This data concept type may be used to specify groups of data elements for other purposes as well. In the context of SNMP, a data frame consists of only the information (variable bindings) to be exchanged. It does not include the application layer header.
Fleet Management	The supervisory functions related to planning, operation, control, and maintenance of fleet vehicles such as buses.
Interchangeable	A condition which 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: From National Telecommunications and Information Administration, U.S. Department of Commerce.
Interoperable	The ability of two or more systems or components to exchange information and use the information that has been exchanged Note: From IEEE Std. 610.12-1990: IEEE Standard Glossary of Software Engineering Terminology.
Management Information Base (MIB)	A structured collection or database of related managed objects defined using Abstract Syntax Notation One (ASN.1). Note: A Management Information Base (MIB) may refer to all managed objects within an implementation or a subset that are specifically to some type of functionality.
Management Station	Defined as a computing platform that manages NTCIP field components, such as a PRS or a CO. A management station may be a traffic management center or a maintenance laptop that a field technician may use on a trip to visit the component.

Term	Definition
Message	Per NTCIP 8004 v02 and as defined in ISO 14817, a message is a grouping of data elements and/or data frames, as well as associated message metadata, that is used to convey a complete unit of information. Note: For the purposes of NTCIP 1211 v02, a message is an abstract description; it is not a specific instance. In the context of SNMP, a message consists of the application layer header and the information (variable binding) to be exchanged.
Preemption	Per NTCIP 1202:2005, the transfer of the normal control (operation) of traffic signals to a special signal control mode for the purpose of servicing railroad crossings, emergency vehicle passage, mass transit vehicle passage, and other special tasks, the control of which requires terminating normal traffic control to provide the service needs of the special task.
Priority	The preferential treatment of one vehicle class (such as a transit vehicle, emergency service vehicle or a commercial fleet vehicle) over another vehicle class at a signalized intersection without causing the traffic signal controllers to drop from coordinated operations. Note: Priority may be accomplished by a number of methods including changing the beginning and end times of greens on identified phases, changing the phase sequence, or inclusion of special phases, without interrupting the general timing relationship between specific green indications at adjacent intersections.
Priority Request	The information that describes a need for priority service based upon user-defined criteria (such as the number of minutes behind schedule, vehicle occupancy levels, vehicle class, etc.). Note: A priority request is sent from a Priority Request Generator to a Priority Request Server.
Priority Request Generator	A logical or physical entity that initiates a priority request.
Priority Request Server	A logical or physical entity that manages and prioritizes one or more priority requests and generates one or more service requests.
Reservice Period	A period of time to limit the frequency that priority requests are serviced.
Service Request	The information that describes a priority service to be processed by the Coordinator within a Traffic Signal Controller. Note: A service request is sent between a Priority Request Server and a Traffic Signal Controller.
Traffic Management	The supervisory functions related to planning, operation, control, and maintenance of traffic control devices.
Traffic Control Assembly	The collection of components that reside in a traffic signal control cabinet. Note: In addition to a traffic signal controller, a typical assembly includes vehicle detection and other input/output devices.

1.5 Abbreviations

The acronyms used in NTCIP 1211 v02 are defined as follows:

APTA American Public Transportation Association
ASN.1 Abstract Syntax Notation One
CO Coordinator
GNSS Global Navigation Satellite System

GLO	Global (Object Definitions) (NTCIP 1201 v03)
ITS	Intelligent Transportation Systems
MIB	Management Information Base
OER	Octet Encoding Rules
PICS	Protocol (or Profile) Implementation Conformance Specification
PRG	Priority Request Generator
PRL	Protocol Requirements List
PRS	Priority Request Server
RFC	Request for Comments
RTM	Requirements Traceability Matrix
SCP	Signal Control and Prioritization
SCP WG	Signal Control and Prioritization Working Group
SNMP	Simple Network Management Protocol
TCIP	Transit Communications Interface Profiles
TED	Time of Estimated Departure
TMC	Traffic or Transit Management Center
TSD	Time of Service Desired

NTCIP 1211 v02 references object definitions that appear in other standards. In those cases, the object name is preceded by the entity acronym followed by a period. For example, the controller-localTime object that is defined in NTCIP 1201 v03 would be referred to as GLO.globalTime.

Section 2
Concept of Operations [Normative]

Section 2 defines user needs that NTCIP 1211 v02 addresses in subsequent sections. Accepted system engineering processes detail that requirements should only be developed to fulfill well-defined user needs. The first stage in this process is to identify the ways in which the system is likely to be used. For NTCIP 1211 v02, this entails identifying the various ways in which transportation operations personnel may use Signal Control and Prioritization (SCP) information to fulfill their duties.

This concept of operations provides:

a) A detailed description of the scope of NTCIP 1211 v02;
b) An explanation of how an SCP system is expected to fit into the larger context of an ITS network;
c) A starting point in the agency specification and procurement process; and
d) An understanding of the perspective of the designers of NTCIP 1211 v02.

This section is intended for all users, including:

a) Transportation operations managers
b) Transportation operations personnel
c) Transportation engineers
d) System integrators
e) Device manufacturers

This section is intended to assist the first three categories of users in understanding how SCP devices can be incorporated in an existing system. For this audience, Section 2 serves as the starting point in the agency specification and procurement process. Users can become familiar with each feature addressed in NTCIP 1211 v02 and determine whether or not a feature is appropriate for their agency-specific implementation. If a feature is appropriate, then an agency's procurement specification should be structured to require support for the feature and all of the mandatory requirements related to that feature.

The last two categories of users can gain a more thorough understanding as to why more detailed requirements exist later in NTCIP 1211 v02.

2.1 Tutorial [Informative]

A concept of operations describes a proposed system from the users' perspective. Typically, a concept of operations is used on a project to ensure that the system developers understand the users' needs. Within NTCIP standards, the concept of operations documents the purpose of each feature for which an NTCIP standard supports a communications interface. The concept of operations also serves as the starting point for users to select those features that may be appropriate for a specific project.

The concept of operations starts with a discussion of the current situation and issues that have led to the need to deploy systems within the scope of a standard and to the development of the standard itself. This discussion permits both potential users and system developers to understand the situation.

The concept of operations then documents key aspects of the proposed system, including:

a) Reference Physical Architecture—The reference physical architecture defines the overall context of the proposed system and defines which specific interfaces are addressed. The reference physical architecture is supplemented with several example physical configurations that describe how the reference physical architecture may be realized in an actual implementation.
b) Architectural Needs—The architectural needs discuss issues and needs relative to the system architecture.
c) Features—The features identify and describe the various functions that users may want components of an SCP system to perform. These features are derived from the high level user needs identified in

the problem statement but are refined and organized into a more manageable structure that forms the basis of the traceability tables contained in Section 3 and Annex A.

Architectural needs and features are collectively called user needs. In Section 3, these user needs define the various functional requirements of an SCP system. Basic systems engineering requires that:

a) each user need traces to one or more functional requirement(s), and
b) each functional requirement derives from at least one user need.

This traceability is shown in the Protocol Requirements List (PRL) Table in Section 3.3.3.

NTCIP 1211 v02 is intended for use in a broad range of prospective implementations. Within the PRL, each user need and requirement is identified as mandatory, optional, or conditional, and users may complete the PRL to clearly define unique aspects of their implementation. Within NTCIP 1211 v02, items marked mandatory are those that relate to the most basic functionality of SCP devices. For specific implementations, the user identifies those optional or conditional needs appropriate for a specific implementation.

Each requirement is presented in the Requirements Traceability Matrix (RTM) in Annex A, which defines how the requirement is fulfilled through the standardized dialogs and data element definitions provided in Sections 4, 5 and 6.

A conformant device may support other user needs, as long as they are conformant with the requirements of NTCIP 1211 v02 and the standards it references (e.g., NTCIP 2301 v02 and NTCIP 1201 v03). For example, a device may support data that has not been defined by NTCIP 1211 v02; however, when exchanged via one of the NTCIP 2301 protocols, the data shall be properly registered with a valid OBJECT IDENTIFIER under the Global ISO Naming Tree.

> Note: Off-the-shelf interoperability and interchangeability can only be obtained by using well documented user needs, along with their corresponding requirements and design, that are broadly supported by the industry as a whole. Users should be aware that designing a system that uses environments or features not defined in a standard or not typically deployed in combination with one another may inhibit the goals of interoperability and interchangeability, especially if the documentation of these user needs is not available for distribution to system integrators. NTCIP 1211 v02 allows implementations to support additional user needs and to support innovation, which is constantly needed within the industry; but users should be aware of the risks involved with using such environments or features.

The concept of operations concludes by describing the degree to which NTCIP 1211 v02 addresses security issues, and by providing a description of how NTCIP 1211 v02 relates to the National ITS Architecture.

2.2 Current Situation and Problem Statement [Informative]

Transportation system managers use SCP in a variety of ways to improve their transportation system operations. The primary uses of SCP data support the following:

a) To provide for vehicles in need of preferential treatment at signalized intersections;
b) To improve the on-time performance of public transportation without degrading the overall performance of the traffic network; and
c) To provide more efficient use of the street network by improving the throughput of travelers and goods.

Signal priority is a process or control strategy to facilitate the movement of fleet vehicles through signalized intersections. SCP is a means of executing the signal priority control strategy. Without SCP, traffic signal controllers at signalized intersections are programmed to treat each vehicle equally, without regard to the type or need of the vehicles at the intersection. With SCP, signal priority can be provided to certain classes of vehicles, if needed.

Signal priority is defined as the preferential treatment of one vehicle class (such as a transit vehicle, emergency service vehicle, or a commercial fleet vehicle) over other vehicle classes at a signalized intersection without causing the traffic signal controllers to drop from coordinated operations. Signal priority may be accomplished by a number of methods including increasing or decreasing the green times on specific phases, changing the phase sequence, omitting a phase, or the inclusion of special phases, without interrupting the general timing relationship between specific green indications at adjacent intersections. The most common vehicle class for which signal priority is given is transit vehicles.

The key to an effective SCP is to facilitate desired fleet vehicle movement at a signalized intersection while minimizing its negative impacts on other vehicles and the traffic network. An effective SCP has little negative impact (and in fact, may have a positive impact) on general traffic. SCP is a cost-effective method to enhance regional mobility through fleet vehicles that transport more people and goods in comparison to single-occupant or smaller vehicles.

SCP is also an inexpensive way to make transit more attractive to travelers. Using SCP can improve schedule adherence, improve transit vehicle efficiency, make transit service more reliable and improve travel times, with minimal negative impacts to normal traffic operations.

The impact of SCP on other vehicles and on the traffic network is what differentiates signal priority from signal pre-emption. Signal priority is a “request for service” that modifies the signal operation in an orderly fashion to provide preferential treatment to vehicles requesting signal priority. Signal pre-emption is a “demand for service” that interrupts the signal operation to immediately service the demand. Signal pre-emption often temporarily disrupts any coordination the signal may have with other traffic signals adjacent to it. Signal pre-emption is usually used with an approaching train at a signal-rail intersection or with emergency vehicles, such as a fire engine responding to an incident.

2.3 Reference Physical Architecture [Informative]

An SCP can be configured in several ways and generally requires integration of elements from two or more organizations. For example, a fleet organization, such as a transit agency, needs to send its request for preferential treatment to the traffic agency that owns and operates the traffic signal controllers. The specific elements between which the communications take place vary depending on how the SCP system is implemented. The communications may be between the fleet management center and the traffic management center, between the fleet vehicle or the traffic signal controller, between a fleet roadside device and the traffic signal controller, or any other combination of the entities mentioned.

The reference physical architecture defines the overall context of an SCP system, including the:

a) Components of an SCP;
b) Typical physical architectures of an SCP; and
c) Specific interfaces addressed by NTCIP 1211 v02.

2.3.1 Components of an SCP

Fundamentally, an SCP consists of three components: a Priority Request Generator (PRG), a Priority Request Server (PRS), and the Coordinator (CO). There are also three secondary components that may make up an SCP system: the Fleet Management Center, the Fleet Vehicle, and the Traffic Management Center.

The following paragraphs describe the primary functions of these components.

2.3.1.1 Priority Request Generator (PRG)

The PRG is a logical entity that can be physically located in various locations. The PRG may be located at a Fleet Management Center, on a Fleet Vehicle, at a Traffic Management Center, within a roadside cabinet, or within the traffic signal controller.

The primary functions of the PRG are as follows:

a) To produce an estimate of the time of service desired for a fleet vehicle approaching the signalized intersection. This estimate is intended to represent the vehicle's arrival time at the intersection and can range from zero (0) (representing a request for immediate service) to some time in the future.
b) To produce an estimate of the time of departure for a vehicle from the intersection's stopping point. This estimate is intended to represent the vehicle's departure time from the intersection and can range from zero (0) (representing a vehicle at the intersection) to some time in the future.
c) To send a request for signal priority, which consists of the time of service desired, the time of estimated departure, and the priority strategy desired to the PRS.
d) To send and receive the status of a priority request from the PRS.

Standards and specifications relating to the design and functions of the PRG are outside the scope of NTCIP 1211 v02.

> Note: The SCP WG considered the specification of strategy in developing NTCIP 1211 V02 and concluded that the PRG may not know the specific strategy or even the phase that would serve the desired movement, but should know the entering approach, and possibly the exit approach, directions that specify the desired movement of the requesting vehicle at the intersection. The SCP WG is considering proposals for addressing this need in NTCIP 1211 v03 while maintaining backward compatibility with NTCIP 1211 v01.

Multiple Fleet Vehicles from different Fleet operations can all produce requests for signal priority.

2.3.1.2 Priority Request Server (PRS)

In most implementations, the PRS is a logical entity that is located in the traffic signal cabinet and can be implemented as either a separate component installed in the traffic signal cabinet or incorporated internally in the traffic signal controller logic. A PRS can also be located in a Traffic Management Center.

The primary functions of the PRS are as follows:

a) To receive priority requests from different PRGs.
b) To send the status of the priority requests received back to the originating PRG.
c) To prioritize the different priority requests received based on the requests' vehicle class level and time of service desired.
d) To exchange a service request with the CO that has the time of service desired, the time of estimated departure and the priority strategy desired for each priority request.
e) To exchange the status of the service requests with the CO.
f) Optionally, to produce a log of all the priority requests received; and the service requests exchanged with the CO.
g) Optionally, to send the contents of the log to the Traffic Management Center.

2.3.1.3 Coordinator (CO)

The CO is a logical entity that is an integral part of a traffic signal controller. The CO has the primary responsibility for processing the service requests received from the PRS. Each service request consists of the requested time of service, the estimated time of departure, and the priority strategy requested for each priority request received. Also, requests for priority service can be sent to a CO, but a CO cannot be commanded to service a priority request by the SCP system.

The primary functions of the CO are as follows:

a) To receive a service request from a PRS.
b) To send the status of a service request received back to the PRS.
c) To implement the priority strategy requested in the service request received from the PRS.
d) Optionally, to produce a log of all the service requests received from the PRS and the priority strategies implemented.
e) Optionally, to send the contents of the log to the traffic management center upon request.

2.3.1.4 Fleet Management Center

A Fleet Management Center's role and responsibility vary depending on how the SCP system is implemented. For some SCP implementations, the Fleet Management Center's responsibility includes receiving vehicle location from Fleet Vehicles, then sending that information to a Traffic Management Center. In other SCP implementations, the Fleet Management Center's responsibility is to receive priority requests from the PRG, then forwarding those priority requests to a Traffic Management Center. There are other SCP implementations where the Fleet Management Center has no functional role in the data exchanges to request and implement signal priority.

2.3.1.5 Fleet Vehicles

The Fleet Vehicles have the primary responsibility for sending their vehicle location either to a PRG or the Fleet Management Center, depending on how the SCP system is implemented.

2.3.1.6 Traffic Management Center

The Traffic Management Center has the primary responsibility for installing, configuring, operating, and maintaining a PRS and a CO. The PRS and CO should be capable of being configured remotely by the Traffic Management Center via a management station. For some SCP implementations, the Traffic Management Center is also responsible for receiving priority requests from Fleet Management Centers and forwarding those priority requests to the PRS.

2.3.1.7 Management Station

A management station is defined as a computing platform that manages NTCIP field components, such as a PRS or a CO. A management station may be a traffic management center or a maintenance laptop that a field technician may use on a trip to visit the component.

2.3.2 Typical Physical Architectures of an SCP

There are many different possible physical configurations associated with an SCP system. This section illustrates only some of the possible physical configurations (or architectures) of an SCP. The differences between the physical architectures are:

a) the physical location of the PRG;
b) if the PRS is a physical or a logical entity, and the location of the PRS; and
c) the intermediary entities between the PRG and the PRS, if any.

Note: The physical architecture illustrations that follow assume that the PRS is implemented as a physical entity. It is possible that a traffic signal controller design integrates all the functions of a PRS as a logical entity within the controller, or that a traffic management center integrates all the functions of a PRS as a logical entity within the traffic management software. It is also possible that the traffic signal controller design integrates all the functions of a PRG as a logical entity within the controller.

2.3.2.1 Physical Architecture – Example 1

The first example physical architecture (Figure 1) represents a case in which the PRG resides in the Fleet Vehicle but there is no direct communications path between the Fleet Vehicle and the Traffic Signal Controller.

In this architecture, the PRG sends a "priority request" but because there is no direct communications path between the Fleet Vehicle and the Traffic Signal Controller, the priority request is routed through the Fleet Management Center (see Interface #6 in Figure 1) and the Traffic Management Center (#5). This example logically treats the Fleet Vehicle, the Fleet Management Center, and the Traffic Management Center as the PRG because the Fleet Management Center and the Traffic Management Center only act as intermediaries to forward the priority request and the status of the priority request between the PRG on the Fleet Vehicle and the PRS.

Thus, for this example, the PRG consists of the Fleet Vehicle, the Fleet Management Center, and the Traffic Management Center. The (“logical”) PRG sends the priority request to the PRS (#1). The PRS can physically reside, either partially or in total, in either the Traffic Management Center or on the roadside, but is a logical entity that receives multiple "priority requests" and sends the status of each “priority request” back to the (“logical”) PRG. Within the (“logical”) PRG, the Traffic Management Center then forwards the status of the priority request back to Fleet Management Center (#5), which in turn may forward the status of the priority request back to the Fleet Vehicle (#6).

NTCIP 1211 v02 does not cover the communications interfaces between the Fleet Vehicle and Fleet Management Center (#6), or between the Fleet Management Center and Traffic Management Center (#5) because NTCIP 1211 v02 is a center-to-field communications standard. Other standards, such as TCIP, a transit communications standard, or TMDD, a center-to-center standard, should be considered for the Fleet Vehicle to Fleet Management Center communications interface and for the Fleet Management Center to Traffic Management Center communications interface, respectively. However, the information exchanges between these entities should include the information necessary for the Fleet Vehicle, Fleet Management Center and the Traffic Management Center to forward and receive the priority request and the status of a priority request.

Besides the (“logical”) PRG and the PRS interface for this physical architecture, NTCIP 1211 v02 addresses the communications interfaces between Traffic Management Center and the PRS (#1); between the Traffic Management Center and the CO (#2); and between the PRS and the CO (#4). The PRS processes the priority requests received from the PRG and sends "service requests" to the CO (#4), the logical entity in the Traffic Signal Controller that processes the service request. The CO then responds to the PRS with the status of the service request.

Both the PRS and the CO also may log events, which are then retrieved by the Traffic Management Center (#1 and #2, respectively). The Traffic Management Center is also responsible for configuring the PRS and the CO.

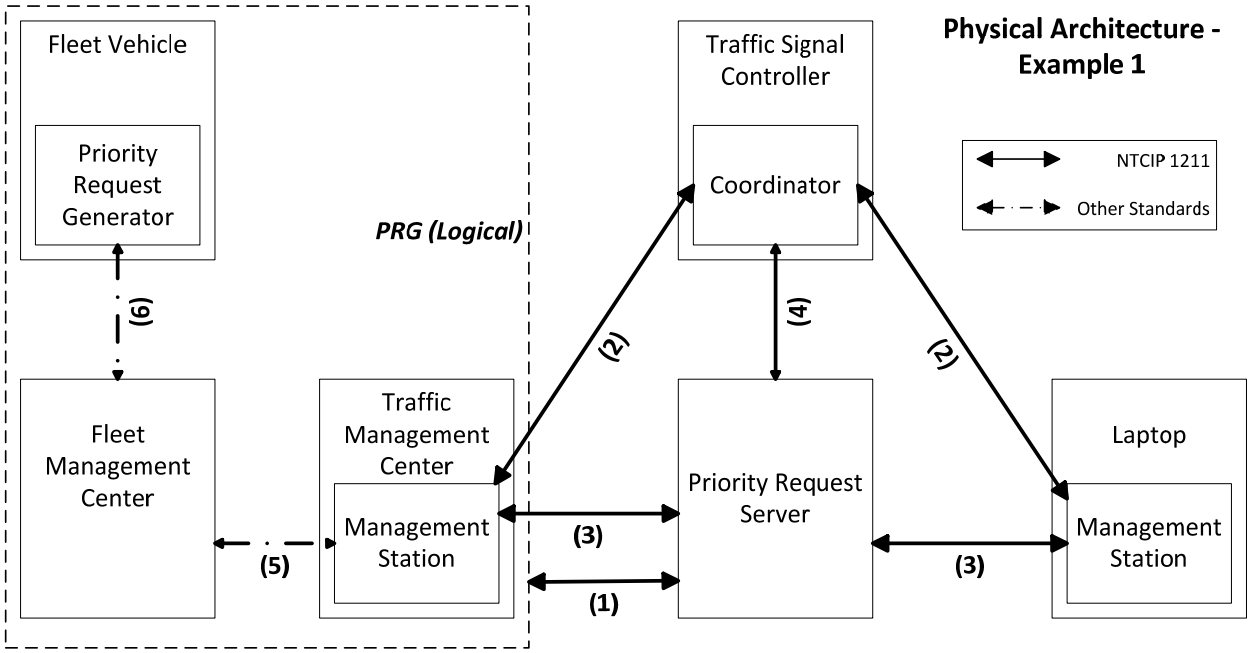

Figure 1 Physical Architecture Example 1

2.3.2.2 Physical Architecture Example 2

Physical architecture example 2 is a slight variation of physical architecture example 1 in that the PRG is physically located in the Fleet Management Center. As shown in Figure 2, the PRG is located in the Fleet Management Center, but there is no direct communication between the Fleet Management Center and the Traffic Signal Controller.

Thus for this physical architecture, the PRG sends a priority request, but the priority request is routed through the Traffic Management Center (see Interface #5 in Figure 2) because there is no direct communications path between the Fleet Management Center and the PRS. This example logically treats the Fleet Management Center and the Traffic Management Center as the PRG, because the Traffic Management Center only acts as an intermediary to forward the priority request and the status of the priority request between the PRG in the Fleet Management Center and the PRS.

Thus, for this example, the PRG consists of the Fleet Management Center and the Traffic Management Center. The (“logical”) PRG sends the priority request to the PRS (#1). The PRS can physically reside, either partially or in total, in either the Traffic Management Center or on the roadside, but is a logical entity that receives multiple "priority requests" and sends the status of each “priority request” back to the (“logical”) PRG. Within the (“logical”) PRG, the Traffic Management Center then forwards the status of the priority request back to Fleet Management Center (#5).

NTCIP 1211 v02 does not cover the communications interface between the Fleet Management Center and the Traffic Management Center (#5) because NTCIP 1211 v02 is a center-to-field communications standard. Other standards, such as TCIP, a transit communications standard, or TMDD, a center-to-center standard, should be considered for the Fleet Management Center to Traffic Management Center communications interface. However, the information exchanges between these entities should include the information necessary for the Traffic Management Center to forward the priority request and the status of a priority request.

Besides the (“logical”) PRG and the PRS interface for this physical architecture, NTCIP 1211 v02 addresses the communications interfaces between Traffic Management Center and the PRS (#1); between the Traffic Management Center and the CO (#2); and between the PRS and the CO (#4). The

processes and data exchanges between these three interfaces are the same as those in physical architecture example #1.

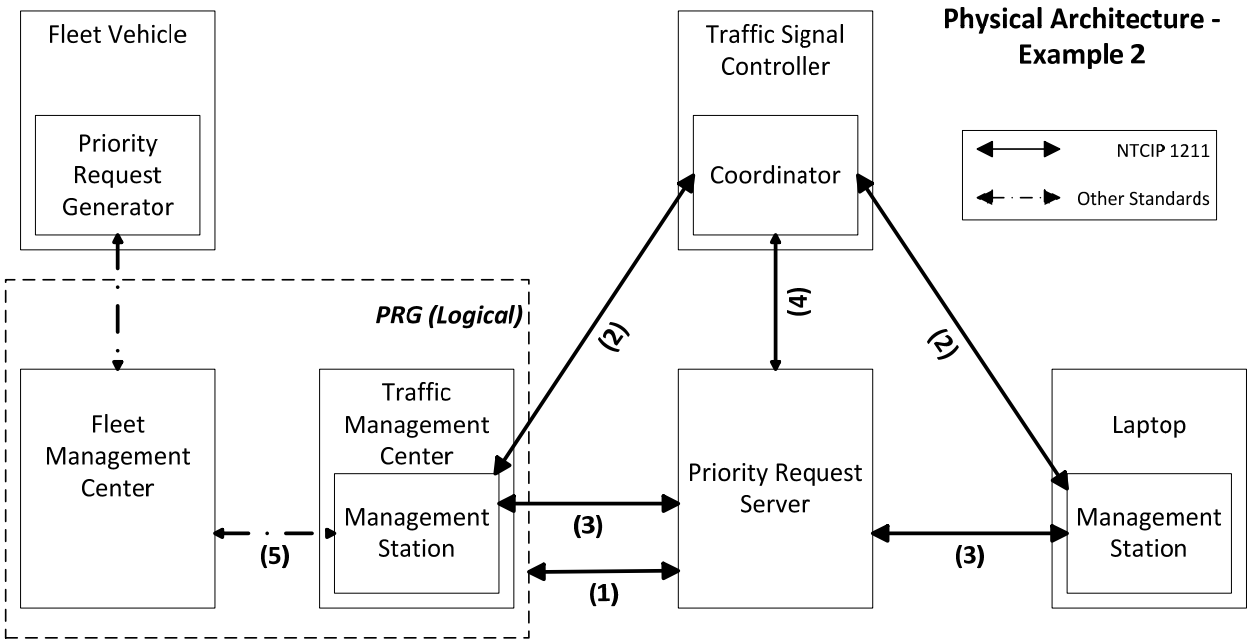

Figure 2 Physical Architecture Example 2

2.3.2.3 Physical Architecture Example 3

Physical architecture example 3 is another variation of the first two physical architectures in that the PRG resides either as a logical or physical entity in the Traffic Management Center (see Figure 3). The PRG initiates a priority request and sends the priority request to the PRS, which can reside in part or in whole in the Traffic Management Center or on the roadside (see Interface #3 in Figure 3). The PRS may receive multiple "priority requests" and returns the status of each “priority request” back to the PRG.

The PRS also processes the priority requests to send "service requests" to the CO, the logical entity in the Traffic Signal Controller that processes the service request (#4). The CO then responds to the PRS with the status of the service request. Both the PRS and the CO may log events, which are retrieved by the Traffic Management Center (#1 and #2, respectively). The Traffic Management Center is also responsible for configuring the PRS and the CO.

For this physical architecture, NTCIP 1211 v02 addresses the communications interfaces between Traffic Management Center and the PRS (#1); between the Traffic Management Center and the CO (#2); between the PRG and the PRS (#3); and between the PRG and the PRS (#4).

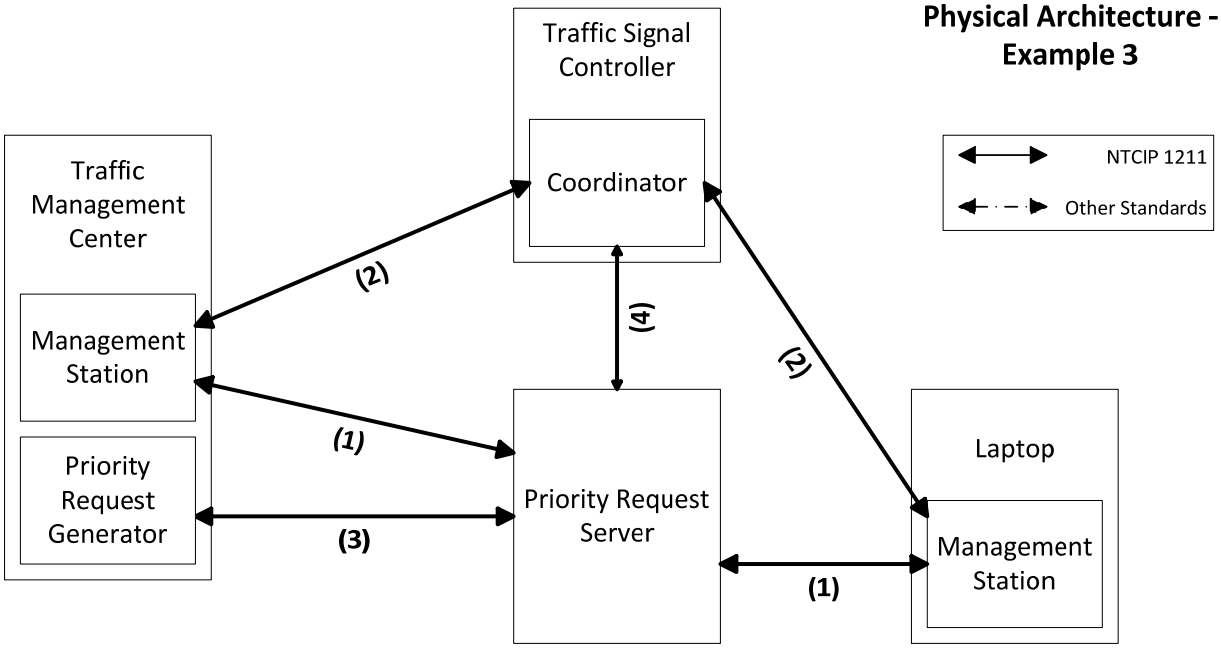

Figure 3 Physical Architecture Example 3

2.3.2.4 Physical Architecture Example 4

The fourth example of a typical SCP physical architecture is one in which there is a direct communications link between the Fleet Vehicle and the traffic signal cabinet (see Figure 4). As in physical architecture example 1, the PRG resides within a Fleet Vehicle and sends a priority request. However, in this example physical architecture, the PRG sends the priority request directly to a PRS, which is a logical or physical entity located in the traffic signal cabinet (see Interface #3 in Figure 4). The PRS receives the priority requests, processes them, and resolves them to a service request that is sent to the CO for processing (#4). The CO then responds back to the PRS with the status of the service request. The PRS may also return the status of each priority request back to the PRG.

Both the PRS and the CO may log events, which are retrieved by the Traffic Management Center (#1 and #2, respectively). The Traffic Management Center is also responsible for configuring the PRS and the CO.

For this physical architecture, NTCIP 1211 v02 covers the communications interfaces between Traffic Management Center and the PRS (#1); between the Traffic Management Center and the CO(#2); between the PRG and the PRS (#3); and between the PRS and the CO (#4).

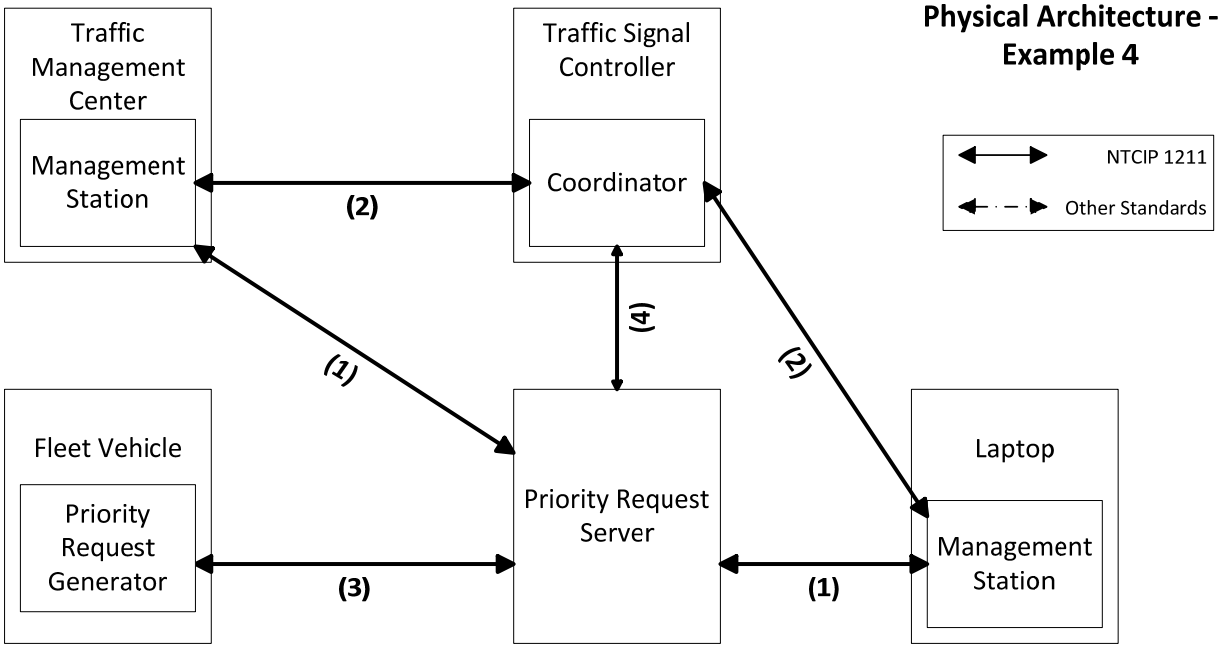

Figure 4 Physical Architecture Example 4

2.3.2.5 Physical Architecture Example 5

Physical architecture example 5 is one where the PRG is located in a field cabinet, and there is a direct communications link between the PRG and the traffic signal cabinet (see Figure 5). There is also a direct communications path between the PRG and either the Fleet Vehicle or the Fleet Management Center. The PRG sends a priority request directly to a PRS, which is a logical or physical entity located in the traffic signal cabinet (See Interface #3 in Figure 5). The PRS receives the priority service requests, processes them, and resolves them to one or more priority service requests that are sent to the CO for processing (#4). The CO then responds back to the PRS with the status of the service request. The PRS may also send the status of each priority request back to the PRG. Both the PRS and the CO may log events, which are retrieved by the Traffic Management Center (#1 and #2, respectively). The Traffic Management Center is also responsible for configuring the PRS and the CO.

For this physical architecture, NTCIP 1211 v02 addresses the communications interfaces between Traffic Management Center and the PRS (#1); between the Traffic Management Center and the CO (#2); between the PRG and the PRS (#3); and between the PRS and the CO (#4).

NTCIP 1211 v02 does not cover the communications interface between the Fleet Management Center and the PRG (#5) or between the Fleet Vehicle and the PRG (#6). Other standards, such as TCIP, a transit communications standard, should be considered for these communications interfaces. However, the information exchanges between these entities should include the information necessary for the PRG to send a priority request.

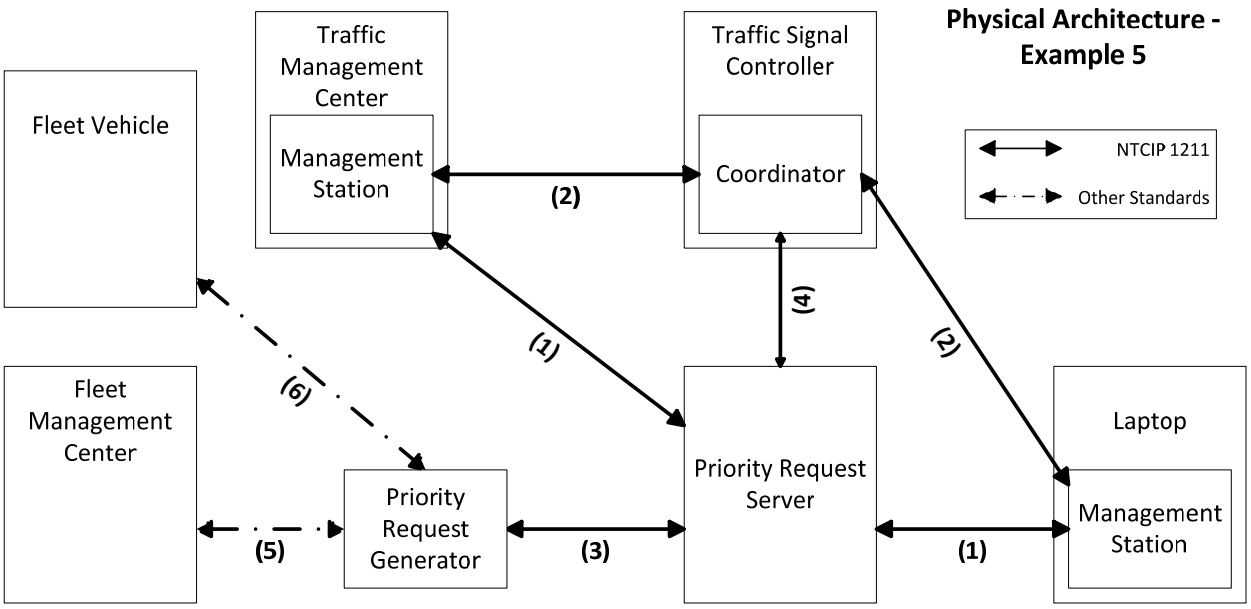

Figure 5 Physical Architecture Example 5

2.3.2.6 Physical Architecture Example 6

A sixth example of a typical SCP physical architecture is one in which there is a direct communications link between the Fleet Vehicle and the traffic signal cabinet and the functions of the PRG, PRS, and CO are all performed within the traffic signal cabinet, i.e., the logical entities PRG, PRS, and CO are all physically located within the traffic signal cabinet (see Figure 6).

This example physical architecture is probably the most common implementation of an SCP system, where an infrared transmitter on a fleet vehicle sends an encoded pulse to a receiver located near the traffic signal controller (see Interface #4 in Figure 6). The receiver then sends a signal to the traffic signal controller through a port, generally a contact closure, to indicate the request for a preferential treatment (#3). Depending on the implementation of the SCP system, there may be multiple ports (or contact closures) between the receiver and the traffic signal controller, one port for each vehicle class or direction for preferential treatment.

Within the traffic signal cabinet, the PRG reads the signal from the transmitter through the appropriate port and selects a priority strategy. The PRS prioritizes the priority requests, if there is more than one, and the CO then implements the priority strategy selected. Both the PRS and the CO may log events, which are retrieved by the Traffic Management Center (#1 and #2, respectively). The Traffic Management Center is also responsible for configuring the PRS and the CO.

Only two communications interfaces exist for this physical architecture that are addressed by NTCIP 1211 v02, between the Traffic Management Center and the PRS (#1), and between the Traffic Management Center and the CO (#2). Since the functions of the PRG, PRS, and the CO are physically integral within the traffic signal cabinet, the “interfaces” between these entities are implementation-specific and not addressed by NTCIP 1211 v02.

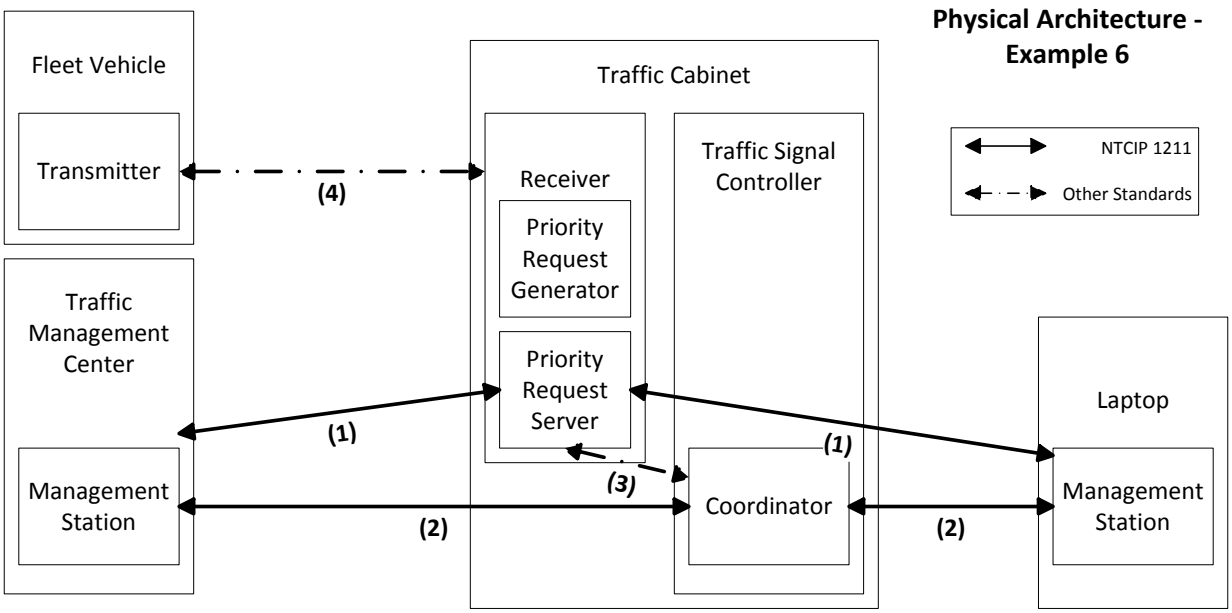

Figure 6 Physical Architecture Example 6

2.3.3 Interfaces

NTCIP 1211 v02 addresses four interfaces of an SCP system:

a) between a management station and a PRS;
b) between a management station and a CO;
c) between a PRG and a PRS; and
d) between a PRS and a CO.

A management station is defined as a computing platform that manages NTCIP field components, such as a PRS or a CO. A management station may reside anywhere, for example, in a traffic management center or as a maintenance laptop that a field technician may use on a trip to visit the component.

Each interface is discussed subsequently.

2.3.3.1 Interface—Management Station and PRS

This interface allows a management station to configure and monitor a PRS. The interface also allows a management station to retrieve event log information from the PRS. The PRS is a logical entity that may be physically located at a traffic management center, a field cabinet, or as an integral part of the traffic signal controller itself. Figure 7 illustrates the logical processes and information flows that pass through this interface.

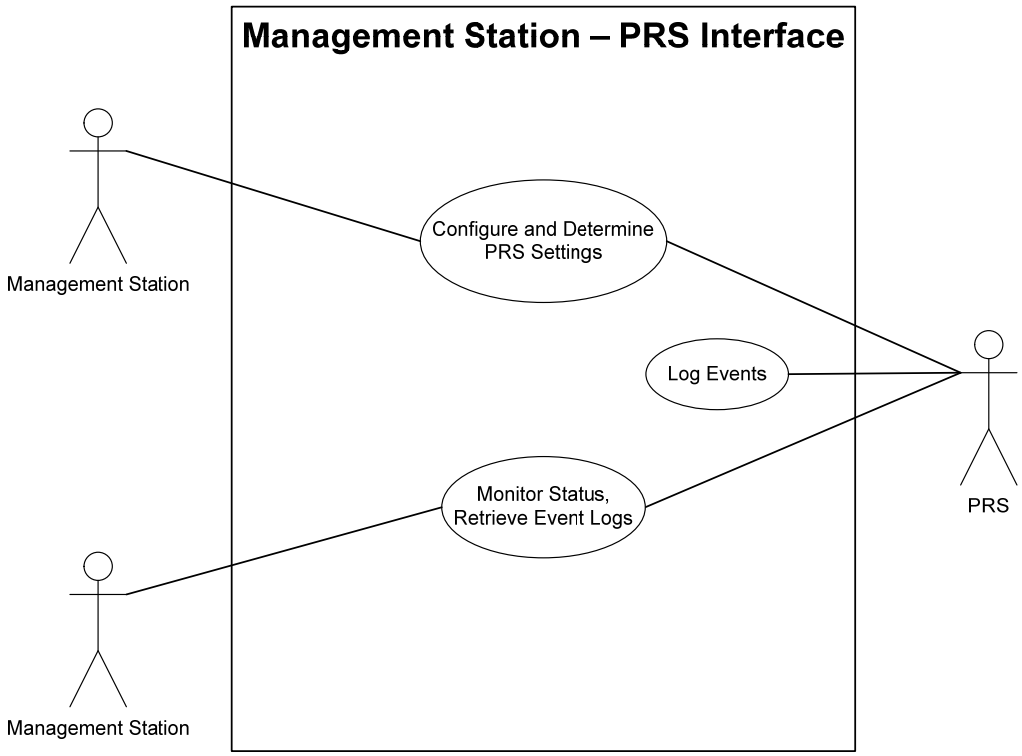

Figure 7 Management Station—PRS Interface

2.3.3.2 Interface—Management Station and CO

This interface allows a management station to configure and monitor a CO. The interface also allows a management station to retrieve event log information from the CO. The CO is a logical entity within the traffic signal controller. Figure 8 illustrates the logical processes and information flows that pass through this interface.

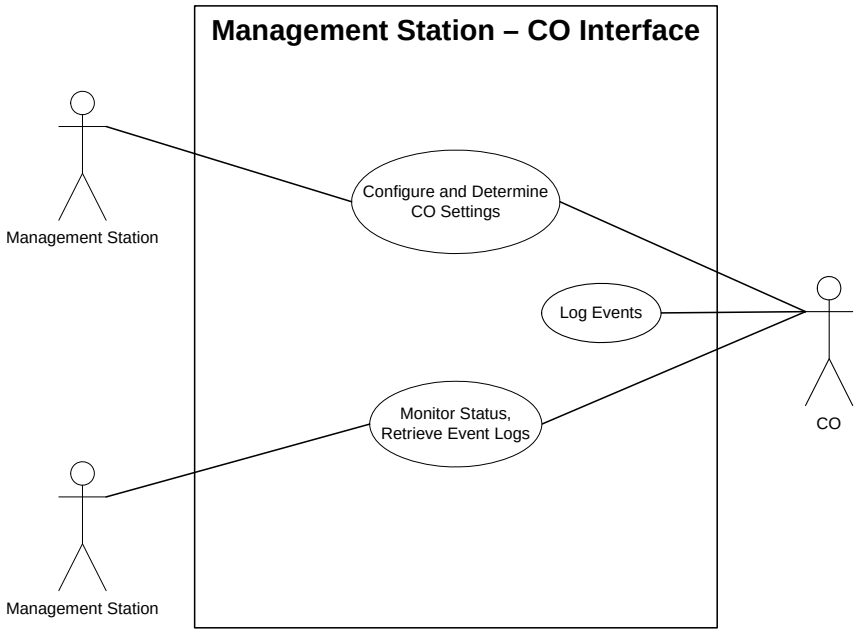

Figure 8 Management Station—CO Interface

2.3.3.3 Interface—PRG and PRS

This interface allows a PRG to send a request for preferential treatment to the PRS. The PRS may also simultaneously receive requests for preferential treatment from other PRGs and has to prioritize these competing requests. The PRS also sends the status of the priority request back to the PRG through the interface upon request. Figure 9 illustrates the logical processes and information flows that pass through this interface.

The physical location of the PRG varies based on the implementation of the SCP. The PRG may be located on a fleet vehicle, at a fleet management center, at a traffic management center or on the roadside in a field cabinet. If there are intermediary entities between the physical PRG and the PRS, as in Physical Architecture Examples 1 and 2, those entities are considered logically to be part of the PRG by NTCIP 1211 v02.

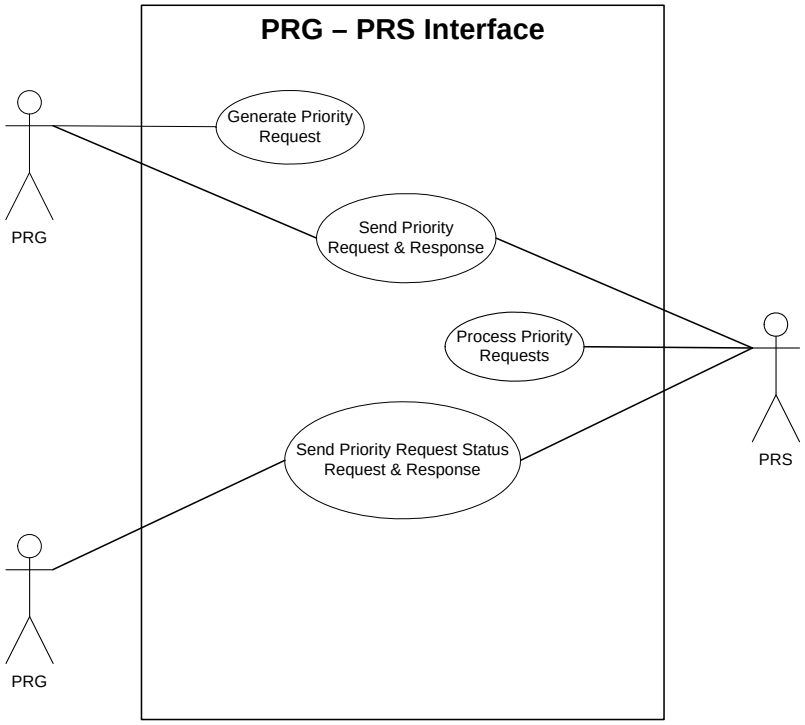

Figure 9 PRG—PRS Interface

2.3.3.4 Interface—PRS and CO

This interface allows the PRS to exchange a service request with the CO. Upon completing prioritization the priority requests it has received, the PRS exchanges this information with the CO. The CO uses the information to implement the priority strategies requested. Upon completion of its service processing, the CO exchanges the status of the priority requests with the PRS. Figure 10 illustrates the logical processes and information flows that pass through this interface.

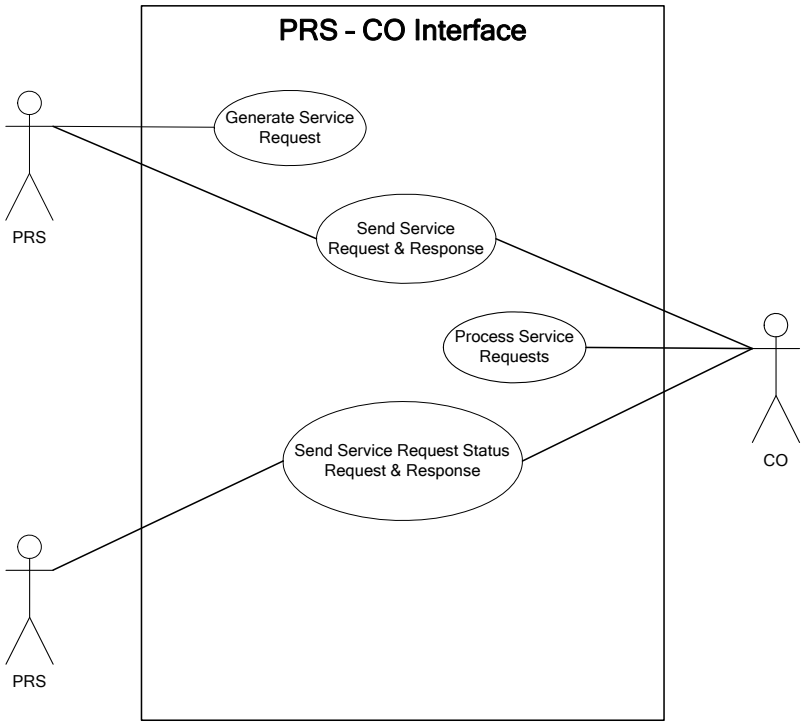

Figure 10 PRS—CO Interface

2.4 Architectural Needs

This section defines the communications environment within which an SCP system is expected to operate.

The operational environment may vary for each of the four interfaces in an SCP system that are addressed by NTCIP 1211 v02, depending on the physical architecture used to implement the SCP system. For some interfaces, such as the PRS and CO interface in physical architecture example 6, both entities may be logical entities that are integral to the same physical device, thus no “physical” communications exist between these entities. For other interfaces, an entity is required to be able to support communications from multiple instances of another entity, such as the PRS that is required to support communications from multiple PRGs, each of which is located on a different fleet vehicle, depicted in physical architecture example 4.

For each of the four interfaces supported by NTCIP 1211 v02, the cost of communications may be minimal and as such an interface may be designed for constant polling; other interfaces may encounter significant costs and as such the interface may be designed to minimize data exchanges.

When deploying an SCP system, the system designer is encouraged to consider which of the following operational environments need to be supported for each of the four interfaces supported by NTCIP 1211 v02.

2.4.1 Integral Entities

For operational environments where two logical entities are integral to the same physical device, the “communications” and interface between these two entities are implementation-specific and are not addressed by NTCIP 1211 v02.

2.4.2 Provide Live Data

This operational environment allows one entity to monitor and control another entity by issuing requests (e.g., requests to access information, alter information, or control the component). In this environment, the entity receiving the requests responds to the requests immediately (e.g., through the provision of live data, success/failure notice of information alteration, or success/failure of the command).

2.4.3 Support Multiple Instances of an Entity

Some operational environments require that one entity support communications with multiple instances of another entity. In that environment, the single entity receiving requests from multiple entities need to respond to each request. For example, a PRS may be required to support receiving and processing priority requests from multiple PRGs.

2.4.4 Provide Compressed Data

Some operational environments have limited data capacity due to limitations in the data rates of the media and/or due to multiple entities or devices sharing the same communications channel. In such environments, compressed data provides the capability for grouping sets of data together so that data can be transmitted more efficiently over telecommunications networks, thereby conserving the limited data capacity of the channel.

For an SCP system, data capacity may be a concern between a management station and the PRS; and a management station and the CO because a management station may configure and control multiple PRSs and COs on a communications channel.

The following subsections identify and describe the specific operational environment for the interface with a management station. These include:

a) Provide Compressed Data between a Management Station and a PRS
b) Provide Compressed Data between a Management Station and a CO

2.4.4.1 Provide Compressed Data between a Management Station and a PRS

This operational environment allows a management station to group sets of data together to a PRS so that data can be transmitted more efficiently over a telecommunications network.

2.4.4.2 Provide Compressed Data between a Management Station and a CO

This operational environment allows a management station to group sets of data together to a CO so that data can be transmitted more efficiently over a telecommunications network.

2.5 Features

The following subsections identify and describe the various user needs (features) that may be offered by an SCP system. It is organized by the four interfaces covered by NTCIP 1211 v02:

a) Interface—Management Station to PRS
b) Interface—Management Station to CO
c) Interface—PRG to PRS
d) Interface—PRS to CO

2.5.1 Interface—Management Station to PRS

The following subsections identify and describe the various features that may be offered between a management station and a PRS in an SCP system. These features include:

a) Manage the PRS
b) Determine Priority Request Criteria
c) Monitor the PRS
d) Retrieve Log Data from the PRS

2.5.1.1 Manage the PRS

The following subsections identify and describe the various features a management station needs to configure the PRS.

2.5.1.1.1 Determine PRS Identity

A management station may need to determine basic information about the PRS, such as the type, technology, manufacturer, model, and version number of the PRS. It includes the ability to access information about both hardware and software elements of the PRS.

2.5.1.1.2 Determine PRS Configuration

A management station may need to determine the version of the configuration data of the PRS. This feature allows an operator to determine if the configuration of a PRS has changed.

2.5.1.1.3 Configure Reservice Period

A management station needs to define the reservice period between when servicing one priority request is completed and when a subsequent priority request is serviced. This feature allows an operator to prevent the PRS from constantly servicing priority requests, therefore disrupting traffic flow. This feature also helps maintain headways to prevent bunching of transit vehicles.

2.5.1.1.4 Configure Time To Live Period

A management station needs to define the maximum period of time that a PRS considers a priority request. This feature allows an operator to manage the performance of the traffic network by not servicing priority requests that are sent by a PRG too early.

2.5.1.1.5 PRS Clock Synchronization

A management station needs to synchronize the clock on the PRS. This feature allows an operator to check that the PRS is synchronized with the traffic signal controller and the event log timestamps are synchronized with the management station.

2.5.1.2 Determine Priority Request Criteria

A management station needs to determine the criteria that a PRS uses to manage priority requests. The feature allows an operator to determine how a PRS is currently programmed to handle priority requests.

2.5.1.3 Monitor the PRS

A management station needs to monitor the status and time of service of the service requests from the PRS. This feature allows an operator to determine what priority strategy a CO is currently using to service a priority request.

2.5.1.4 Retrieve Log Data from the PRS

A management station may need to retrieve a PRS's event logs. Examples of events that may need to be logged include the time when a priority request was received, information contained in the priority request, if a service request to a CO was sent, and what parameters were selected for the service request by the PRS. This feature may be used for performance monitoring purposes, such as to determine how often a priority request is received from a PRG, how often a priority request is granted,

and how often a priority strategy is selected.

2.5.2 Interface—Management Station to CO

The following subsections identify and describe the various features that may be offered between a management station and a CO in an SCP system. These features include:

a) Configure Priority Strategies
b) Determine Priority Strategies
c) Monitor the CO
d) Retrieve Log Data from the CO

2.5.2.1 Configure Priority Strategies

A management station needs to define and edit the priority strategies that a CO may implement to provide signal priority. Each priority strategy defines the phases being serviced, the phases to be omitted, the maximum green time that can be reduced, or the maximum green time that can be extended to service the priority. This feature allows an operator to configure the priority strategies so that the traffic signal controller minimizes any negative impacts on the traffic network when servicing a priority request.

2.5.2.2 Determine Priority Strategies

A management station needs to determine what priority strategies have been programmed in a CO. This feature allows an operator to determine what phases are serviced, what phases are omitted, and how much time a green phase can be extended or reduced for each priority strategy.

2.5.2.3 Monitor the CO

A management station needs to monitor the status of service requests at the CO. This feature allows an operator to determine what priority strategy a CO is currently using to service a priority request.

2.5.2.4 Retrieve Log Data from the CO

A management station may need to retrieve a CO's event logs. Examples of events that may need to be logged include the time when a service request was received, information contained in the service request, and which strategy was implemented by the CO. This feature may be used for performance monitoring purposes, such as to determine the effectiveness of the operational strategies and how often a priority request is serviced.

2.5.3 Interface—PRG to PRS

The following subsections identify and describe the various features that may be offered between a PRG and a PRS in an SCP system. These features include:

a) Exchange Priority Requests
b) Exchange Priority Request Status

2.5.3.1 Exchange Priority Requests

A PRG needs to send priority requests to a PRS. A priority request consists of the class of the vehicle requesting priority, strategy selected, time of service desired and the estimated time of departure. This feature provides a PRS with the information necessary to determine if a priority request should be granted.

2.5.3.2 Exchange Priority Request Status

A PRG needs to receive the status of a priority request from a PRS. This feature allows a PRG to verify if a previously sent priority request has been received. The PRG may also use the status to inform the fleet

vehicle or fleet management center who requested preferential treatment if its priority request has been granted or not.

2.5.4 Interface—PRS to CO

The following subsections identify and describe the various features that may be offered between a PRS and a CO in an SCP system. These features include:

a) Exchange Service Requests
b) Exchange Service Request Status

2.5.4.1 Exchange Service Requests

If a priority request is granted, a service request needs to be exchanged between the PRS and the CO. Each service request specifies a priority strategy to be implemented, the time of service requested, and the time of estimated departure. This feature provides the priority strategy to the CO for processing and servicing.

2.5.4.2 Exchange Service Request Status

A PRS and a CO need to exchange the status of a service request. This feature allows the PRS to verify if a previously sent priority request has been received by the CO. The PRS may also use the status to determine if the CO has provided the signal priority.

2.5.5 Backward Compatibility Needs

2.5.5.1 Backward Compatible with NTCIP 1211 v01

A newer transportation system component may need to communicate with other components that conform to NTCIP 1211 v01.

2.6 Security

NTCIP 1211 v02 does not address any security issues. Any security pertaining to protecting the communications within the SCP system should be implemented either physically by protecting the communications access points, or logically by enabling security features associated with the underlying communications protocols.

2.7 Relationship to the ITS National Architecture [Informative]

NTCIP 1211 v02 addresses seven National ITS Architecture Flows. These flows are:

a) Local signal priority request
b) Right-of-way request notification
c) Signal control command
d) Signal control status
e) Traffic control priority request
f) Traffic control priority status
g) Transit vehicle schedule performance

Each Architecture Flow is associated with one of the following interfaces identified within the National ITS Architecture as:

a) Between the Traffic Management Center (Traffic Management Subsystem (TMS)) and the Traffic Signal Controller (Roadway Subsystem (RS))
b) Between the Traffic Management Center (Traffic Management Subsystem (TMS)) and the Transit Management Center (Transit Management Subsystem (TRMS))
c) Between the Transit Vehicle (Transit Vehicle Subsystem (TRVS)) and the Traffic Signal Controller

(Roadway Subsystem (RS))

The National ITS Architecture Flows are identified with the interfaces and user needs (features) as indicated in Table 1.

Table 1 Interface / User Need and Architecture Flow

Interface / User Need	Source	Architecture Flow	Destination	Definition
Management Station – PRS / Manage the PRS	TMS	signal control commands	RS	Control of traffic signal controllers or field masters including clock synchronization.
Management Station – CO / Configure Priority Strategies	TMS	signal control commands	RS	Control of traffic signal controllers or field masters including clock synchronization.
PRG – PRS / Exchange Priority Requests	RS	right-of-way request notification	TMS	Notice that a request has occurred for signal prioritization, signal preemption, pedestrian call, multi-modal crossing activation, or other source for right-of-way.
PRG – PRS / Exchange Priority Requests	TRVS	local signal priority request	RS	Request from a vehicle to a signalized intersection for priority at that intersection.
PRG – PRS / Exchange Priority Requests	TRMS	traffic control priority request	TMS	Request for signal priority at one or more intersections along a particular route.
PRG – PRS / Exchange Priority Requests	TRVS	transit vehicle schedule performance	TRMS	Estimated times of arrival and anticipated schedule deviations reported by a transit vehicle.
PRG – PRS / Exchange Priority Request Status	TMS	traffic control priority status	TRMS	Status of signal priority request functions at the roadside (e.g. enabled or disabled).
Management Station – PRS / Monitor the PRS	RS	signal control status	TMS	Operational and status data of traffic signal control equipment including operating condition and current indications.
Management Station – CO / Monitor the CO	RS	signal control status	TMS	Operational and status data of traffic signal control equipment including operating condition and current indications.

Section 3
Functional Requirements [Normative]

Section 3 defines the Functional Requirements based on the user needs identified in the Concept of Operations (see Section 2). Section 3 includes:

a) A tutorial.
b) Protocol Requirements List (PRL)—A Functional Requirement is a requirement of a given function and therefore is only required to be implemented if the associated functionality (e.g., user need) is selected through the use of the PRL. The PRL also indicates which of the items are mandatory, conditional, or optional. The PRL can be used by procurement personnel to specify the desired features of an SCP system or can be used by a manufacturer to document the features supported by their implementation.
c) Architectural Requirements—These are requirements related to the architectural needs defined in Section 2.4.
d) Data Exchange and Operational Environment Requirements—These are requirements related to the features identified in Section 2.5 that can be realized through a data exchange. For example, this includes the requirement to be able to exchange service requests between a PRS and a CO.
e) Supplemental Non-communications Requirements--These are additional requirements derived from the Concept of Operations that do not fall into one of the above two categories. For example, they include requirements related to clearing expired priority requests, which may be a supplemental requirement to Exchange Priority Requests.
f) Generic Requirements—There are requirements that are generic to all NTCIP field devices. For example, clock synchronization of devices is a requirement that is considered generic to all NTCIP devices. These requirements can be found in Annex G.

Section 3 is intended for all readers, including:

a) Transportation operations managers
b) Transportation operations personnel
c) Transportation engineers
d) System integrators
e) Device manufacturers

For the first three categories of readers, Section 3 is useful in understanding the details that NTCIP 1211 v02 requires of an SCP. For these readers, Section 3.3.3 is particularly useful in preparing procurement specifications and assist in mapping the various rows of this table to the more detailed text contained within the other sections.

For the last two categories of readers, this section is useful to fully understand what is required of equipment meeting this interface standard. The table in Section 3.3.3 may be used to document the capabilities of their implementations.

3.1 Tutorial [Informative]

This Functional Requirements section defines the formal requirements that are intended to satisfy the user needs identified in Section 2. This is achieved through the development of a PRL that traces each user need to one or more requirements defined in this section. The details of each requirement are then presented following the PRL. The functional requirements are presented in three broad categories as follows:

a) Architectural Requirements—These requirements define the required behavior of the system in exchanging data across the communications interface, including any restrictions to general architectural requirements, based upon the architectural needs identified in the Concept of Operations.
b) Data Exchange Requirements—These requirements define the required behavior of the system in exchanging data across the communications interface based upon the features identified in the Concept of Operations.
c) Supplemental Requirements—These requirements define additional requirements of the system that are derived from the architectural and/or data exchange requirements, but are not themselves architectural or data exchange requirements. A given supplemental requirement may relate to multiple architectural and/or data exchange requirements. Supplemental requirements include capabilities of the equipment (e.g., service processing or clearing expired priority requests).

3.2 Scope Of The Interface [Informative]

<In the opinion of the responsible NTCIP working group, this section does not apply in the context of NTCIP 1211 v02.>

3.3 Protocol Requirements List (PRL)

The PRL, provided in the table defined in Section 3.3.3, maps the user needs defined in Section 2 to the requirements defined in Section 3. The PRL can be used by:

a) A user or specification writer to indicate which requirements are to be implemented in a project-specific implementation.
b) The protocol implementer, as a checklist to reduce the risk of failure to conform to the standard through oversight.
c) The supplier and user, as a detailed indication of the capabilities of the implementation.
d) The user, as a basis for initially checking the potential interoperability with another implementation.

3.3.1 Notation [Informative]

The following notations and symbols are used to indicate status and conditional status in the PRL within all NTCIP standards. Not all of these notations and symbols may be used within NTCIP 1211 v02.

3.3.1.1 Conformance Symbols

The symbols in Table 2 are used to indicate status under the Conformance column in the PRL.

Table 2 Conformance Symbols

Symbol	Status
M	Mandatory
M.#	Support of every item of the group labeled by the same numeral # is required, but only one is active at a time
O	Optional
O.# (range)	Part of an option group. Support of the number of items indicated by the '(range)' is required from all options labeled with the same numeral #
C	Conditional
N/A	Not-applicable (i.e. logically impossible in the scope of the standard)
X	Excluded or prohibited

The O.# (range) notation is used to show a set of selectable options (e.g., O.2 (1..*) would indicate that one or more of the option group 2 options shall be implemented). Two character combinations are used for dynamic requirements. In this case, the first character refers to the static (implementation) status, and the second refers to the dynamic (use); thus "MO" means "mandatory to be implemented, optional to be used."

3.3.1.2 Conditional Status Notation

The following predicate notations in Table 3 may be used.

Table 3 Conditional Status Notation

<predicate>:	This notation introduces a single item that is conditional on the <predicate>.
<predicate>::	This notation introduces a table or a group of tables, all of which are conditional on the <predicate>.
(predicate)	This notation introduces the first occurrence of the predicate. The feature associated with this notation is the base feature for all options that have this predicate in their conformance column.

The <predicate>: notation means that the status following it applies only when the PRL states that the feature or features identified by the predicate are supported. In the simplest case, <predicate> is the identifying tag of a single PRL item. The <predicate>:: notation may precede a table or group of tables in a section or subsection. When the group predicate is true then the associated section shall be completed. The symbol <predicate> also may be a Boolean expression composed of several indices. "AND", "OR", and "NOT" shall be used to indicate the Boolean logical operations.

3.3.1.3 Support Column Symbols

The Support column in the PRL can be used by a procurement specification to identify the required features for the given procurement or by an implementer to identify which features have been implemented. In either case, the user circles the appropriate answer (Yes, No, or N/A) in the support column:

Table 4 Support Column Entries

Yes	Supported by the implementation.
No	Not supported by the implementation.
N/A	Not applicable

3.3.2 Instructions for Completing the PRL [Informative]

In the ‘Support’ column, each response shall be selected either from the indicated set of responses (for example: Yes / No / NA), or it shall reference additional items that are to be attached (for example, list of traffic signal controllers to be supported by an implementation).

If a conditional requirement is inapplicable, use the Not Applicable (NA) choice. If a mandatory requirement is not satisfied, exception information shall be supplied by entering a reference Xi, where i is a unique identifier, to an accompanying rationale for the non-conformance. When the status is expressed as a two-character combination (as defined in 3.3.3.1 above), the response shall address each element of the requirement; e.g., for the requirement "mo," the possible compliant responses are "yy" or "yn."

> Note: An agency specification can allow for flexibility in a deliverable by leaving the selection in the Support column blank for a given row.

3.3.2.1 Conformance Definition

To claim "Conformance" to NTCIP 1211 v02, the vendor shall minimally fulfill the mandatory requirements as identified in the PRL table (See Table 5).

> Note: The reader and user of NTCIP 1211 v02 is advised that 'conformance' to NTCIP 1211 v02 should not be confused with 'compliance' to a specification. NTCIP 1211 v02 is as broad as possible to allow a very simple SCP implementation to be 'conformant' to NTCIP 1211 v02. An agency specification needs to identify the requirements of a particular project and needs to require the support of those requirements. A specification writer is advised to match the requirements of a project with the corresponding standardized requirements defined in NTCIP 1211 v02 to achieve interoperability. This means that functions and requirements defined as 'optional' in NTCIP 1211 v02 might need to be selected in a specification (in effect made 'mandatory' for the project-specific specification).

A conformant device may offer additional (optional) features, as long as they are conformant with the requirements of NTCIP 1211 v02 and the standards it references (e.g., NTCIP 1201 v03 and NTCIP 2301 v02). For example, to claim conformance to additional features, an implementation shall conform to all of the mandatory requirements that trace to the subject user needs in the PRL, AND shall fulfill the requirement by using all of the dialogs and data elements traced to the subject requirement in the Requirements Traceability Matrix (RTM) in Annex A.

A device may also support data that has not been defined by NTCIP 1211 v02; however, when exchanged via one of the NTCIP 2301 v02 protocols, the data shall be properly registered with a valid OBJECT IDENTIFIER under the Global ISO Naming Tree.

> Note: Off-the-shelf interoperability and interchangeability can only be obtained through well documented features broadly supported by the industry as a whole. Designing a system that uses features not defined in a standard or not typically deployed in combination with one another inhibits the goals of interoperability and interchangeability, especially if the documentation of these features is not available for distribution to system integrators. Standards allow the use of additional features to support innovation, which is constantly needed within the industry; but users should be aware of the risks involved with using such features.

3.3.2.2 Backward Compatibility and Support for Different Versions of NTCIP 1211

In NTCIP 1211 v02, the enhancement of certain functions caused corresponding objects to be added and modified. A device conformant with NTCIP 1211 v02 shall by default support functions (and resulting objects) from all existing versions, if said device is required to support that particular functionality.

For example, NTCIP 1211 v02 includes an additional object to support an absolute time reference in the priority request message. NTCIP 1211 v01 does not contain this additional object. To provide maximum backward compatibility, a field device that wants to claim conformance to NTCIP 1211 v02, but also wishes to exchange priority request messages with a field device that conforms with NTCIP 1211 v01, is required to support both priority request messages.

However, a specification writer might determine that support of an older version is not required and may state this within the PRL table (the table contains within the Additional Specifications column statements where a user can de-select the support of any existing version).

3.3.3 Protocol Requirements List (PRL) Table

In addition to the Conformance column and the Support column, which were discussed in Sections 3.3.1 and 3.3.2, the additional columns in the PRL table are the user needs columns, requirements columns and the additional specifications column.

> Note: Visit www.ntcip.org for information on availability of electronic copies of the PRLs.

3.3.3.1 User Needs Column

The user needs are defined within Section 2 and the PRL is based upon the user needs within that Section. The section number and user need name are indicated within these columns.

3.3.3.2 Requirements Column

The requirements are defined within Section 3 and the PRL references the traces from user needs to these requirements. The section number and functional requirements name are indicated within these columns.

3.3.3.3 Additional Specifications Column

The "Additional Specifications" column may (and should) be used by a procurement specification to provide additional notes and requirements for the product to be procured or may be used by an implementer to provide any additional details about the implementation. In some cases, default text already exists in this field, which the user should complete to fully specify the equipment. However, additional text can be added to this field as needed to fully specify a feature.

Table 5 Protocol Requirements List (PRL)

Protocol Requirements List (PRL)						
User Need ID	**User Need**	**FR ID**	**Functional Requirement**	**Conformance**	**Support**	**Additional Specifications**
2.4	Architectural Needs					
2.4.1	Integral Entities			C	Yes / NA	Where two entities are integral to the same physical device, the interface between these entities is implementation-specific.
2.4.2	Provide Live Data			M	Yes	
		3.4.1.1	Provide Data	M	Yes	
		3.4.1.2	Receive Data	M	Yes	
		3.4.1.3	Explore Data	M	Yes	
		3.6.1	Response Time for Requests	M	Yes	The Response Time for all requests shall be ___ milliseconds (25-500: Default=100).
2.4.3	Support Multiple Instances of an Entity			M	Yes	
		3.4.1.1	Provide Data	M	Yes	An agent shall be capable of providing data to at least ___ (1-10:Default=10) managers at any time.
		3.4.1.2	Receive Data	M	Yes	An agent shall be capable of receiving data from at least ___ (1-10:Default=10) managers at any time.
		3.4.1.3	Explore Data	M	Yes	An agent shall be capable of dynamically providing data to at least ___ (1-10:Default=10) managers at any time.
2.4.4	Provide Compressed Data					
2.4.4.1	Provide Compressed Data between a Management Station and a PRS			M	Yes	

Protocol Requirements List (PRL)						
User Need ID	**User Need**	**FR ID**	**Functional Requirement**	**Conformance**	**Support**	**Additional Specifications**
		3.5.1.1	Set Reservice Period	M	Yes	
		3.5.1.2	Set Time To Live Period	M	Yes	
		3.5.1.3.1	Retrieve Priority Request Settings	M	Yes	
2.4.4.2	Provide Compressed Data between a Management Station and a CO			M	Yes	
		3.5.2.1.1	Set Priority Strategy Configuration	M	Yes	
		3.5.2.2.1	Retrieve Priority Strategy Settings	M	Yes	
2.5	Features					
2.5.1	Interface – Management Station to PRS			M	Yes	
2.5.1.1	Manage the PRS			M	Yes	
2.5.1.1.1	Determine PRS Identity			C	Yes / No / NA	Note: This may be NA if the PRS is integral to the traffic signal controller and the traffic signal controller already supports Device Identity.
		H.2.1	Determine Device Component Information	M	Yes	
		H.2.3	Determine Supported Standards	M	Yes	
		H.2.4	Determine System Name	O	Yes / No	
2.5.1.1.2	Determine PRS Configuration			C	Yes / No / NA	Note: This may be NA if the PRS is integral to the traffic signal controller and the traffic signal controller already supports Device Configuration.
		H.2.2	Determine Device Configuration Identifier	M	Yes	
2.5.1.1.3	Configure Reservice Period			M	Yes	

Protocol Requirements List (PRL)						
User Need ID	**User Need**	**FR ID**	**Functional Requirement**	**Conformance**	**Support**	**Additional Specifications**
		3.5.1.1	Set Reservice Period	M	Yes	
2.5.1.1.4	Configure Time To Live Period			M	Yes	
		3.5.1.2	Set Time To Live Period	M	Yes	
		3.6.2.2	Clear Expired Priority Requests	M	Yes	
2.5.1.1.5	PRS Clock Synchronization			C	Yes / No / NA	Note: This may be NA if the PRS is internal to the traffic signal controller and the traffic signal controller already supports clock synchronization.
		H.2.5.1	Set Time	M	Yes	
		H.2.5.2	Set Time Zone	M	Yes	
		H.2.5.3	Set Daylight Savings Mode	M	Yes	
		H.2.5.4	Verify Current Time	M	Yes	
2.5.1.2	Determine Priority Request Criteria			M	Yes	
		3.5.1.3.1	Retrieve Priority Request Settings	M	Yes	
		3.5.1.3.2	Retrieve Reservice Period for a Vehicle Class	M	Yes	
		3.5.1.3.3	Retrieve Priority Request Time To Live Value	M	Yes	
2.5.1.3	Monitor the PRS			O	Yes / No	
		3.5.1.4	Monitor the Status of the PRS	M	Yes	
2.5.1.4	Retrieve Log Data from the PRS			C	Yes / No / NA	Note: This may be NA if the PRS is integral to the traffic signal controller and the traffic signal controller already supports event logging.
		H.2.5.1	Set Time	M	Yes	
		H.2.5.2	Set Time Zone	M	Yes	
		H.2.5.3	Set Daylight Savings Mode	M	Yes	

Protocol Requirements List (PRL)						
User Need ID	**User Need**	**FR ID**	**Functional Requirement**	**Conformance**	**Support**	**Additional Specifications**
		H.2.5.4	Verify Current Time	M	Yes	
		H.2.6.1	Retrieve Current Configuration of Logging Service	M	Yes	
		H.2.6.2	Configure Logging Service	M	Yes	
		H.2.6.3	Retrieve Logged Data	M	Yes	
		H.2.6.4	Clear Log	M	Yes	
		H.2.6.5	Determine Capabilities of Event Logging Service	M	Yes	
		H.2.6.6	Determine Total Number of Logged Events	M	Yes	
		H.2.7.1	Record and Timestamp Events	M	Yes	
		H.2.7.2	Support a Number of Event Classes	M	Yes	The PRS shall support at least ____ event classes.
		H.2.7.3	Support a Number of Event Types to Monitor	M	Yes	The PRS shall support at least ____ event types.
		H.2.7.4.1	Support On-Change Events	M	Yes	
		H.2.7.4.2	Support Greater Than Events	M	Yes	
		H.2.7.4.3	Support Less Than Events	M	Yes	
		H.2.7.4.4	Support Hysteresis Events	M	Yes	
		H.2.7.4.5	Support Periodic Events	M	Yes	
		H.2.7.4.6	Support Bit-flag Events	M	Yes	
		H.2.7.4.7	Support Event Monitoring on Any Data	M	Yes	
		H.2.8	Support a Number of Events to Store in Log	M	Yes	The PRS shall be capable of storing at least ____ events in the event log file.
2.5.2	Interface – Management Station to CO			M	Yes	

Protocol Requirements List (PRL)						
User Need ID	**User Need**	**FR ID**	**Functional Requirement**	**Conformance**	**Support**	**Additional Specifications**
2.5.2.1	Configure Priority Strategies			M	Yes	Note: The definition and selection of the strategy is system- and implementation-specific, and may vary from system to system. The user should be aware that differences in definition and selection may result in an interoperability issue.
		3.5.2.1.1	Set Priority Strategy Configuration	M	Yes	
		3.5.2.1.2	Define Default Coordination Pattern	M	Yes	
		3.5.2.1.3	Define Maximum Priority Strategies Supported	O	Yes / No	
		3.5.2.1.4	Define Maximum Service Requests To Consider	O	Yes / No	
2.5.2.2	Determine Priority Strategies			M	Yes	
		3.5.2.2.1	Retrieve Priority Strategy Settings	M	Yes	
		3.5.2.2.2	Retrieve Priority Strategies	M	Yes	
		3.5.2.2.3	Retrieve Priority Splits	M	Yes	
		3.5.2.2.4	Retrieve Default Coordination Pattern	M	Yes	
		3.5.2.2.5	Retrieve Maximum Priority Strategies Supported	O	Yes / No	
		3.5.2.2.6	Retrieve Maximum Service Requests To Consider	O	Yes / No	
2.5.2.3	Monitor the CO			M	Yes	
		3.5.2.3	Monitor the Status of the CO	M	Yes	
2.5.2.4	Retrieve Log Data from the CO			C	Yes / No / NA	Note: This may be NA if the traffic signal controller already supports event logging.

Protocol Requirements List (PRL)						
User Need ID	**User Need**	**FR ID**	**Functional Requirement**	**Conformance**	**Support**	**Additional Specifications**
		H.2.5.1	Set Time	M	Yes	
		H.2.5.2	Set Time Zone	M	Yes	
		H.2.5.3	Set Daylight Savings Mode	M	Yes	
		H.2.5.4	Verify Current Time	M	Yes	
		H.2.6.1	Retrieve Current Configuration of Logging Service	M	Yes	
		H.2.6.2	Configure Logging Service	M	Yes	
		H.2.6.3	Retrieve Logged Data	M	Yes	
		H.2.6.4	Clear Log	M	Yes	
		H.2.6.5	Determine Capabilities of Event Logging Service	M	Yes	
		H.2.6.6	Determine Total Number of Logged Events	M	Yes	
		H.2.7.1	Record and Timestamp Events	M	Yes	
		H.2.7.2	Support a Number of Event Classes	M	Yes	The CO shall support at least ____ event classes.
		H.2.7.3	Support a Number of Event Types to Monitor	M	Yes	The CO shall support at least ____ event types.
		H.2.7.4.1	Support On-Change Events	M	Yes	
		H.2.7.4.2	Support Greater Than Events	M	Yes	
		H.2.7.4.3	Support Less Than Events	M	Yes	
		H.2.7.4.4	Support Hysteresis Events	M	Yes	
		H.2.7.4.5	Support Periodic Events	M	Yes	
		H.2.7.4.6	Support Bit-flag Events	M	Yes	
		H.2.7.4.7	Support Event Monitoring on Any Data	M	Yes	
		H.2.8	Support a Number of Events to Store in Log	M	Yes	The CO shall be capable of storing at least ____ events in the event log file.

Protocol Requirements List (PRL)						
User Need ID	**User Need**	**FR ID**	**Functional Requirement**	**Conformance**	**Support**	**Additional Specifications**
2.5.3	Interface – PRG to PRS			C	Yes / No / NA	If the PRG and PRS are integral to the same physical device, the interface between these entities is implementation-specific.
2.5.3.1	Exchange Priority Requests			M	Yes	
		3.5.3.1.1	Initiate a Priority Request	M	Yes	
		3.5.3.1.2	Send a Priority Request Update	M	Yes	
		3.5.3.1.3	Send a Cancel Priority Request	M	Yes	
		3.5.3.1.4	Send a Clear Priority Request	M	Yes	
		3.6.2.1	Support Multiple Priority Requests	M	Yes	The PRS shall be capable of supporting at least ___ (1-10:Default=10) and no more than ____ (1-10:Default=10) priority requests.
2.5.3.2	Exchange Priority Request Status			M	Yes	
		3.5.3.2	Receive Priority Request Status	M	Yes	
2.5.4	Interface – PRS to CO			C	Yes / No / NA	If the PRS and CO are integral to the same physical device, the interface between these entities is implementation-specific.
2.5.4.1	Exchange Service Requests			M	Yes	
		3.5.4.1	Exchange Service Request	M	Yes	The PRS or the CO shall poll each other no less than once per _____ milliseconds (100-1000: Default=100).
		3.6.3	Process Service Requests	M	Yes	
2.5.4.2	Exchange Service Request Status			M	Yes	
		3.5.4.2	Exchange Service Request Status	M	Yes	

Protocol Requirements List (PRL)						
User Need ID	**User Need**	**FR ID**	**Functional Requirement**	**Conformance**	**Support**	**Additional Specifications**
2.5.4	Backward Compatibility Needs					
2.5.5.1	Backward Compatible with NTCIP 1211 v01			O	Yes / No	Note: These object definitions have not been deprecated to address interoperability issues with NTCIP 1211 v01. The associated objects were deprecated and replaced by newer objects that have a wider scope or that have been changed to ease implementation. Pay close attention to the implementation and interoperability of these objects.
		3.5.3.1.5	Initiate a Priority Request—NTCIP 1211 v01	C	Yes / NA	If the PRG and PRS are integral to the same physical device, the interface between these entities is implementation-specific.
		3.5.3.1.6	Send a Priority Request Update—NTCIP 1211 v01	C	Yes / NA	If the PRS and CO are integral to the same physical device, the interface between these entities is implementation-specific.
		3.6.2.3	Support Multiple Priority Requests—NTCIP 1211 v01	M	Yes	The PRS shall be capable of supporting at least ___ (1-10:Default=10) and no more than ____ (1-10:Default=10) priority requests.

3.4 Architectural Requirements

Some architectural needs are fully met through the generic architectural requirements defined in Annex G. Only architectural requirements unique to NTCIP 1211 v02 are defined in this section.

3.4.1 Support Communications From Multiple Entities

Requirements for responding to requests follow.

3.4.1.1 Provide Data

An entity acting as an agent shall be able to provide to more than one entity any set of data requested by the entities, each acting as a manager, e.g., a management station, at any time.

3.4.1.2 Receive Data

An entity acting as an agent shall be able to receive data (e.g., configuration data, commands, etc.) from more than one entity, acting as a manager, e.g., a management station, at any time.

3.4.1.3 Explore Data

An entity acting as an agent shall be able to dynamically provide to more than one entity what data and data instances are requested by the entities, each acting as a manager, e.g., a management station, at any time.

3.5 Data Exchange and Operational Environment Requirements

The operation of an SCP system has been categorized into the four interfaces covered by NTCIP 1211 v02:

a) Interface—Management Station to PRS
b) Interface—Management Station to CO
c) Interface—PRG to PRS
d) Interface—PRS to CO

In the Concept of Operations (Section 2), each interface has been broken down into subsections. The Data Exchange Requirements also follow this structure.

3.5.1 Interface—Management Station to PRS

The requirements for data exchanges between a management station and a PRS follow.

3.5.1.1 Set Reservice Period

The PRS shall allow the management station to configure the reservice period, in seconds, for all vehicle classes. The reservice period defines the minimal amount of time that shall pass before another priority request is serviced. A different reservice period can be defined for each of ten vehicle classes.

3.5.1.2 Set Time To Live Period

The PRS shall allow the management station to configure the time-to-live period, in seconds. The time-to-live period defines the maximum amount of time a PRS considers a priority request for servicing.

3.5.1.3 Retrieve Priority Request Server Settings

The requirements for a management station to retrieve the configuration of a PRS follow.

3.5.1.3.1 Retrieve Priority Request Settings

The management station shall retrieve from the PRS the reservice period, in seconds, for each vehicle class and the time, in seconds, that a priority request may exist in the PRS.

3.5.1.3.2 Retrieve Reservice Period for a Vehicle Class

The management station shall retrieve from the PRS the criteria used to determine the reservice period, in seconds, for a specific vehicle class.

3.5.1.3.3 Retrieve Priority Request Time To Live Value

The management station shall retrieve from the PRS the maximum amount of time, in seconds, that the PRS considers a priority request after receiving the priority request.

3.5.1.4 Monitor the Status of the PRS

The management station shall monitor the PRS to determine what priority strategy is being implemented, if any. The PRS status information consists of the status, the priority strategy requested, the time of service desired, and the time of estimated departure for each priority request.

3.5.2 Interface—Management Station to CO

The requirements for data exchanges between a management station and a CO follow.

3.5.2.1 Configure the CO

The requirements for a management station to set the configuration of a CO to service priority requests follow.

3.5.2.1.1 Set Priority Strategy Configuration

The CO shall allow a management station to define the parameters for each priority strategy supported by the CO. The parameters for each priority strategy are the phases affected, the phases omitted, the maximum extension of the green time allowed, in seconds, and the maximum reduction of the green time allowed, in seconds.

3.5.2.1.2 Define Default Coordination Pattern

The CO shall allow a management station to define the default coordination pattern to be used when the traffic signal controller is not operating in a coordinated mode.

3.5.2.1.3 Define Maximum Priority Strategies Supported

The CO shall allow a management station to define the maximum number of priority strategies that the CO supports.

3.5.2.1.4 Define Maximum Service Requests To Consider

The CO shall allow a management station to define the maximum number of service requests that the CO considers when selecting a priority strategy to execute.

3.5.2.2 Retrieve Priority Strategy Configuration

The requirements for a management station to retrieve the configuration of a CO follow.

3.5.2.2.1 Retrieve Priority Strategy Settings

The management station shall retrieve from the CO the phases affected, the maximum extension allowed, in seconds, and the maximum reduction allowed, in seconds, for each priority strategy supported by the CO.

3.5.2.2.2 Retrieve Priority Strategies

The management station shall retrieve from the CO the phases to be serviced during priority service, the phases to be omitted during priority service, the pedestrian movements to be omitted during priority service, and a description of the priority strategy for a specific priority strategy.

3.5.2.2.3 Retrieve Priority Splits

The management station shall retrieve from the CO the maximum extension allowed, in seconds, and the maximum reduction allowed, in seconds, for a priority strategy.

3.5.2.2.4 Retrieve Default Coordination Pattern

The management station shall retrieve from the CO the default coordination pattern to be used if the traffic signal controller is not operating in a coordinated mode.

3.5.2.2.5 Retrieve Maximum Priority Strategies Supported

The management station shall retrieve from the CO the maximum number of priority strategies supported by the CO.

3.5.2.2.6 Retrieve Maximum Service Requests To Consider

The management station shall retrieve from the CO the maximum number of service requests that the CO considers when selecting a priority strategy to execute.

3.5.2.3 Monitor the Status of the CO

The management station shall monitor the CO to determine what priority strategy is being implemented, if any. The CO status information consists of the status, the priority strategy requested, the time of service desired, and the time of estimated departure for each priority request.

3.5.3 Interface – PRG to PRS

The requirements for data exchanges between a PRG and a PRS follow.

3.5.3.1 Receive Priority Requests

The requirements for exchanging priority requests between a PRG and a PRS follow.

3.5.3.1.1 Initiate a Priority Request

A PRG shall send a priority request message to a PRS to initiate a new priority request. The priority request information consists of a unique priority request identification number, the identification number of the vehicle making the request, the class type of the vehicle making the request, the class level of the vehicle making the request, the strategy number requested, the time of service desired (in seconds), the estimated time of departure (in seconds), and the absolute time reference used by the PRG.

3.5.3.1.2 Send a Priority Request Update

A PRG shall send a priority request update message to a PRS to update the time of service desired, in seconds, and the estimated time of departure, in seconds, for a previously sent priority request. The priority request update message also consists of the unique priority request identification number, the identification number of the vehicle making the request, the class type of the vehicle making the request, the strategy number requested, and the absolute time reference used by the PRG.

3.5.3.1.3 Send a Cancel Priority Request

A PRG shall send a cancel priority request message to a PRS to cancel a previously sent priority request. The cancel priority request message consists of the unique priority request identification number, the identification number of the vehicle making the request, the class type of the vehicle making the request, and the strategy number requested of the priority request to be cancelled.

3.5.3.1.4 Send a Clear Priority Request

A PRG shall send a clear priority request message to a PRS to clear all information in a previously sent priority request. The clear priority request message consists of the unique priority request identification number, the identification number of the vehicle making the request, the class type of the vehicle making the request, and the strategy number requested of the priority request to be cleared.

3.5.3.1.5 Initiate a Priority Request—NTCIP 1211 v01

A PRG shall send a priority request message conformant to NTCIP 1211 v01 to a PRS to initiate a new priority request. The priority request information consists of a unique priority request identification number, the identification number of the vehicle making the request, the class type of the vehicle making the request, the class level of the vehicle making the request, the strategy number requested, the time of service desired (in seconds), and the estimated time of departure (in seconds).

3.5.3.1.6 Send a Priority Request Update—NTCIP 1211 v01

A PRG shall send a priority request update message conformant to NTCIP 1211 v01 to a PRS to update the time of service desired, in seconds, and the estimated time of departure, in seconds, for a previously sent priority request. The priority request update message also consists of the unique priority request identification number, the identification number of the vehicle making the request, the class type of the vehicle making the request, and the strategy number requested of the priority request to be updated.

3.5.3.2 Receive Priority Request Status

A PRG shall receive the status of a priority request from the PRS. The status of a priority request consists of the unique priority request identification number, the identification number of the vehicle making the request, the class type of the vehicle making the request, the strategy number requested, and the status of the priority request.

3.5.4 Interface—PRS to CO

The requirements for data exchanges between the PRS and the CO follow.

3.5.4.1 Exchange Service Request

A PRS and a CO shall exchange service request information. The service request information consists of the priority strategy selected, time of service requested, in seconds, and time of estimated departure, in seconds. Upon receiving the service request information, the CO may adjust the signal timing to provide priority while maintaining coordination.

3.5.4.2 Exchange Service Request Status

A PRS and a CO shall exchange the status of a service request. The status of a service request consists of the status, the priority strategy requested, the time of service desired, and the time of estimated departure for each service request.

3.6 Supplemental Non-Communications Requirements

Supplemental requirements for SCP follow. These requirements do not directly involve communications via the communications interfaces addressed by NTCIP 1211 v02, but, if the supplemental requirement is selected in the PRL, the implementation shall fulfill the stated requirement to claim conformance to NTCIP 1211 v02.

3.6.1 Response Time for Requests

The SCP shall process all requests in accordance with all of the rules of the relevant base standards (i.e., NTCIP 1103 v02 and NTCIP 2301), including updating the value in the database and initiating the transmission of the appropriate response (assuming that the SCP has permission to transmit) within the Response Time. If the specification does not indicate the Response Time, the Response Time shall be 100 milliseconds. The Response Time is measured as the time between the receiving of the last byte of the request and the transmission of the first byte of the response.

3.6.2 Process Priority Requests

The following subsections identify and describe the various requirements to be performed by a PRS to process priority requests.

3.6.2.1 Support Multiple Priority Requests

Upon receiving of a new priority request, a PRS shall check if the new priority request has a higher priority than any other “active” priority requests that have been received. If the new priority request has a higher priority, the PRS services the new priority request, and queues the other active priority requests. The user need is to accept priority requests from more than one PRG, and to provide preferential the priority request

3.6.2.2 Clear Expired Priority Requests

A PRS shall continuously scan all priority requests to determine if a priority request should be cleared automatically. A PRS should automatically clear a priority request if the service time request has been exceeded.

3.6.2.3 Support Multiple Priority Requests—NTCIP 1211 v01

Upon receiving a new NTCIP 1211 v01-conformant priority request message, a PRS shall check if the new priority request has a higher priority than any other “active” priority requests that have been received. If the new priority request has a higher priority, the PRS services the new priority request, and queues the other active priority requests. The user need is to accept priority requests from more than one PRG, and to provide preferential the priority request.

3.6.3 Process Service Requests

Upon receiving a service request, the CO shall process the service request as defined by the priority strategy settings and other parameters defined in the traffic signal controller. The priority strategy settings consist of the phases to be serviced during priority service, the phases to be omitted during priority service, the pedestrian movements to be omitted during priority service, the maximum extension allowed, in seconds, and the maximum reduction allowed, in seconds, time of service requested, in seconds, time of estimated departure, in seconds, and the status of a service request.

The user need is to provide the traffic signal controller with the information needed to provide priority service.

Section 4
Dialogs [Normative]

Section 4 is intended for product developers such as SCP device manufacturers and system integrators. Other parties might find Section 4 and Section 5 helpful to gain a full understanding of the design details of the standard.

Section 4 presents the standardized dialogs (i.e., sequence of data exchanges) that fulfill various Data Exchange requirements defined in Section 3.5.

The NTCIP device standards effort is based on SNMP. SNMP communications between two entities are largely driven by a manager or a management system, which executes applications that monitor and controls the other entity, known as the agent. Thus most of the requirements for a device standard define how the device shall respond to the various possible actions the manager might take.

SNMP offers a high degree of flexibility as to how the manager structures its requests. For example, with SNMP, the manager can do any of the following:

a) Send only those requests that are critical at the current time, whereas a standardized dialog typically sends requests relating to all associated data, regardless of whether it is critical for current purposes
b) Combine a number of requests in a single packet, whereas a standardized dialog dictates the exact contents of each packet
c) Separate a group of requests into multiple packets, whereas a standardized dialog dictates the exact contents of each packet
d) Interweave requests from multiple dialogs, whereas a standardized dialog dictates the exact ordering of messages, which are not interrupted with other messages

This flexibility can be a powerful tool allowing a manager to optimize the use of communications facilities, which is the primary reason that SNMP was chosen as the core NTCIP protocol for devices. However, the flexibility also means that there are numerous allowable variations in the management process that a manager may choose to use.

Unfortunately, this flexibility presents a challenge to ensuring interoperability. While a conformant SCP system is required to support any allowed sequence within NTCIP 1211 v02, ensuring that a given SCP system actually supports every possible combination would be impractical. Instead, most agencies only require that the SCP system be tested to a standard set of procedures, which would use standardized dialogs (as defined in Section 4.2, Annex A, and Annex G). To improve communications efficiency, managers may use non-standard dialogs (e.g., a combination of GET and/or SET requests that is not defined as a standardized dialog, but which a conformant device is required to support according to the ACCESS and SetConstraint rules defined in Section 4.2 and Section 5). Because these more efficient dialogs may not be known until the acquisition of the management station, which may be years after the acquisition of the SCP system, there is a potential for an interoperability problem to arise.

To overcome this complication, this section defines a lowest common denominator approach to the communications interfaces defined by 1211 v02. This section defines the standardized dialog for each Data Exchange Requirement. Managers may support other dialogs to fulfill these same requirements, as long as these dialogs are consistent with the rules defined in NTCIP 1211 v02. Such a manager is termed a ‘consistent manager’. A consistent manager interoperates with any ‘conformant’ device. However, since an agency cannot be certain that a device is 100% conformant to every possible scenario (given practical constraints), interoperability problems could still arise.

A ‘conformant manager’ is required to offer a mode in which it only uses the standardized dialogs as

defined in this section. With this limited definition, there is relatively little variability in what constitutes a conformant manager. Thus, fully testing a manager for conformance is a relatively straight forward process that can be done within the practical constraints faced by most procuring agencies. Thus, a conformant manager provides an agency with a much greater chance of achieving interoperability with off-the-shelf devices that have been tested against NTCIP 1211 v02 and the designation of such a system is intended to provide a guaranteed base level of interoperability.

The rules for the standardized dialogs are as follows:

a) The dialogs are defined by a sequence of GET or SET requests. These requests shall equate to the GET and SET operations defined in Annex G.3.1 and Annex G.3.3, respectively, and shall be sent as a single message.
b) The contents of each request are identified by an object name. Each object name consists of an object type and an instance identifier. Formal definitions of each object type are provided in Section 5 of NTCIP 1211 v02 and NTCIP 1201 v03. The meaning of the instance identifier is provided by these same definitions coupled with standard SNMP rules (see RFC 1212).
c) Each message shall contain all of the objects as shown, unless otherwise indicated.
d) A message shall not contain any other objects.
e) The contents of each message sent by the manager may appear in any order.

   Note: Ideally, the order of objects should match the order as shown in NTCIP 1211 v02, to provide for the highest probability of interoperability. However, it is recognized that many implementations may use off-the-shelf software, which may prevent the designation of an exact ordering of objects and as a result, this ordering is not a requirement of NTCIP 1211 v02.

f) After sending a message, the manager shall not send any other data across the communications channel until the earlier of:
   i) The manager receiving a response from the agent; or
   ii) The expiration of the response time.
g) If the response indicates an error occurred in the operation, the manager shall exit the process, unless specific error-handling rules are specified by the dialog.
h) Dialogs containing a sequence of only GET requests may request objects in any order.

However, since consistent managers can alter the order of requests, NTCIP 1211 v02 defines rules for when certain data exchanges are allowed. Unless otherwise indicated, a conformant device shall allow an object to be retrieved (through a GET request) or altered (through a SET request, if the object is write-able) at any time. However, the access to some data is associated with a state machine and Section 4.3 defines the various rules that apply to these state machines.

4.1 Tutorial [Informative]

The Requirements Traceability Matrix (RTM) presented in Annex A identifies the standardized dialog that can be used to achieve each of the data exchange requirements defined in Section 3.5. Simple data exchange requirements reference one of the generic SNMP dialogs along with a list of data elements (see Annex G). These equate to a single message being sent (e.g., a GET request) containing the referenced data elements followed by the appropriate response per the generic dialog specification.

This section defines the standardized dialogs for the more complicated data exchange requirements. Each of these dialogs is defined by a number of steps. Many of the steps reference data elements that are defined in Section 5. These data elements are also shown in the corresponding row of the RTM along with their precise section number.

The dialogs may also be accompanied by an informative figure that provides a graphical depiction of the normative text. The figures conform to the Unified Modeling Language and depict the manager as an outside actor sending a series of messages to the agent and the agent returning responses. If there is any conflict between the figure and the text, the text takes precedence.

4.2 Specified Dialogs

This section provides the standardized data exchange sequences that can be used by the components of an SCP system to ensure interoperable implementations for the various data exchange requirements identified in Section 3.5. This section only includes dialogs that have special semantics or impose special restrictions on the operations that are allowed.

The dialogs are organized by the four communications interfaces covered by NTCIP 1211 v02:

a) Interface—Management Station to PRS
b) Interface—Management Station to CO
c) Interface—PRG to PRS
d) Interface—PRS to CO

4.2.1 Interface—Management Station to PRS

Section 4.2.1 provides the dialogs that may occur between the management station and the PRS. This section includes only those dialogs that have special semantics or impose special restrictions on the operations are allowed. For this communications interface, the management station is always the manager and the PRS is always the agent.

A management station shall be responsible for configuration and monitoring of any PRS that resides in a Traffic Signal Controller.

4.2.1.1 Configure the PRS

As illustrated in Figure 11, the standardized dialog for a management station to configure the PRS shall be as follows:

a) The management station shall SET prsProgramData.

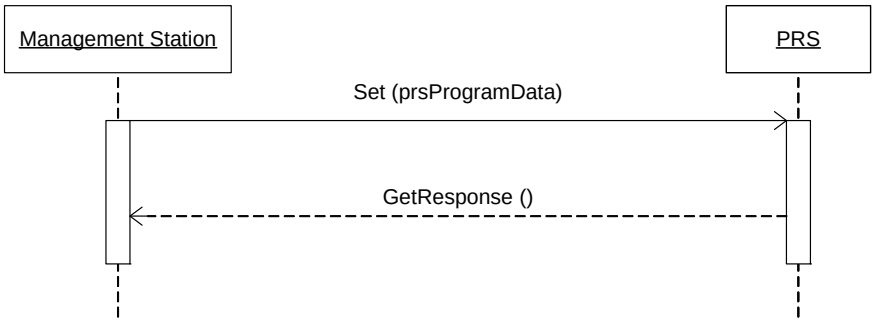

Figure 11 Configuring the PRS Sequence Diagram

4.2.2 Interface—Management Station to CO

Section 4.2.2 provides the dialogs that may occur between the management station and the CO. This section includes only those dialogs that have special semantics or impose special restrictions on the operations are allowed. For this communications interface, the management station is always the manager and the CO is always the agent.

A management station shall be responsible for configuration and monitoring of any CO that resides in a traffic signal controller.

4.2.2.1 Set the Priority Strategy Configuration

4.2.2.1.1 Standardized Dialog

The following is the standardized dialog for a management station to configure the priority strategy settings in a CO. The management station shall use ‘dbCreateTransaction’, as defined in NTCIP 1201 v03 Section 2.3.1, to SET this object. The CO shall NOT allow a normal SNMP SET. The use case diagram for ‘dbCreateTransaction’ is depicted in Figure 12.

The standardized dialog to configure the priority strategy settings in a CO is:

a) The traffic signal controller shall be in transaction mode (See Section 4.2.2.1.3).
b) The management station shall SET coStrategyPlanBlock.x.
c) The management station shall SET coStrategySplitsBlock.x.
d) The management station shall SET priorityStrategyDefaultCoordPattern.
e) The consistency checks to be performed on downloaded data when the "verify" state is commanded is defined in Section 4.2.2.1.2.
f) The controller shall exit transaction mode.

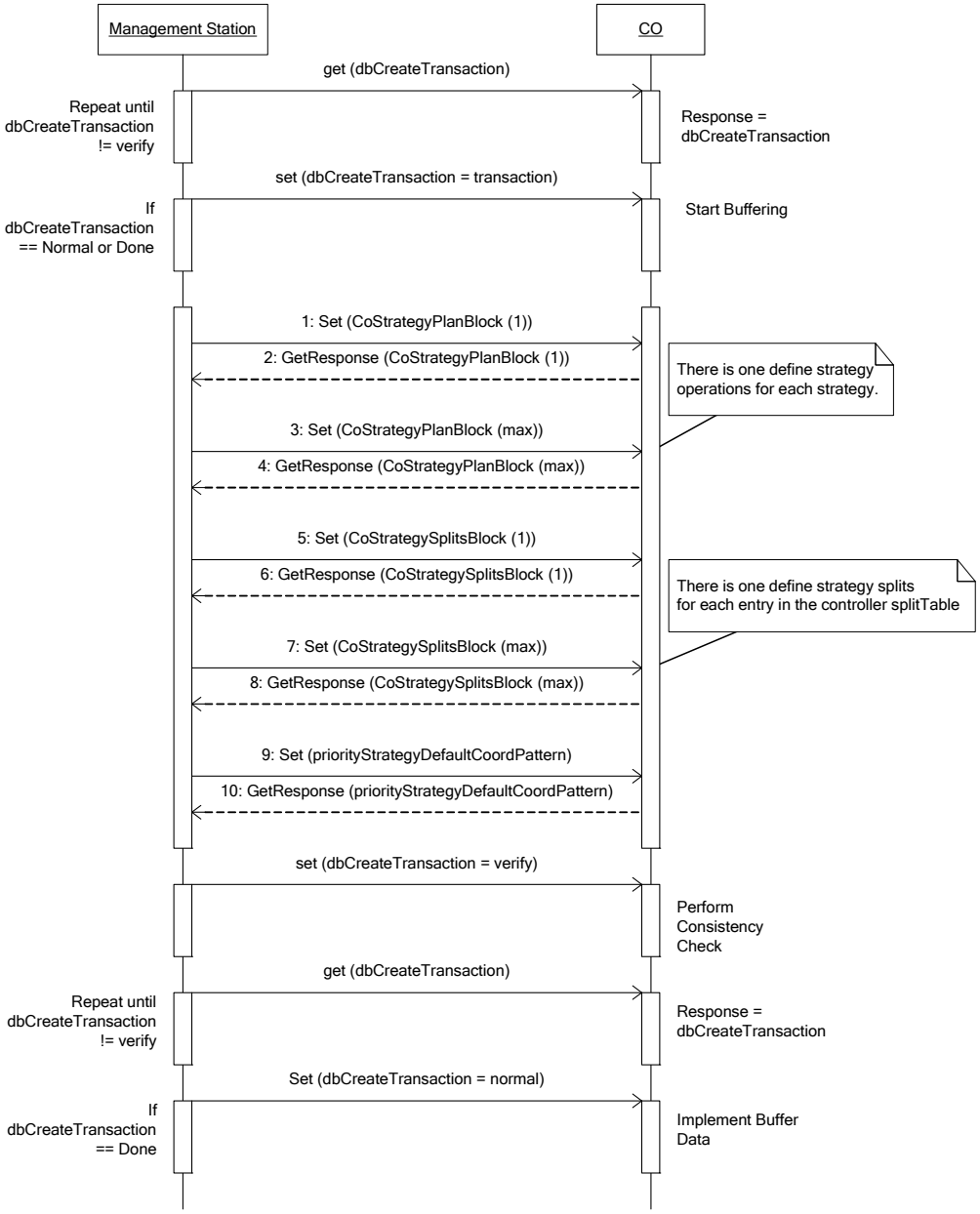

Figure 12 Priority Strategy Settings

4.2.2.1.2 Consistency Check

Consistency checks assure that certain critical objects are checked "in context" and treated as interrelated values rather than separate non-related data items.

When data is downloaded to a CO and the controller is operating in the "transaction" mode, as defined by the dbCreateTransaction object defined in NTCIP 1201 v03, consistency checks shall be performed on downloaded data when the "verify" state is commanded. The consistency checks that shall occur and corresponding error messages are described below. Error messages, if any, may be examined by reading the dbTransactionError object once the CO has entered the "done" mode.

The following rules shall apply to the consistency check:

a) The consistency checks defined in NTCIP 1202:2005 Annex B shall also be performed.
b) The following objects define functionality related to phase assignments. Consistency checks insure that phases specified by these objects may operate concurrently and are defined only once in each string parameter. The value "xx" corresponds to priorityStrategyNumber.

   Priority Strategy Service Phases (priorityStrategyServicePhases)
   Priority Strategy Phase Omits (priorityStrategyPhaseOmits)
   SCP Split Coordinated Phases (splitCoordPhase)

c) When more than one service phase is specified and the defined phases CANNOT time concurrently, the error message, "STRATEGY xx PHASE CG FAULT" shall be returned.
d) When more than one service phase is specified and the defined phases are in the same ring, the error message, ""STRATEGY xx PHASE RING FAULT" shall be returned.
e) When a defined service phase is in the string parameter more than once, the error message, "STRATEGY xx PHASE MULTI FAULT" shall be returned.
f) When a defined service phase is disabled, the error message "STRATEGY PHASE DISABLE FAULT" shall be returned.

Note: The order of the checks is not defined. Therefore, for a given set of 'bad' data, the Error Message between different units may be inconsistent.

4.2.2.1.3 dbCreateTransaction

Figure 13 (a sequence diagram) represents the dialog for using dbCreateTransaction to set data, as defined in NTCIP 1201 v03. If there is any discrepancy regarding dbCreateTransaction between the sequence diagram shown here and NTCIP 1201 v03, NTCIP 1201 v03 shall take precedence.

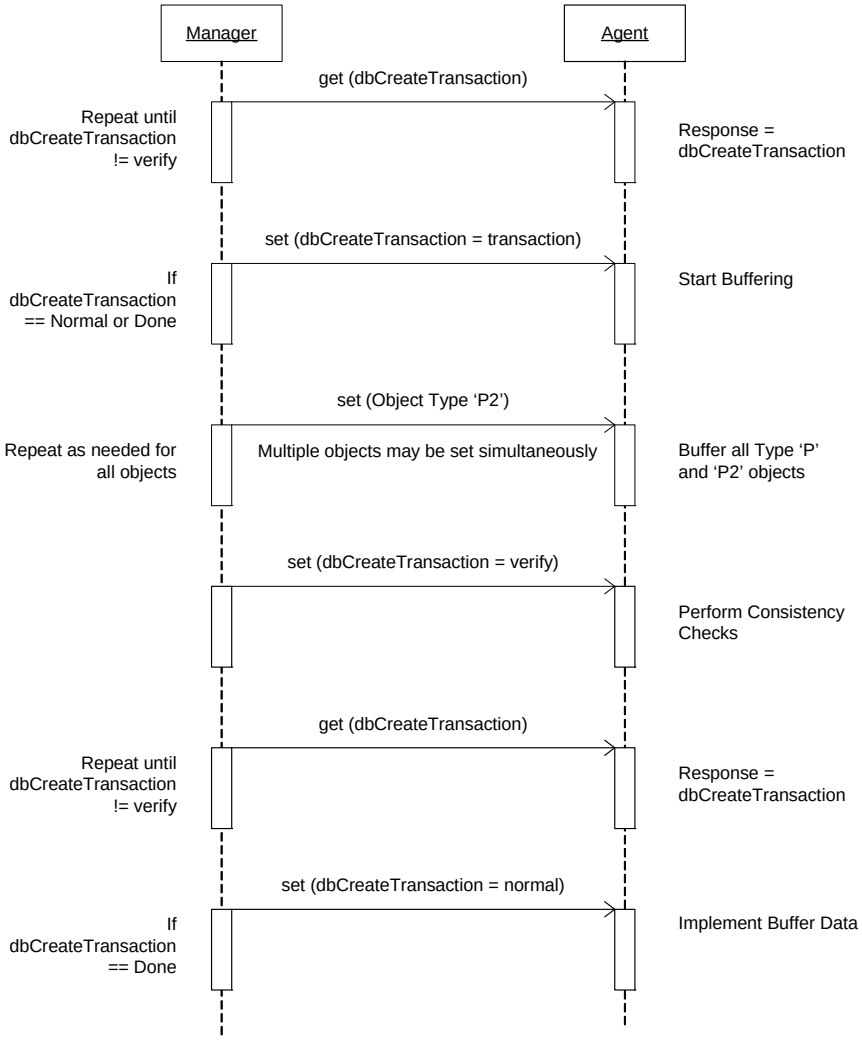

Figure 13 dbCreateTransaction

4.2.2.2 Retrieve Block Object Of Priority Strategy Settings

As illustrated in Figure 14, the standardized dialog for a management station to retrieve the block object of priority strategy settings in a CO shall be as follows:

a) The management station shall SET scpBlockGetControl, with scpBlockDataType = '0x00' and scpBlockDataID = '0x00'.
b) The management station shall GET scpBlockData.
c) The management station shall SET scpBlockGetControl, with scpBlockDataType = '0x00' and scpBlockDataID = '0x01'.
d) The management station shall GET scpBlockData.

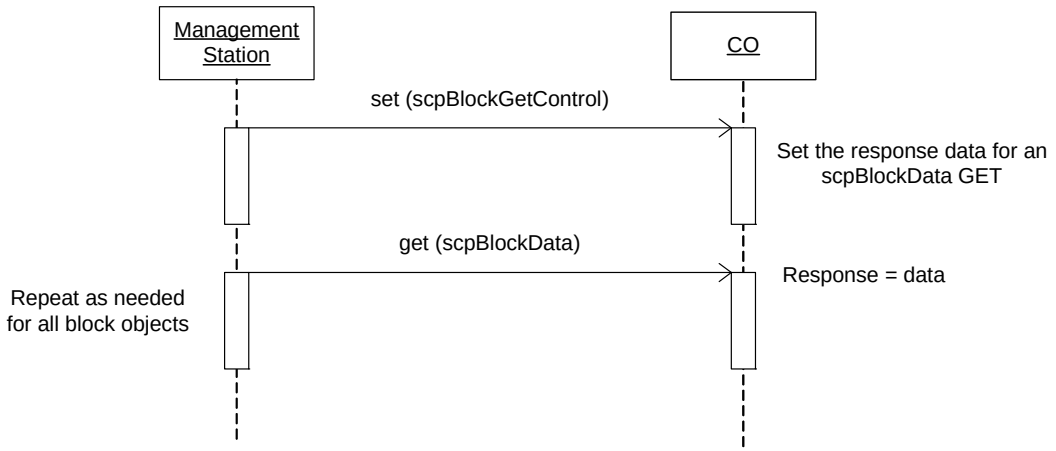

Figure 14 Get Block Data

4.2.3 Interface—PRG to PRS

Section 4.2.3 provides the dialogs that may occur between the PRG and the PRS. This section includes only those dialogs that have special semantics or impose special restrictions on the operations are allowed. For this communications interface, the PRG is always the manager and the PRS is always the agent.

If the PRG and the PRS are implemented as an integral part of the same physical device, the method by which the PRG and the PRS exchange information is implementation specific.

4.2.3.1 Exchange Priority Request

4.2.3.1.1 Standardized Dialog

As illustrated in Figure 15, the standardized dialog for a PRS to receive a priority request from a PRG shall be as follows:

a) The PRG shall SET prgPriorityRequestAbsolute with the desired values. This shall cause the PRS to perform a priority request check definition (see Section 4.2.3.1.2).
b) If the PRS sends a GetResponse with 'noError', the priority request has been accepted and the PRS is waiting for the CO to complete its service processing. The PRG may then exit the process.

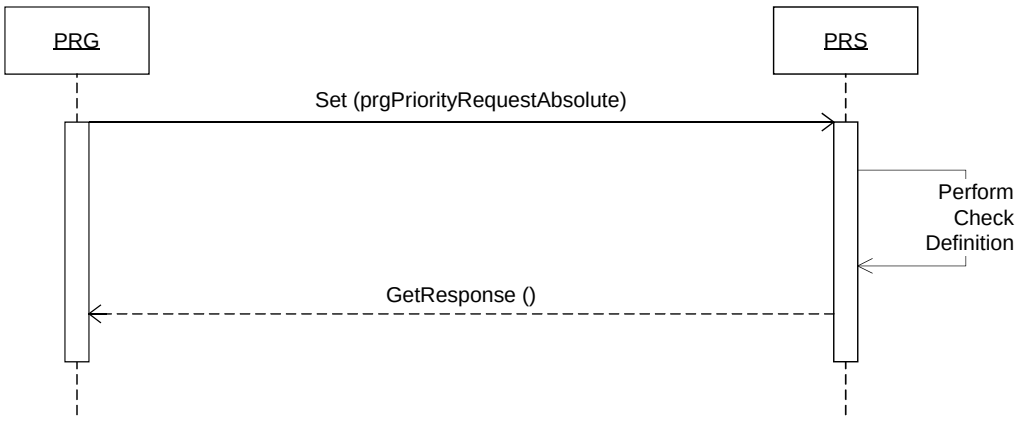

Figure 15 Exchange Priority Request

4.2.3.1.2 Check Definition

Upon receiving a SET prgPriorityRequestAbsolute message from a PRG, the PRS shall perform several checks and processes before responding with an acknowledgement.

a) If the SET length is NOT 29, or if the contents of any message OID field cannot be parsed to fit the SYNTAX defined for the referenced object, then the PRS shall return an Error Status of ‘badValue (3)’ to the PRG and no further processing takes place.
b) If the priorityRequestTable does not have at least one row with a priorityRequestStatusInPRS of ‘idleNotValid’, the PRS shall return an Error Status of ‘noSuchName (2)’ to the PRG and no further processing takes place. (This error-status can be considered as a buffer full error.)
c) If the above checks are completed without error, the PRS shall store the contents of the prgPriorityRequestAbsolute message in the appropriate priorityRequestTableEntry.
d) The PRS shall set priorityRequestTimeOfMessage in the appropriate priorityRequestTableEntry.
e) The PRS shall set priorityRequestTimeToLive in the appropriate priorityRequestTableEntry.
f) The PRS shall set priorityRequestTimeOfServiceDesiredInPRS in the appropriate priorityRequestTableEntry.
g) The PRS shall set priorityRequestTimeOfEstimatedDepartureInPRS in the appropriate priorityRequestTableEntry.
h) The PRS shall check if the reservice time has been exceeded by comparing the appropriate priorityRequestReserviceClass”X”Time against the priorityRequestReserviceTimer. If the priorityRequestReserviceTimer is not greater than the appropriate priorityRequestReserviceClass”X”Time, the PRS shall set priorityRequestStatusInPRS for the appropriate priorityRequestTableEntry to ‘reserviceError (9)’. Otherwise, the PRS shall set priorityRequestStatusInPRS for the appropriate PriorityRequestTableEntry to ‘readyQueued (2)’.
i) If prsBusy is ‘True’ and the status of a lower Class Type request is ‘activeProcessing’ or ‘activeAdjustNotNeeded’, the PRS shall set priorityRequestStatusInPRS of the lower Class Type to ‘activeOverride’.
j) The PRS shall return an Error Status of ‘noError’ to the PRG.

4.2.3.2 Exchange Priority Update

4.2.3.2.1 Standardized Dialog

As illustrated in Figure 16, the standardized dialog for a PRS to receive a priority request update from a PRG shall be as follows:

a) The PRG shall SET prgPriorityUpdateAbsolute with the desired values. This shall cause the PRS to perform a priority request update check definition (See Section 4.2.3.2.2).
b) If the PRS sends a GetResponse with 'noError', the priority request update has been accepted and the PRS is waiting for the CO to complete its service processing. The PRG may then exit the process.

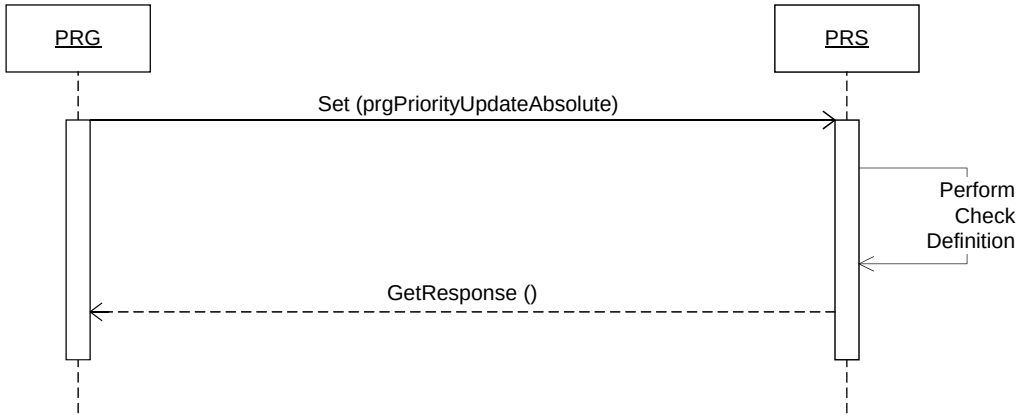

Figure 16 Exchange Priority Request Update

4.2.3.2.2 Check Definition

Upon receiving a SET prgPriorityUpdateAbsolute message from a PRG, the PRS shall perform several checks before responding with an acknowledgement.

a) If the SET length is NOT 29, or if the contents of any message OID field cannot be parsed to fit the SYNTAX defined for the referenced object, then the PRS shall return an Error Status of ‘badValue (3)’ to the PRG and no further processing takes place.
b) The PRS shall check for a matching entry in the priorityRequestTable. A matching entry requires a priorityRequestStatusInPRS of other than ‘idleNotQueued’ and all of the following to have the same values as in the SET prgPriorityUpdateAbsolute message:
    i) priorityRequestID,
    ii) priorityRequestVehicleID,
    iii) priorityRequestVehicleClassType,
    iv) priorityRequestVehicleClassLevel, and
    v) priorityRequestServiceStrategyNumber
c) If the PRS is unable to find a matching entry in the priorityRequestTable, the PRS shall return an Error Status of ‘noSuchName (2)’ to the PRG and no further processing takes place.
d) If the above checks are completed without error, the PRS shall set priorityRequestTimeOfServiceDesired and priorityRequestTimeOfEstimatedDeparture in the appropriate PriorityRequestTableEntry to the values received.
e) The PRS shall set priorityRequestTimeOfServiceDesiredInPRS and priorityRequestTimeOfEstimatedDepartureInPRS in the appropriate priorityRequestTableEntry.
f) The PRS shall return an Error Status of ‘noError’ to the PRG.

4.2.3.3 Exchange Priority Cancel

4.2.3.3.1 Standardized Dialog

As illustrated in Figure 17, the standardized dialog for a PRS to receive a priority request cancel from a PRG shall be as follows:

a) The PRG shall SET prgPriorityCancel with the desired values. This shall cause the PRS to perform a priority request cancel check definition (See Section 4.2.3.3.2).
b) If the PRS sends a GetResponse with 'noError', the priority request cancel has been accepted and the PRS shall attempt to cancel the priority request. The PRG may then exit the process.

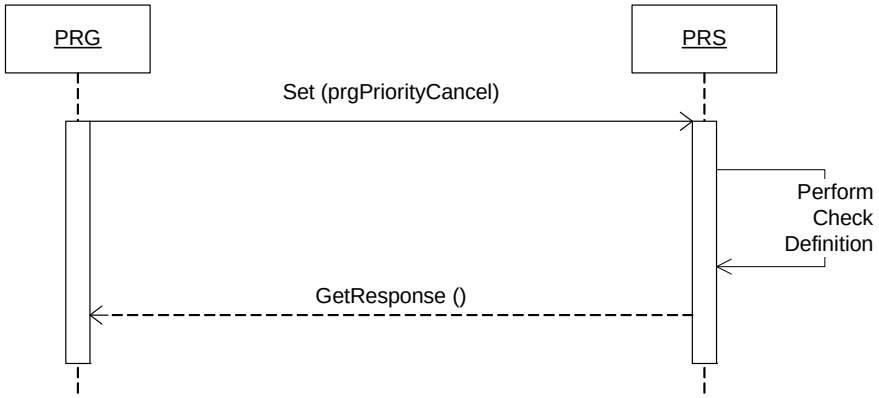

Figure 17 Exchange Priority Request Cancel

4.2.3.3.2 Check Definition

Upon receiving a SET prgPriorityCancel message from a PRG, the PRS shall perform several checks before responding with an acknowledgement.

a) If the SET length is NOT 25, or if the contents of any message OID field cannot be parsed to fit the SYNTAX defined for the referenced object, then the PRS shall return an Error Status of ‘badValue (3)’ to the PRG and no further processing takes place.
b) The PRS shall check for a matching entry in the priorityRequestTable. A matching entry requires a priorityRequestStatusInPRS of other than ‘idleNotQueued’ and all of the following to have the same values as in the SET prgPriorityCancel message:
    i) priorityRequestID,
    ii) priorityRequestVehicleID,
    iii) priorityRequestVehicleClassType,
    iv) priorityRequestVehicleClassLevel, and
    v) priorityRequestServiceStrategyNumber
c) If the PRS is unable to find a matching entry in the priorityRequestTable, the PRS shall return an Error Status of ‘noSuchName (2)’ to the PRG and no further processing takes place.
d) If the above checks are completed without error, the PRS shall return an Error Status of ‘noError’ to the PRG.
e) If priorityRequestStatusInPRS in the appropriate PriorityRequestTableEntry is ‘readyQueued’ or ‘readyOverridden’, then the PRS shall set priorityRequestStatusInPRS to ‘closedCanceled’.
f) If priorityRequestStatusInPRS in the appropriate PriorityRequestTableEntry is ‘activeProcessing’ or ‘activeAdjustNotNeeded’, the PRS shall set priorityRequestStatusInPRS to ‘activeCancel’.

4.2.3.4 Exchange Priority Clear

4.2.3.4.1 Standardized Dialog

The standardized dialog for a PRS to receive a priority request clear from a PRG shall be as follows:

a) The PRG shall SET prgPriorityClear with the desired values. This shall cause the PRS to perform a priority request clear check (see Section 4.2.3.4.2).
b) If the PRS sends a GetResponse with 'noError', the priority request clear has been accepted and the PRS is waiting for the CO to complete its service processing. The PRG may then exit the process.

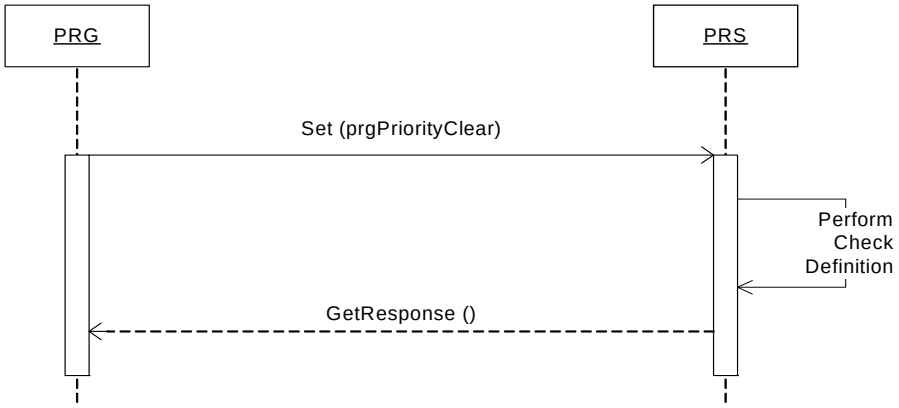

Figure 18 Exchange Priority Request Clear

4.2.3.4.2 Check Definition

Upon receiving a SET prgPriorityClear message from a PRG, the PRS shall perform several checks before responding with an acknowledgement.

a) If the SET length is NOT 25, or if the contents of any message OID field cannot be parsed to fit the SYNTAX defined for the referenced object, then the PRS shall return an Error Status of 'badValue (3)' to the PRG and no further processing takes place.
b) The PRS shall check for a matching entry in the priorityRequestTable. A matching entry requires a priorityRequestStatusInPRS of other than 'idleNotQueued' and all of the following to have the same values as in the SET prgPriorityClear message:
   i) priorityRequestID,
   ii) priorityRequestVehicleID,
   iii) priorityRequestVehicleClassType,
   iv) priorityRequestVehicleClassLevel, and
   v) priorityRequestServiceStrategyNumber
c) If the PRS is unable to find a matching entry in the priorityRequestTable, the PRS shall return an Error Status of 'noSuchName (2)' to the PRG and no further processing takes place.
d) If priorityRequestStatusInPRS in the appropriate priorityRequestTableEntry is not 'closedCanceled', 'reserviceError', 'closedTimeToLiveError', 'closedFlash', 'closedTimerError', 'closedStrategyError', or 'closedCompleted', the PRS shall return an Error Status of 'genError' to the PRG and no further processing takes place.
e) If the above checks pass, the PRS shall set all information in the appropriate priorityRequestTableEntry to its default value state.
f) The PRS shall set priorityRequestStatusInPRS in the appropriate PriorityRequestTableEntry to 'idleNotValid'.
g) The PRS shall return an Error Status of 'noError' to the PRG.

4.2.3.5 Exchange Priority Request Status

4.2.3.5.1 Standardized Dialog

The standardized dialog for a PRS to receive a priority request status from a PRG shall be as follows:

a) The PRG shall SET prgPriorityStatusControl on the PRS with the desired values. This shall cause the PRS to perform a priority request status check (See Section 4.2.3.5.2).
b) If the PRS sends a GetResponse with 'NoError', the PRG shall GET prgPriorityStatusBuffer from the PRS.

c) If the prgPriorityStatusControl has invalid data (i.e., it has not been updated yet), the PRS shall send a GetResponse with an Error Status of ‘badValue (3)’ to the PRG.
d) If prgPriorityStatusControl has valid data, the PRS shall utilize values currently in prgPriorityStatusControl to define the data to be returned in the GetResponse.

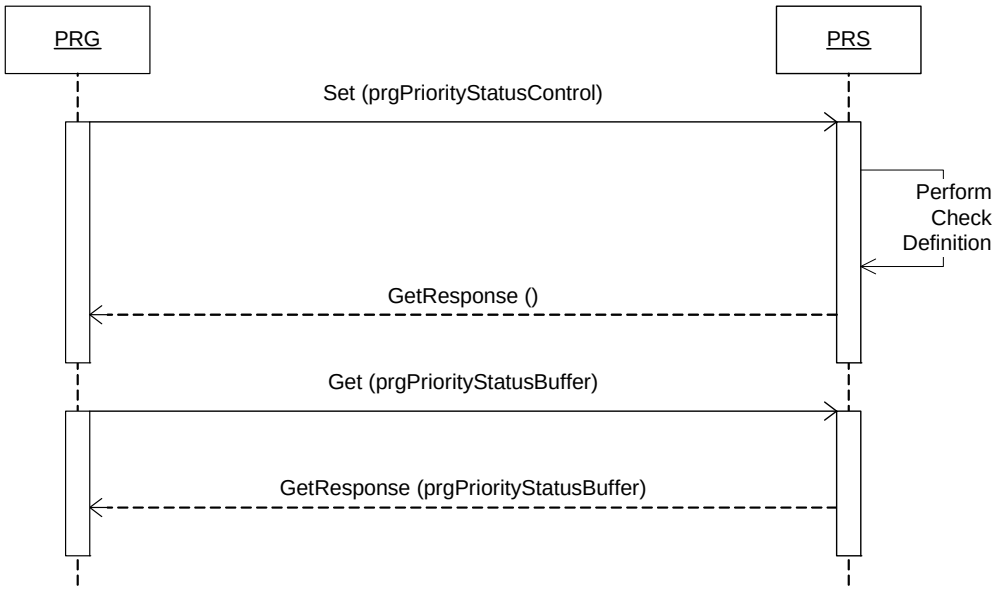

Figure 19 Exchange Priority Request Status

4.2.3.5.2 Check Definition

Upon receiving a SET prgPriorityStatusControl message, the PRS shall perform several checks before responding with an acknowledgement.

a) If the SET length is NOT 21, or if the contents of any message OID field cannot be parsed to fit the SYNTAX defined for the referenced object, then the PRS shall return an Error Status of ‘badValue (3)’ to the PRG and no further processing takes place.
b) The PRS shall check for a matching entry in the priorityRequestTable. A matching entry requires a priorityRequestStatusInPRS of other than ‘idleNotQueued’ and all of the following to have the same values as in the SET prgPriorityStatusControl message:
    i) priorityRequestID,
    ii) priorityRequestVehicleID,
    iii) priorityRequestVehicleClassType,
    iv) priorityRequestVehicleClassLevel, and
    v) priorityRequestServiceStrategyNumber
c) If the PRS is unable to find a matching entry in the priorityRequestTable, the PRS shall return an Error Status of ‘noSuchName (2)’ to the PRG and no further processing takes place.
d) If the above checks pass, the PRS shall set all information from the appropriate priorityRequestTableEntry to the prgPriorityStatusBuffer.
e) The PRS shall return an Error Status of ‘noError’ to the PRG.

4.2.3.6 Exchange Priority Request—NTCIP 1211 v01

4.2.3.6.1 Standardized Dialog

The standardized dialog for a PRS to receive a NTCIP 1211 v01-conformant priority request from a PRG shall be as follows:

a) The PRG shall SET prgPriorityRequest with the desired values. This shall cause the PRS to perform a priority request check definition (See Section 4.2.3.6.2).
b) If the PRS sends a GetResponse with 'noError', the priority request has been accepted and the PRS is waiting for the CO to complete its service processing. The PRG may then exit the process.

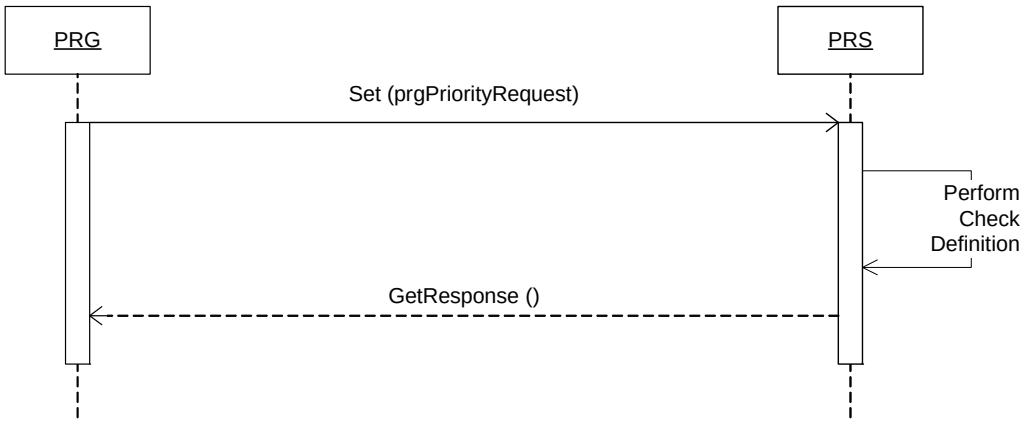

Figure 20 Exchange Priority Request

4.2.3.6.2 Check Definition

Upon receiving a SET prgPriorityRequest message from a PRG, the PRS shall perform several checks and processes before responding with an acknowledgement.

a) If the SET length is NOT 25, or if the contents of any message OID field cannot be parsed to fit the SYNTAX defined for the referenced object, then the PRS shall return an Error Status of ‘badValue (3)’ to the PRG and no further processing takes place.
b) If the priorityRequestTable does not have at least one row with a priorityRequestStatusInPRS of ‘idleNotValid’, the PRS shall return an Error Status of ‘noSuchName (2)’ to the PRG and no further processing takes place. (This error-status can be considered as a buffer full error.)
c) If the above checks are completed without error, the PRS shall store the contents of the prgPriorityRequest message in the appropriate priorityRequestTableEntry.
d) The PRS shall set priorityRequestTimeOfMessage in the appropriate priorityRequestTableEntry.
e) The PRS shall set priorityRequestTimeToLive in the appropriate priorityRequestTableEntry.
f) The PRS shall set priorityRequestTimeOfServiceDesiredInPRS in the appropriate priorityRequestTableEntry.
g) The PRS shall set priorityRequestTimeOfEstimatedDepartureInPRS in the appropriate priorityRequestTableEntry.
h) The PRS shall check if the reservice time has been exceeded by comparing the appropriate priorityRequestReserviceClass”X”Time against the priorityRequestReserviceTimer. If the priorityRequestReserviceTimer is not greater than the appropriate priorityRequestReserviceClass”X”Time, the PRS shall set priorityRequestStatusInPRS for the appropriate priorityRequestTableEntry to ‘reserviceError (9)’. Otherwise, the PRS shall set priorityRequestStatusInPRS for the appropriate PriorityRequestTableEntry to ‘readyQueued (2)’.
i) If prsBusy is ‘True’ and the status of a lower Class Type request is ‘activeProcessing’ or

'activeAdjustNotNeeded', the PRS shall set priorityRequestStatusInPRS of the lower Class Type to 'activeOverride'.

j) The PRS shall return an Error Status of 'noError' to the PRG.

4.2.3.7 Exchange Priority Update—NTCIP 1211 v01

4.2.3.7.1 Standardized Dialog

The standardized dialog for a PRS to receive a NTCIP 1211 v01-conformant priority request update from a PRG shall be as follows:

a) The PRG shall SET prgPriorityUpdate with the desired values. This shall cause the PRS to perform a priority request update check definition (See Section 4.2.3.7.2).
b) If the PRS sends a GetResponse with 'noError', the priority request update has been accepted and the PRS is waiting for the CO to complete its service processing. The PRG may then exit the process.

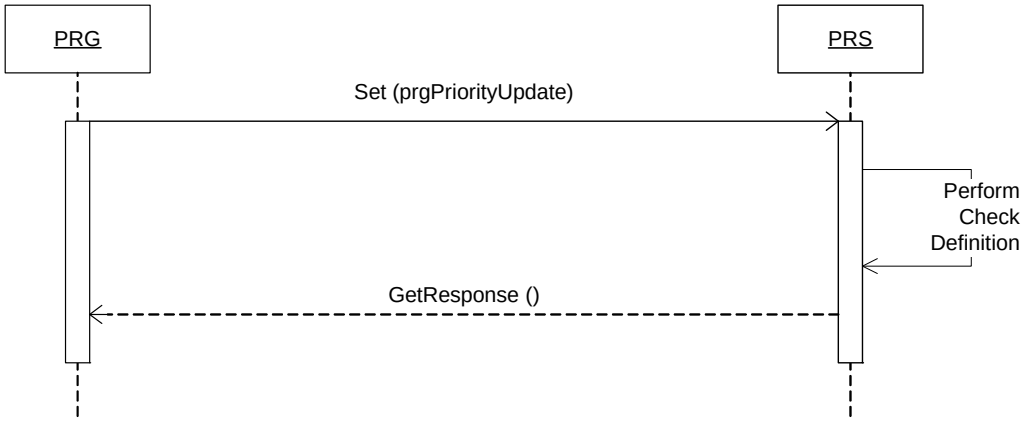

Figure 21 Exchange Priority Request Update

4.2.3.7.2 Check Definition

Upon receiving a SET prgPriorityUpdate message from a PRG, the PRS shall perform several checks before responding with an acknowledgement.

a) If the SET length is NOT 25, or if the contents of any message OID field cannot be parsed to fit the SYNTAX defined for the referenced object, then the PRS shall return an Error Status of 'badValue (3)' to the PRG and no further processing takes place.
b) The PRS shall check for a matching entry in the priorityRequestTable. A matching entry requires a priorityRequestStatusInPRS of other than 'idleNotQueued' and all of the following to have the same values as in the SET prgPriorityUpdate message:
    i) priorityRequestID,
    ii) priorityRequestVehicleID,
    iii) priorityRequestVehicleClassType,
    iv) priorityRequestVehicleClassLevel, and
    v) priorityRequestServiceStrategyNumber
c) If the PRS is unable to find a matching entry in the priorityRequestTable, the PRS shall return an Error Status of 'noSuchName (2)' to the PRG and no further processing takes place.
d) If the above checks are completed without error, the PRS shall set priorityRequestTimeOfServiceDesired and priorityRequestTimeOfEstimatedDeparture in the appropriate PriorityRequestTableEntry to the values received.

e) The PRS shall set priorityRequestTimeOfServiceDesiredInPRS and priorityRequestTimeOfEstimatedDepartureInPRS in the appropriate priorityRequestTableEntry.
f) The PRS shall return an Error Status of ‘noError’ to the PRG.

4.2.4 Interface—PRS to CO

Section 4.2.4 provides the dialogs that may occur between the PRS and the CO. This section includes only those dialogs that have special semantics or impose special restrictions on the operations are allowed.

For this communications interface, which entity acts as the manager and which entity acts as the agent, is implementation specific. Also, if the PRS is implemented as an integral part of the traffic signal controller, the method by which the PRS and CO exchange information is implementation specific.

If the PRS is implemented as a separate entity and is located within the traffic control assembly, the assumption is that the CO acts as the manager and the PRS acts as the agent (i.e. intra-cabinet communications). In this case, the CO shall send the PRS a ‘service request’ message, as an SNMP GetRequest consisting of a single variable binding as defined in Section 4.2.4.1.1, to retrieve information about priority service requests. The CO shall also send the PRS the same ‘service request’ message, but as an SNMP SetRequest to provide status about priority service requests.

If the PRS is implemented as a separate entity but is NOT located within the traffic control assembly, the assumption is that the PRS acts as the manager and the CO acts as the agent (i.e. systems communications). In this case, the PRS shall send the CO a ‘service request’ message, as an SNMP SetRequest consisting of a single variable binding as defined in Section 4.2.4.1.2, to retrieve information about priority service requests. The PRS shall also send the CO the same "service request" message, but as an SNMP GetRequest to provide status about priority service requests.

4.2.4.1 Exchange Service Request

The standardized dialog for exchanging a service request between a CO and a PRS depends on which entity is acting as the manager and which entity is acting as the agent.

4.2.4.1.1 Exchange Service Request—CO is Manager

The following assumes that the CO is the manager and the PRS is the agent.

4.2.4.1.1.1 Standardized Dialog

As illustrated in Figure 22, the standardized dialog for a CO to receive a priority service request from a PRS shall be as follows:

a) The CO shall always act upon priorityRequestEntryNumber ‘1’. This does not preclude a CO from acting upon or considering additional service requests.
b) The CO shall GET prsServiceRequest from the PRS.
c) The PRS shall send a GetResponse with the values of the objects contained within the PRS:

   i) priorityRequestServiceStrategyNumber.x
   ii) priorityRequestTimeOfServiceDesiredInPRS.x
   iii) priorityRequestTimeOfEstimatedDepartureInPRS.x
   iv) priorityRequestStatusInPRS.x
   v) prsBusy

   Note: All ten rows are always transferred and the transfer includes prsBusy.

d) Upon receiving the prsServiceRequest from the PRS, if prsBusy is ‘True’, the CO shall ignore the contents of prsServiceRequest and send the GET prsServiceRequest to the PRS again.
e) Upon receiving the prsServiceRequest from the PRS, if prsBusy is ‘False’, the CO shall perform a get service request check (See Section 4.2.4.1.1.2).
f) After the CO completes its get service request check, the CO shall perform its service processing (See Section 4.2.4.1.3).
g) The CO shall SET prsServiceRequest to update the corresponding status in the PRS with the values:

   i) priorityRequestServiceStrategyNumber.x
   ii) priorityStrategyRequestedTimeOfServiceDesired.x
   iii) priorityStrategyRequestedTimeOfEstimatedDeparture.x
   iv) priorityStrategyRequestStatusInCO.x
   v) coBusy

   Note: All ten rows are always transferred and the transfer includes coBusy.

   Note: The CO may send a SET prsServiceRequest before the CO completes the above checks because the communications channel might impose timing constraints on how often a message is sent rather than when a message is ready to be sent. In this event, the coBusy is set to ‘False’.

h) Upon receiving the SET prsServiceRequest from the CO, if coBusy is ‘True’, the PRS shall ignore the contents of the SET prsServiceRequest.
i) Upon receiving the SET prsServiceRequest from the CO, if coBusy is ‘False’, the PRS shall process the SetRequest and store the values in the objects within it.

   priorityRequestServiceStrategyNumber.x
   priorityStrategyRequestedTimeOfServiceDesired.x > priorityRequestTimeOfServiceDesiredInPRS.x
   priorityStrategyRequestedTimeOfEstimatedDeparture.x > priorityRequestTimeOfEstimatedDepartureInPRS.x
   priorityStrategyRequestStatusInCO.x > priorityRequestStatusInPRS.x
   prsBusy

j) The PRS shall send a GetResponse to the CO with value field set to the values it just received.
k) When the CO receives the GetResponse from the PRS, the values received are discarded.
l) The PRS shall set prsBusy to ‘True’ and perform its prioritization processing (See Section 4.2.4.1.4).
m) Upon completing its prioritization process, the PRS shall set prsBusy to ‘False’ and then send SET prsServiceRequest to the CO and repeat the process over again.

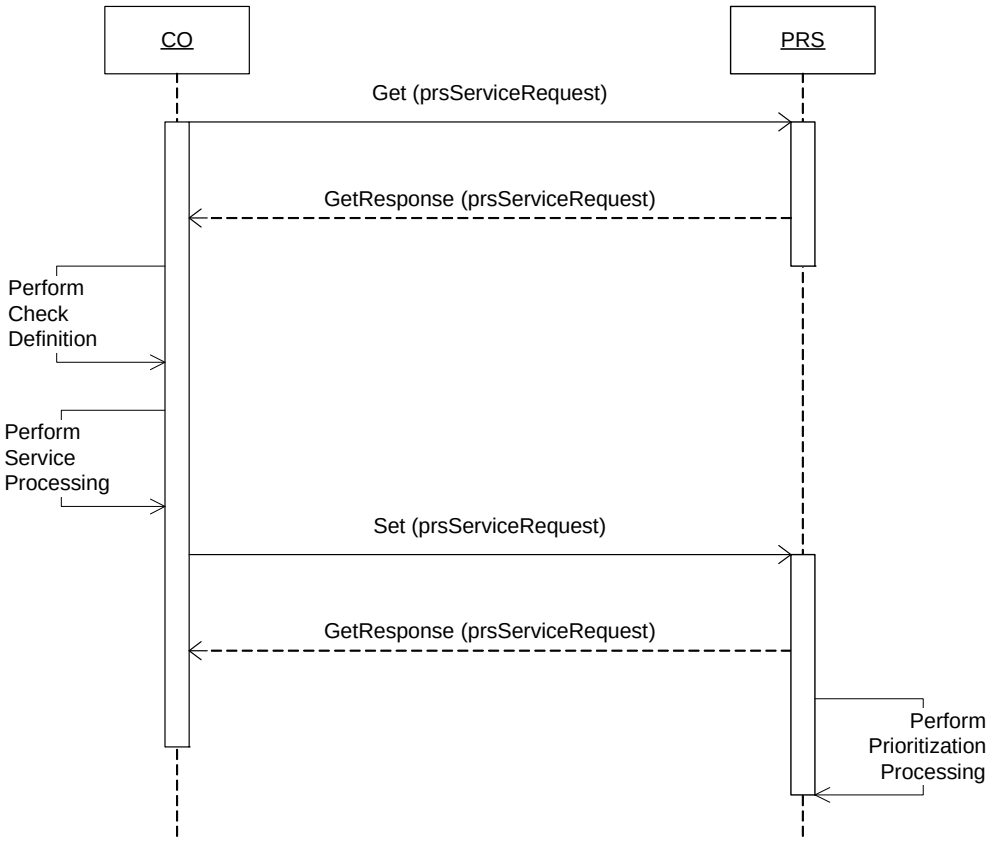

Figure 22 Exchange Service Request (CO is Manager)

4.2.4.1.1.2 Check Definition

Upon receiving a GetResponse (prsServiceRequest) from the PRS, the CO shall perform several checks before processing the service request.

a) The CO shall set coBusy to 'True'.
b) The CO shall store the equivalent objects from the PRS within the CO:

priorityRequestServiceStrategyNumber.x > priorityStrategyRequested.x
priorityRequestTimeOfServiceDesiredInPRS.x > priorityStrategyRequestedTimeOfServiceDesired.x
priorityRequestTimeOfEstimatedDepartureInPRS.x > priorityStrategyRequestedTimeOfEstimatedDeparture.x
priorityRequestStatusInPRS.x > priorityStrategyRequestStatusInCO.x

c) If the priorityStrategyRequestStatusInCO is 'readyQueued', and if the values in priorityStrategyRequestedTimeOfServiceDesired and priorityStrategyRequestedTimeOfEstimatedDeparture cannot meet some criteria defined by the CO, the CO shall SET priorityStrategyRequestStatusInCO to 'closedTimerError'.
d) If the priorityStrategyRequestStatusInCO is 'readyQueued', and if the coordination values associated with the priorityStrategyRequested cannot meet some criteria defined by the CO, the CO shall SET priorityStrategyRequestStatusInCO to 'closedStrategyError'.

e) If the priorityStrategyRequestStatusInCO is ‘readyQueued’, and if the CO determines that a request can be met by current timing, the CO shall SET priorityStrategyRequestStatusInCO to ‘activeAdjustNotNeeded’.
f) If the priorityStrategyRequestStatusInCO is ‘readyQueued’, and if the CO determines that the controller unit is operating in flash, the CO shall SET priorityStrategyRequestStatusInCO to ‘closedFlash’.
g) If the request passes these checks, the CO shall SET priorityStrategyRequestStatusInCO to ‘activeProcessing’.

4.2.4.1.2 Exchange Service Request—PRS is Manager

The following assumes that the PRS is the manager and the CO is the agent.

As illustrated in Figure 23, the standardized dialog for a CO to receive a priority service request from a PRS shall be as follows:

a) The CO shall always act upon priorityRequestedEntryNumber ‘1’. This does not preclude a CO from acting upon or considering other requests.
b) The PRS shall SET prsServiceRequest to the CO with the values of the objects contained within the PRS:

   priorityRequestServiceStrategyNumber.x
   priorityRequestTimeOfServiceDesiredInPRS.x
   priorityRequestTimeOfEstimatedDepartureInPRS.x
   priorityRequestStatusInPRS.x
   prsBusy

   Note: All ten rows are always transferred and the transfer includes prsBusy.

c) Upon receiving the SET prsServiceRequest from the PRS, the CO shall set coBusy to ‘True’ and send a GetResponse to the PRS with value field set to the values it just received in the message.
d) The CO shall process the SetRequest and store the equivalent objects from the PRS within the CO:

   priorityRequestServiceStrategyNumber.x > priorityStrategyRequested.x
   priorityRequestTimeOfServiceDesiredInPRS.x > priorityStrategyRequestedTimeOfServiceDesired.x
   priorityRequestTimeOfEstimatedDepartureInPRS.x >
   priorityStrategyRequestedTimeOfEstimatedDeparture.x
   priorityRequestStatusInPRS.x > priorityStrategyRequestStatusInCO.x

   Note: All ten rows are always transferred and the transfer includes prsBusy.

e) The CO shall perform its service processing (See Section 4.2.4.1.3).
f) The PRS shall then send a Get prsServiceRequest to the CO with value field set to ‘Null’.
g) Upon receiving the Get prsServiceRequest, the CO shall send a GetResponse with the value field set to the values of the objects contained within the CO:

   priorityRequestServiceStrategyNumber.x
   priorityStrategyRequestedTimeOfServiceDesired.x
   priorityStrategyRequestedTimeOfEstimatedDeparture.x
   priorityStrategyRequestStatusInCO.x
   coBusy

h) Upon receiving the GetResponse from the CO, if coBusy is ‘True’, the PRS shall continue to send a Get prsServiceRequest to the CO until the coBusy is ‘False’.
i) Upon receiving the GetResponse from the CO, if coBusy is ‘False’, the PRS shall set prsBusy to ‘True’ and perform its prioritization processing (See Section 4.2.4.1.4).

j) Upon completing its prioritization process, the PRS shall set prsBusy to ‘False’ and then SET prsServiceRequest to the CO and repeat the process over again.

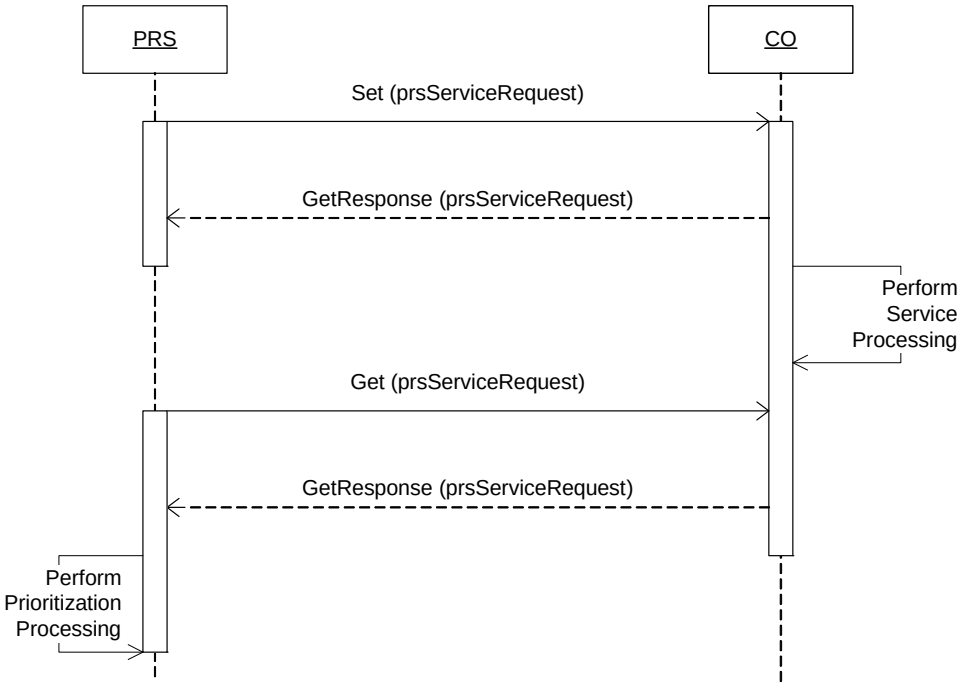

Figure 23 Exchange Service Request (PRS is Manager)

4.2.4.1.3 Service Processing

The CO shall be responsible for implementing the service request as a strategy defined by various parameters in the priorityStrategyTable and the current running coordination pattern (see NTCIP 1202:2005). The exact nature of how a CO responds is dependent upon its ability to respond to one or more than one service request at a time. The CO shall implement the strategy(ies) as defined by the following:

a) The SCPs coordPatternStatus variable (or priorityStrategyDefaultCoordPattern if not coordinated) shall indicate that the following shall apply:
   i) priorityStrategyMaximumReductionTime
   ii) priorityStrategyMaximumExtensionTime
b) The priorityStrategyRequested shall indicate that the following shall apply:
   i) priorityStrategyPhaseOmits
   ii) priorityStrategyPedOmits
c) The CO shall then consider:
   i) priorityStrategyRequestedTimeOfServiceDesired
   ii) priorityStrategyRequestedTimeOfEstimatedDeparture
d) Based on these parameters, the CO shall adjust the "split" timing of the phases to give priority to the fleet vehicle while maintaining coordination. The controller unit has 'maintained coordination' following completion of a priority service request if upon resumption of normal operation the split timings occur at the appropriate points in the cycle. The intent is that the controller cycles back to or remains in the coordinated phase(s) such that they are green at the zero point in the cycle following completion of the priority service request and then time the allotted split time from that point. Skipping the coordinated phase(s) in any one cycle shall not be permitted.

e) A service request is complete when [priorityRequestTimeOfServiceDesired (TSD) and priorityRequestTimeOfEstimatedDeparture (TED) have passed OR priority phase(s) have timed the maximum split extension OR priority has been cancelled] AND non-priority split reduction(s) equal priority split extension(s). Once the service request is complete, priority omits (phase and pedestrian) shall be removed.
f) Once the CO has completed the service request, the CO shall set priorityStrategyRequestStatusInCO to ‘closedCompleted’ and reset priorityRequestReserviceTimer to ‘0’.
g) If priorityStrategyRequestStatusInCO is ‘activeOverride’, the CO may be able to complete the current request without affecting the need to cancel the CO. If it can, the CO would return priorityStrategyRequestStatusInCO set to ‘activeNotOverridden’. If not, it completes the active strategy as though the TSD and TED have passed and then sets priorityStrategyRequestStatusInCO to ‘readyOverridden’.
h) If priorityStrategyRequestStatusInCO is ‘activeCancel’, the CO shall complete the active strategy as though the TSD and TED have passed. When the CO completes the strategy, priorityStrategyRequestStatusInCO is set to ‘closedCanceled’.
i) Upon completion of the service processing, the CO shall set coBusy to ‘True’.

4.2.4.1.4 Prioritization Processing

When prsBusy is ‘True’, the following prioritization processing shall take place.

The PRS shall scan all priorityRequestTable entries.

a) If the priorityRequestStatusInPRS is ‘readyX’, or ‘closedX’, AND if the value priorityRequestTimeToLive is greater than or equal to the global time (priorityRequestTimeToLive >= GLO.globalTime), then the PRS shall set all information in the priorityRequestTableEntry to its default value state and set priorityRequestStatusInPRS in the priorityRequestTableEntry to ‘idleNotValid’.
b) If the value of priorityRequestTimeOfServiceDesiredInPRS is more than priorityRequestTimeToLive then priorityRequestStatusInPRS is set to ‘closedTimeToLiveError’.
c) If none of the entries in the priorityRequestTable has a priorityRequestStatusInPRS of ‘activeX’, then:
    i) Rows where priorityRequestStatusInPRS is ‘readyQueued’ shall be prioritized according to highest priorityRequestVehicleClassType, highest priorityRequestVehicleClassLevel, and then soonest priorityRequestTimeOfServiceDesiredInPRS. The row with the highest value of each becomes priorityRequestEntryNumber "1". The row with the next highest of each shall become priorityRequestEntryNumber "2", etc.
    ii) These are followed by any row where priorityRequestStatusInPRS is ‘readyOverriden’.
    iii) These are followed by any row where priorityRequestStatusInPRS is ‘closedX’.
    iv) These are followed by any row where priorityRequestStatusInPRS is ‘idleNotValid’.
d) The PRS shall set prsBusy to ‘False’.

The priorityRequestEntryNumber of these rows follows in order and does not necessarily reflect a priority.

4.2.4.2 Exchange Service Request Status

The standardized dialog for exchanging a service request status between a CO and a PRS depends on which entity is acting as the manager and which entity is acting as the agent.

4.2.4.2.1 Exchange Service Request Status (CO is Manager)

The following assumes that the CO is the manager and the PRS is the agent.

The standardized dialog for a PRS to get the status of any request for priority service from a CO shall be as follows:

a) The CO shall SET prsServiceRequest on the PRS.
b) Upon receiving the SET prsServiceRequest from the CO, if coBusy is ‘True’, the PRS shall know the CO is still processing.
c) Upon receiving the SET prsServiceRequest from the CO, if coBusy is ‘False’, the PRS shall process

the SetRequest and store the values in the objects within it.
d) The PRS shall send a GetResponse to the CO with value field set to the values it just received.

4.2.4.2.2 Exchange Service Request Status (PRS is Manager)

The following assumes that the PRS is the manager and the CO is the agent.

The standardized dialog for a PRS to get the status of any request for priority service from a CO shall be as follows:

a) The PRS shall GET prsServiceRequest from the CO.
b) Upon receiving the Get prsServiceRequest, the CO shall send a GetResponse with the value field set to the values of the objects contained within the CO:

   i) priorityRequestServiceStrategyNumber.x
   ii) priorityStrategyRequestedTimeOfServiceDesired.x
   iii) priorityStrategyRequestedTimeOfEstimatedDeparture.x
   iv) priorityStrategyRequestStatusInCO.x
   v) coBusy

   Note: All ten rows are always transferred and the transfer includes coBusy.

4.2.5 prsBusy State

The prsBusy state is used to synchronize the processing of status changes in the PRS. If prsBusy is ‘True’, it indicates that the PRS is “busy” initializing, or processing a priority request. If prsBusy is ‘False’, it indicates that the PRS is not busy, and thus the information in the priorityRequestTable is valid.

The following rules shall apply to the PRS:

a) Upon PRS initialization, the PRS shall set prsBusy to ‘True’.
b) Upon completing its initialization, the PRS shall set prsBusy to ‘False’.
c) If the CO is the manager, when the CO sends a SetRequest (prsServiceRequest) to the PRS, the PRS shall set prsBusy to ‘True’.
d) If the PRS is the manager, when the PRS sends a GetRequest (prsServiceRequest) to the CO and if the coBusy is ‘False’ in the GetResponse (prsServiceRequest) from the CO, the PRS shall set prsBusy to ‘True’.
e) When the PRS completes its prioritization processes (basically any change in priorityRequestStatusInPRS), the PRS shall set prsBusy to ‘False’ and pass the state to the CO so that the CO knows that the PRS has completed its prioritization process of requests in its queue. The CO can act upon the information it received when prsBusy is ‘False’.

4.2.6 coBusy State

The coBusy state is used to synchronize the processing of status changes in the CO. If coBusy is ‘True’, it indicates that the CO is “busy” initializing, or processing a service request. If coBusy is ‘False’, it indicates that the CO is not busy, and thus the information in the priorityStrategyRequestedTable is valid.

The following rules shall apply to the CO:

a) Upon CO initialization, the CO shall set coBusy to ‘True’.
b) Upon completing its initialization, the CO shall set coBusy to ‘False’.
c) If the CO is the manager, when the CO sends a GetRequest (prsServiceRequest) to the PRS and if prsBusy is ‘False’ in the GetResponse (prsServiceRequest), the CO shall set coBusy to ‘True’.
d) If the PRS is the manager, when the PRS sends a SetRequest (prsServiceRequest) to the CO, the CO shall set coBusy to ‘True’.

e) When the CO completes its service processing (basically any change in priorityRequestStatusInCO), the CO shall set coBusy to ‘False’ and pass the state to the PRS so that the PRS knows that the CO has completed its servicing the requests in its queue. The PRS can act upon the information it received when coBusy to ‘False’.

4.3 State-Transition Diagrams

State Transition diagrams are included for those objects that have states or manage states. The State Transition Diagrams include state-transition tables (listing of the possible state transitions), legitimate transitions, and any illegitimate transitions.

> “State-transition diagrams describe all of the states that an object can have, the events under which an object changes state (transitions), the conditions that must be fulfilled before the transition will occur (guards), and the activities undertaken during the life of an object (actions).” (Reference: State-Transition Diagrams: Testing UML Models, Part 4 by Lee Copeland)

The following subsections define the states for various object classes that may be supported by the PRS and the CO.

4.3.1 Request Status Definition

Figure 24 is a UML State Transition Diagram for priorityRequestStatusInPRS and priorityStrategyRequestStatusInCO. See Sections 5.1.1.1.9 and 5.2.1.2.5 for more detail about what states an entity can set and what states an entity just processes.

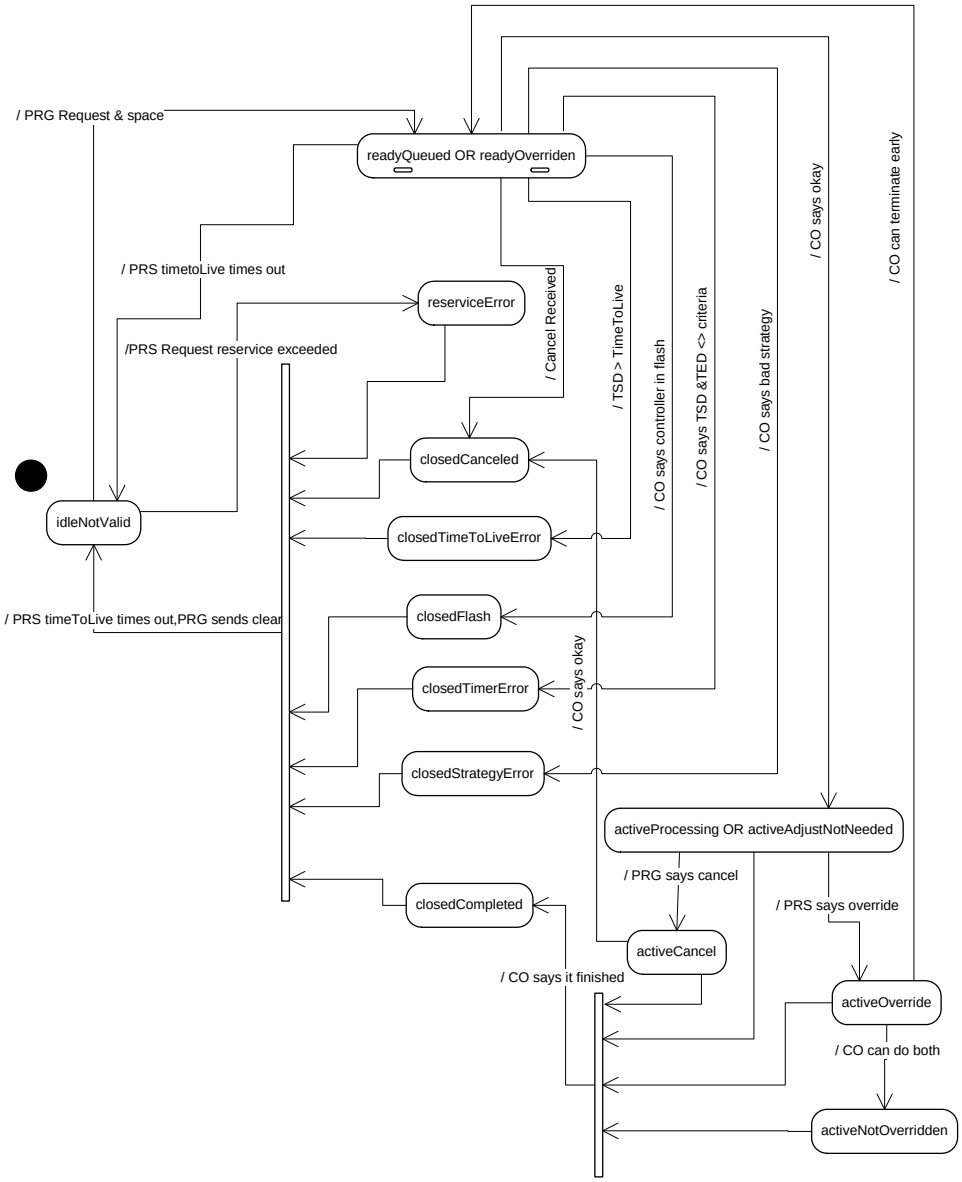

Figure 24 Status Transition Diagram

Section 5
Management Information Base (MIB) [Normative]

5.1 Priority Request Server MIB

This section defines the information requirements of the PRS portion of an SCP system. The objects are defined using the OBJECT-TYPE macro as specified in RFC 1212 and NTCIP 8004 v02. The text provided from Sections 5.1 through the end of Clause 5.2.7 (except the clause headings) constitutes the standard NTCIP1211-2008 Management Information Base (MIB) for a PRS.

The sections below present the objects in lexicographical order of their OBJECT IDENTIFIERS, which correspond to their physical location within the global naming tree. Most of the objects defined in NTCIP 1211 v02 reside under the "scp" node of the global naming tree. To aid in object management, the SCP node has been subdivided into logical categories; each defined by a node under the SCP node. The individual objects are then located under the appropriate node.

Conformance requirements for any object is determined by the use of the Requirements Traceability Matrix (RTM) in Annex A. To support any defined Requirement, an implementation shall support all objects to which the Requirement traces in the RTM. The value of the STATUS field for every object in the MIB is "mandatory", and indicates that it is mandatory if any associated Requirement is selected.

The full standardized range of all objects defined within NTCIP 1211 v02 shall be supported, except as otherwise noted. This MIB is managed by the NTCIP SCP Working Group and proprietary features should be defined through vendor-specific nodes in vendor-specific extensions to this MIB. All values not explicitly defined (e.g., enumerated values not listed, bits not defined, etc.) are reserved for future use by the SCP Working Group.

A computer readable format of this MIB is available from NEMA (ntcip@nema.org). The MIB has been verified using SMICng Version 2.2.07 (Book).

5.1.1 Object Definitions - PRS

```
PRS-MIB1 DEFINITIONS ::= BEGIN

-- This group of objects represents the data elements that would reside in
-- a Priority Request Server.

IMPORTS

      EntryStatus, OwnerString, OerString,
      Byte, UByte, Short, Ushort, Long, Ulong
      FROM NEMA-SMI

      BITMAP16, scp
      FROM NEMA-SMI2

      OBJECT-TYPE
      FROM RFC-1212

      priorityRequestServer ::= {scp 1}

      TrueFalse ::= INTEGER (0 | 255)
```



```
      FALSE ::= 0
      TRUE ::= NOT FALSE
```

5.1.1.1 Priority Request Table

```
priorityRequestTable OBJECT-TYPE
      SYNTAX      SEQUENCE OF PriorityRequestEntry
      ACCESS      not-accessible
      STATUS      mandatory
      DESCRIPTION
            "A static table containing the parameters associated with each
            signal control priority request. The number of rows in this table
            is equal to 10."
::= { priorityRequestServer 1 }

priorityRequestTableEntry OBJECT-TYPE
      SYNTAX      PriorityRequestTableEntry
      ACCESS      not-accessible
      STATUS      mandatory
      DESCRIPTION
            "This object defines the parameters that are associated with
            priority requests."
      INDEX       { priorityRequestEntryNumber }
::= { priorityRequestTable 1 }

PriorityRequestTableEntry ::= SEQUENCE {
      priorityRequestEntryNumber                INTEGER,
      priorityRequestID                         INTEGER,
      priorityRequestVehicleID                  OCTET STRING (SIZE 17),
      priorityRequestVehicleClassType           INTEGER,
      priorityRequestVehicleClassLevel          INTEGER,
      priorityRequestServiceStrategyNumber      INTEGER,
      priorityRequestTimeOfServiceDesired       INTEGER,
      priorityRequestTimeOfEstimatedDeparture   INTEGER,
      priorityRequestStatusInPRS                INTEGER,
      priorityRequestTimeOfMessage              INTEGER,
      priorityRequestTimeToLive                 INTEGER,
      priorityRequestTimeOfServiceDesiredInPRS  INTEGER,
      priorityRequestTimeOfEstimatedDepartureInPRS    INTEGER,
      priorityRequestTimeOfRequest              INTEGER
      }
```

5.1.1.1.1 Priority Request Entry Number

```
priorityRequestEntryNumber OBJECT-TYPE
      SYNTAX      INTEGER (1..10)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object represents the row number in the priority request
            table. It is used by a management application for identification
            purposes."
::= { priorityRequestTableEntry 1 }
```

5.1.1.1.2 Priority Request ID

```
priorityRequestID OBJECT-TYPE
```

```
      SYNTAX      INTEGER (1..255)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object is the 'PRG requested' priority request identifier.
            It is assigned by the priority request generator so that further
            information related to a priority request can be identified. It
            shall be unique for this intersection from a vehicle ID of
            vehicle type."
      DEFVAL { 1 }
::= { priorityRequestTableEntry 2 }
```

5.1.1.1.3 Priority Request Vehicle ID

```
priorityRequestVehicleID OBJECT-TYPE
      SYNTAX      OCTET STRING (SIZE 17)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object is the 'PRG requested' globally unique identifier of
            the entity requesting priority. For fleet vehicles, this is
            assumed to be the Vehicle Identification Number (VIN). For
            management centers, the value is not defined but shall still be
            unique to differentiate the source of the priority request."
      DEFVAL { "INVALID-VEH-ID-##" }            -- considered invalid
::= { priorityRequestTableEntry 3 }
```

5.1.1.1.4 Priority Request Vehicle Class Type

```
priorityRequestVehicleClassType OBJECT-TYPE
      SYNTAX      INTEGER (1..10)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object is the 'PRG requested' class type (relative priority
            of a request). The order of precedence is by class type with
            1                                         highest;
            …
            10                                        lowest
            A request with a higher class type shall override a lower class
            type."
      DEFVAL { 10 }
::= { priorityRequestTableEntry 4 }
```

5.1.1.1.5 Priority Request Vehicle Class Level

```
priorityRequestVehicleClassLevel OBJECT-TYPE
      SYNTAX      INTEGER (1..10)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object is the 'PRG requested' class level (relative
            priority of a request within each class of request). The order of
            precedence is by class type and then class level.
            1                                         highest;
            …
            10                                        lowest
```



```
            A request with a higher class level does NOT override a lower
            class level."
      DEFVAL { 10 }
::= { priorityRequestTableEntry 5 }
```

5.1.1.1.6 Priority Request Service Strategy Number

```
priorityRequestServiceStrategyNumber OBJECT-TYPE
      SYNTAX      INTEGER (0..255)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object is the 'PRG requested' strategy (1..255) or 0
            indicating the PRS has reset the value to none.

            The PRS only checks for non-zero value in a request. The CO
            determines whether a specific value represents a valid strategy.
            A PRG shall never set this to a value of zero (0). If it does, a
            PRS shall return a badValue error."
      DEFVAL { 0 }
::= { priorityRequestTableEntry 6 }
```

5.1.1.1.7 Priority Request Time of Service Desired

```
priorityRequestTimeOfServiceDesired OBJECT-TYPE
      SYNTAX      INTEGER (1..65535)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object is the 'PRG requested' desired time in seconds to
            arrive at the intersection's stopping point relative to the
            receipt of the message. A near side stopping point is assumed to
            be sufficiently close to the intersection's stop bar that regular
            queues frequently back up across the stopping point. In this
            case, advance queue clearance prior to the arrival of fleet
            vehicle is normally required. For all practical purposes, arrival
            at the stopping point is the same as arrival at the stop bar.
            This is a relative time.

            Note that it is the responsibility of the PRG to take into
            account any communications delays between the PRG and the PRS."
      DEFVAL { 1 }
::= { priorityRequestTableEntry 7 }
```

5.1.1.1.8 Priority Request Time of Estimated Departure

```
priorityRequestTimeOfEstimatedDeparture OBJECT-TYPE
      SYNTAX      INTEGER (1..65535)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object is the 'PRG requested' estimated time in seconds of
            departure from the intersection's stopping point relative to the
            receipt of the message.
            This is a relative time.
```

```
            Note that it is the responsibility of the PRG to take into
            account any communications delays between the PRG and the PRS."
      DEFVAL { 1 }
::= { priorityRequestTableEntry 8 }
```

5.1.1.1.9 Priority Request Status

```
priorityRequestStatusInPRS OBJECT-TYPE
      SYNTAX      INTEGER { idleNotValid (1),
                  readyQueued (2),
                  readyOverridden (3),
                  activeProcessing (4),
                  activeCancel (5),
                  activeOverride (6),
                  activeNotOverridden (7),
                  closedCanceled (8),
                  ReserviceError (9),
                  closedTimeToLiveError (10),
                  closedTimerError (11),
                  closedStrategyError (12),
                  closedCompleted (13),
                  activeAdjustNotNeeded (14),
                  closedFlash (15) }
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object provides status information about requests in the
            PRG. It is also passed to the CO as control information.
                  idleNotValid      PRG determined that row does not contain
                                    valid data
                  readyQueued       PRG has validated the request but is
                                    waiting for the CO to activate
                  readyOverridden   CO has overridden the request
                  activeProcessing  CO is processing the requested strategy
                  activeCancel      PRS has asked that request be canceled
                  activeOverride    PRS has asked that request be overridden
                  activeNotOverridden     CO did not process the requested
                                    override
                  closedCanceled    CO has canceled the request
                  reserviceError    PRS determined that the request came too
                                    soon after a previous request
                  closedTimeToLiveError   PRG determined that TSD exceeds the
                                    time to live
                  closedTimerError  CO indicated that the requested times
                                    could not be met
                  closedStrategyError     CO indicated that the requested
                                    strategy was not valid
                  closedCompleted   CO has completed the requested strategy
                  activeAdjustNotNeeded   CO indicated that the request can
                                    be met by the current timing and no
                                    adjustment is necessary
                  closedFlash       CO indicated that the controller is in
                                    flash

            Upon receipt of a activeProcessing or a activeAdjustNotNeeded
            from the CO, the PRS may change the status to:
                  activeCancel      - upon receipt of a prgPriorityCancel
```



```
                  activeOverride    - upon receipt of a prgPriorityRequest
                                    with higher class

            Upon receipt of a closedTimerError, closedStrategyError,
            closedCanceled, closedCompleted, closedTimeToLiveError or
            closedFlash from the CO, the PRS may change the status to:
                  idleNotValid      – upon receipt of a prgPriorityClear or a
                                    timeout of TimeToLive

            Upon receipt of a activeNotOverridden from the CO, the PRS may
            change the status to:
                  activeCancel      - upon receipt of a prgPriorityCancel

            Upon receipt of a readyOverridden from the CO, the PRS may change
            the status to:
                  readyQueued       – upon completion of overriding request
                  closedCanceled    - upon receipt of a prgPriorityCancel
                  closedTimeToLiveError   – a timeout of TimeToLive

            When the status is idleNotValid, the PRS may change the status
            to:
                  readyQueued       – upon receipt of a valid
                                    prgPriorityRequest
                  reserviceError    - upon receipt of a prgPriorityRequest
                                    that came too soon after a previous
                                    request

            When the status is readyQueued, the PRS may change the status to:
                  closedCanceled    - upon receipt of a prgPriorityCancel
                  closedTimeToLiveError   – a timeout of TimeToLive

            Note: A change of status is predicated on the 'busy' signal being
            set."
      DEFVAL { idleNotValid }
::= { priorityRequestTableEntry 9 }
```

5.1.1.1.10 Priority Request Time of Message

```
priorityRequestTimeOfMessage OBJECT-TYPE
      SYNTAX      INTEGER --(0..4294967295)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object represents the initial time of receipt of the
            priority request. The value of GLO.globalTime is copied to this
            object when new priority request is first entered in the
            priorityRequestTable.
            Alternatively, this value may be computed or measured by the
            device depending on the specific system implementation as
            directed by the priorityRequestTimeOfRequest object below."
      DEFVAL { 0 }
::= { priorityRequestTableEntry 10 }
```

5.1.1.1.11 Priority Request Time to Live

```
priorityRequestTimeToLive OBJECT-TYPE
      SYNTAX      INTEGER -- (0..4294967295)
```

```
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object represents the initial time of receipt of the
            priority request priorityRequestTimeOfMessage plus
            priorityRequestTimeToLiveValue. When GLO.globalTime is equivalent
            to this value, a priority request that has a
            priorityRequestStatusInPRS of 'readyX' or 'closedX' shall have
            the priorityRequestStatusInPRS set to 'idleNotValid'."
::= { priorityRequestTableEntry 11 }
```

5.1.1.1.12 Priority Request Time of Service Desired (CO)

```
priorityRequestTimeOfServiceDesiredInPRS OBJECT-TYPE
      SYNTAX      INTEGER -- (0..4294967295)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object is the 'PRS requested' desired time (GLO.globalTime)
            to arrive at the intersection's stopping point. It is derived by
            adding priorityRequestTimeOfServiceDesired to
            priorityRequestTimeOfMessage."
::= { priorityRequestTableEntry 12 }
```

5.1.1.1.13 Priority Request Time of Estimated Departure (PRS)

```
priorityRequestTimeOfEstimatedDepartureInPRS    OBJECT-TYPE
      SYNTAX      INTEGER -- (0..4294967295)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object is the 'PRS requested' time (GLO.globalTime) of
            departure from the intersection's stopping point. It is derived
            by adding priorityRequestTimeOfEstimatedDeparture to
            priorityRequestTimeOfMessage."
::= { priorityRequestTableEntry 13 }
```

5.1.1.1.14 Priority Request Time of Request

```
priorityRequestTimeOfRequest OBJECT-TYPE
      SYNTAX      INTEGER --(0..4294967295)
      ACCESS      read-only
      STATUS      optional
      DESCRIPTION
            "This object represents the initial time within the PRG that the
            priority request was generated (transmitted) by the PRG. This
            value represents the number of seconds since midnight January 1,
            1970. The value is used to compensate for time differences
            between the clocks of the PRG and PRS and the variability of the
            communications delays between the PRG and the PRS where they are
            separate entities. Note that this object assumes that the PRS has
            prior knowledge of the time reference used by the PRG (e.g. GPS).
            If this object contains a zero, then the
            priorityRequestTimeOfMessage represents the time the message is
            received. However, if this contains a non-zero value, then it is
            used to compare the time in the PRS to the time the message was
```



```
            generated, adjusted for any time differences, and used to set the
            value of priorityRequestTimeOfMessage."
      DEFVAL { 0 }
::= { priorityRequestTableEntry 14 }
```

5.1.1.2 Priority Request Server Busy

```
prsBusy OBJECT-TYPE
      SYNTAX      TrueFalse
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object is used to indicate to a CO that the PRS is
            initializing or otherwise processing and, when TRUE, the
            information included with it in a prsServiceRequest response is
            not valid. It is used to synchronize exchanges between a PRS and
            CO.

            When the CO is acting as manager and the PRS is acting as an
            agent, prsBusy is set TRUE when the CO performs a SetRequest
            prsServiceRequest.

            When the PRS is acting as a manager and the CO is acting as the
            agent, prsBusy is set TRUE when the CO responds to a GetRequest
            prsServiceRequest from the PRS with coBusy set to FALSE.”
::= { priorityRequestServer 2 }
```

5.1.1.3 Priority Request Time to Live Value

```
priorityRequestTimeToLiveValue OBJECT-TYPE
      SYNTAX      INTEGER (0..65535)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object represents the value (expressed in seconds) that is
            added to the current value of GLO.globalTime and loaded into
            priorityRequestTimeToLive when a priority request is initially
            entered in the table."
      DEFVAL { 0 }
::= { priorityRequestServer 3 }
```

5.1.1.4 Priority Request Reservice Timer

```
priorityRequestReserviceTimer OBJECT-TYPE
      SYNTAX      Gauge (0..65535)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object represents a timer that is reset to zero at the
            completion of any strategy (priorityRequestStatusInPRS is
            initially set equal to 'closedCompleted'). It is incremented
            every second up to its maximum value of 65535 where it latches.

            This timer is compared to Priority Request Reservice Class X
            Times to determine whether an priority request that comes right
            after a previous will be honored."
::= { priorityRequestServer 4 }
```

5.1.1.5 Priority Request Reservice Class 1 Time

```
priorityRequestReserviceClass1Time OBJECT-TYPE
      SYNTAX      INTEGER (0..65535)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object represents the time in seconds between the end of
            ANY strategy and the start of any new Class Type 1 priority
            request. The value of the priorityRequestReserviceTimer shall be
            equal to or greater than this value for priority request to be
            honored."
      DEFVAL { 0 }
::= { priorityRequestServer 5 }
```

5.1.1.6 Priority Request Reservice Class 2 Time

```
priorityRequestReserviceClass2Time OBJECT-TYPE
      SYNTAX      INTEGER (0..65535)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object represents the time in seconds between the end of
            ANY strategy and the start of any new Class Type 2 priority
            request. The value of the priorityRequestReserviceTimer shall be
            equal to or greater than this value for priority request to be
            honored."
      DEFVAL { 0 }
::= { priorityRequestServer 6 }
```

5.1.1.7 Priority Request Reservice Class 3 Time

```
priorityRequestReserviceClass3Time OBJECT-TYPE
      SYNTAX      INTEGER (0..65535)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object represents the time in seconds between the end of
            ANY strategy and the start of any new Class Type 3 priority
            request. The value of the priorityRequestReserviceTimer shall be
            equal to or greater than this value for priority request to be
            honored."
      DEFVAL { 0 }
::= { priorityRequestServer 7 }
```

5.1.1.8 Priority Request Reservice Class 4 Time

```
priorityRequestReserviceClass4Time OBJECT-TYPE
      SYNTAX      INTEGER (0..65535)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object represents the time in seconds between the end of
            ANY strategy and the start of any new Class Type 4 priority
            request. The value of the priorityRequestReserviceTimer shall be
```



```
            equal to or greater than this value for priority request to be
            honored."
      DEFVAL { 0 }
::= { priorityRequestServer 8 }
```

5.1.1.9 Priority Request Reservice Class 5 Time

```
priorityRequestReserviceClass5Time OBJECT-TYPE
      SYNTAX      INTEGER (0..65535)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object represents the time in seconds between the end of
            ANY strategy and the start of any new Class Type 5 priority
            request. The value of the priorityRequestReserviceTimer shall be
            equal to or greater than this value for priority request to be
            honored."
      DEFVAL { 0 }
::= { priorityRequestServer 9 }
```

5.1.1.10 Priority Request Reservice Class 6 Time

```
priorityRequestReserviceClass6Time OBJECT-TYPE
      SYNTAX      INTEGER (0..65535)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object represents the time in seconds between the end of
            ANY strategy and the start of any new Class Type 6 priority
            request. The value of the priorityRequestReserviceTimer shall be
            equal to or greater than this value for priority request to be
            honored."
      DEFVAL { 0 }
::= { priorityRequestServer 10 }
```

5.1.1.11 Priority Request Reservice Class 7 Time

```
priorityRequestReserviceClass7Time OBJECT-TYPE
      SYNTAX      INTEGER (0..65535)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object represents the time in seconds between the end of
            ANY strategy and the start of any new Class Type 7 priority
            request. The value of the priorityRequestReserviceTimer shall be
            equal to or greater than this value for priority request to be
            honored."
      DEFVAL { 0 }
::= { priorityRequestServer 11 }
```

5.1.1.12 Priority Request Reservice Class 8 Time

```
priorityRequestReserviceClass8Time OBJECT-TYPE
      SYNTAX      INTEGER (0..65535)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
```

```
            "This object represents the time in seconds between the end of
            ANY strategy and the start of any new Class Type 8 priority
            request. The value of the priorityRequestReserviceTimer shall be
            equal to or greater than this value for priority request to be
            honored."
      DEFVAL { 0 }
::= { priorityRequestServer 12 }
```

5.1.1.13 Priority Request Reservice Class 9 Time

```
priorityRequestReserviceClass9Time OBJECT-TYPE
      SYNTAX      INTEGER (0..65535)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object represents the time in seconds between the end of
            ANY strategy and the start of any new Class Type 9 priority
            request. \ The value of the priorityRequestReserviceTimer shall
            be equal to or greater than this value for priority request to be
            honored."
      DEFVAL { 0 }
::= { priorityRequestServer 13 }
```

5.1.1.14 Priority Request Reservice Class 10 Time

```
priorityRequestReserviceClass10Time OBJECT-TYPE
      SYNTAX      INTEGER (0..65535)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object represents the time in seconds between the end of
            ANY strategy and the start of any new Class Type 10 priority
            request. The value of the priorityRequestReserviceTimer shall be
            equal to or greater than this value for priority request to be
            honored."
      DEFVAL { 0 }
::= { priorityRequestServer 14 }
```

5.1.2 Block Definitions - PRS

```
priorityRequestMessages ::= { scp 2 }
```

5.1.2.1 Priority Request Message

```
prgPriorityRequest OBJECT-TYPE
      SYNTAX      OCTET STRING {SIZE(25)}
      ACCESS      write-only
      STATUS      mandatory
      DESCRIPTION
            "<Definition> An OER encoded string of reference parameters to
            initiate a new priority request.

            The parameter values in this string when sent from a PRG to a PRS
            are:
            priorityRequestID                       INTEGER (1..255)
            priorityRequestVehicleID                OCTET STRING (SIZE 17)
            priorityRequestVehicleClassType         INTEGER (1..10)
```



```
            priorityRequestVehicleClassLevel          INTEGER (1..10)
            priorityRequestServiceStrategyNumber      INTEGER (1..255)
            priorityRequestTimeOfServiceDesired       INTEGER (1..65535)
            priorityRequestTimeOfEstimatedDeparture   INTEGER (1..65535)"
::= { priorityRequestMessages 1 }
```

5.1.2.2 Priority Update Message

```
prgPriorityUpdate OBJECT-TYPE
      SYNTAX      OCTET STRING {SIZE(25)}
      ACCESS      write-only
      STATUS      mandatory
      DESCRIPTION
            "<Definition> An OER encoded string of reference parameters to
            update a priority request.

            The parameter values in this string when sent from a PRG to a PRS
            are:
            priorityRequestID                         INTEGER (1..255)
            priorityRequestVehicleID                  OCTET STRING (SIZE 17)
            priorityRequestVehicleClassType           INTEGER (1..10)
            priorityRequestVehicleClassLevel          INTEGER (1..10)
            priorityRequestServiceStrategyNumber      INTEGER (1..255)
            priorityRequestTimeOfServiceDesired       INTEGER (1..65535)
            priorityRequestTimeOfEstimatedDeparture   INTEGER (1..65535)"
::= { priorityRequestMessages 2 }
```

5.1.2.3 Priority Status Control Message

```
prgPriorityStatusControl OBJECT-TYPE
      SYNTAX      OCTET STRING {SIZE(21)}
      ACCESS      write-only
      STATUS      mandatory
      DESCRIPTION
            "<Definition> An OER encoded string of reference parameters used
            to retrieve the status of a priority request.

            The parameter values in this string when sent from a PRG to a PRS
            are:
            priorityRequestID                         INTEGER (1..255)
            priorityRequestVehicleID                  OCTET STRING (SIZE 17)
            priorityRequestVehicleClassType           INTEGER (1..10)
            priorityRequestVehicleClassLevel          INTEGER (1..10)
            priorityRequestServiceStrategyNumber      INTEGER (1..255)"
::= { priorityRequestMessages 3 }
```

5.1.2.4 Priority Status Buffer Message

```
prgPriorityStatusBuffer OBJECT-TYPE
      SYNTAX      OCTET STRING {SIZE(23)}
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "<Definition> An OER encoded string defining the status of a
             priority request.
```

```
            The parameter values in this string when retrieved from a PRG
            are:
            priorityRequestID                          INTEGER (1..255)
            priorityRequestVehicleID                   OCTET STRING (SIZE 17)
            priorityRequestVehicleClassType            INTEGER (1..10)
            priorityRequestVehicleClassLevel           INTEGER (1..10)
            priorityRequestServiceStrategyNumber       INTEGER (1..255)
            priorityRequestStatusInPRS                 INTEGER (1..255)"
::= { priorityRequestMessages 4 }
```

5.1.2.5 Priority Cancel Message

```
prgPriorityCancel OBJECT-TYPE
      SYNTAX      OCTET STRING {SIZE(21)}
      ACCESS      write-only
      STATUS      mandatory
      DESCRIPTION
            "<Definition> An OER encoded string of reference parameters to
            cancel a priority request.

            The parameter values in this string when sent from a PRG to a PRS
            are:
            priorityRequestID                          INTEGER (1..255)
            priorityRequestVehicleID                   OCTET STRING (SIZE 17)
            priorityRequestVehicleClassType            INTEGER (1..10)
            priorityRequestVehicleClassLevel           INTEGER (1..10)
            priorityRequestServiceStrategyNumber       INTEGER (1..255)"
::= { priorityRequestMessages 5 }
```

5.1.2.6 Priority Clear Message

```
prgPriorityClear OBJECT-TYPE
      SYNTAX      OCTET STRING {SIZE(21)}
      ACCESS      write-only
      STATUS      mandatory
      DESCRIPTION
            "<Definition> An OER encoded string of reference parameters to
            clear a priority request.

            The parameter values in this string when sent from a PRG to a PRS
            are:
            priorityRequestID                          INTEGER (1..255)
            priorityRequestVehicleID                   OCTET STRING (SIZE 17)
            priorityRequestVehicleClassType            INTEGER (1..10)
            priorityRequestVehicleClassLevel           INTEGER (1..10)
            priorityRequestServiceStrategyNumber       INTEGER (1..255)"
::= { priorityRequestMessages 6 }
```

5.1.2.7 Program Data

```
prsProgramData OBJECT-TYPE
      SYNTAX      OCTET STRING {SIZE(23)}
      ACCESS      read-write
      STATUS      mandatory
      DESCRIPTION
            "<Definition> An OER encoded string of reference parameters for
            exchanges between a management station and a PRS.
```



```
            The parameter values in this string are:
            priorityRequestTimeToLiveValue            INTEGER (0..65535)
            priorityRequestReserviceClass1Time        INTEGER (0..65535)
            priorityRequestReserviceClass2Time        INTEGER (0..65535)
            priorityRequestReserviceClass3Time        INTEGER (0..65535)
            priorityRequestReserviceClass4Time        INTEGER (0..65535)
            priorityRequestReserviceClass5Time        INTEGER (0..65535)
            priorityRequestReserviceClass6Time        INTEGER (0..65535)
            priorityRequestReserviceClass7Time        INTEGER (0..65535)
            priorityRequestReserviceClass8Time        INTEGER (0..65535)
            priorityRequestReserviceClass9Time        INTEGER (0..65535)
            priorityRequestReserviceClass10Time       INTEGER (0..65535)"
::= { priorityRequestMessages 7 }
```

5.1.2.8 Priority Request (Absolute Time Reference) Message

```
prgPriorityRequestAbsolute OBJECT-TYPE
      SYNTAX      OCTET STRING {SIZE(29)}
      ACCESS      write-only
      STATUS      mandatory
      DESCRIPTION
            "<Definition> An OER encoded string of reference parameters to
            initiate a new priority request based on an absolute time
            reference used by the PRG.

            The parameter values in this string when sent from a PRG to a PRS
            are:
            priorityRequestID                          INTEGER (1..255)
            priorityRequestVehicleID                   OCTET STRING (SIZE 17)
            priorityRequestVehicleClassType            INTEGER (1..10)
            priorityRequestVehicleClassLevel           INTEGER (1..10)
            priorityRequestServiceStrategyNumber       INTEGER (1..255)
            priorityRequestTimeOfServiceDesired        INTEGER (1..65535)
            priorityRequestTimeOfEstimatedDeparture    INTEGER (1..65535)
            priorityRequestTimeOfRequest             INTEGER (0.. 4294967295)"
::= { priorityRequestMessages 8 }
```

5.1.2.9 Priority Update (Absolute Time Reference) Message

```
prgPriorityUpdateAbsolute OBJECT-TYPE
      SYNTAX      OCTET STRING {SIZE(29)}
      ACCESS      write-only
      STATUS      mandatory
      DESCRIPTION
            "<Definition> An OER encoded string of reference parameters to
            update a priority request based on an absolute time reference
            used by the PRG.

            The parameter values in this string when sent from a PRG to a PRS
            are:
            priorityRequestID                          INTEGER (1..255)
            priorityRequestVehicleID                   OCTET STRING (SIZE 17)
            priorityRequestVehicleClassType            INTEGER (1..10)
            priorityRequestVehicleClassLevel           INTEGER (1..10)
            priorityRequestServiceStrategyNumber       INTEGER (1..255)
            priorityRequestTimeOfServiceDesired        INTEGER (1..65535)
```

```
            priorityRequestTimeOfEstimatedDeparture   INTEGER (1..65535)
            priorityRequestTimeOfRequest            INTEGER (0.. 4294967295)"
::= { priorityRequestMessages 9 }

END -- PRS-MIB1
```

5.2 Coordinator MIB

Section 5.2 defines the information requirements of the CO portion of an SCP system. The objects are defined using the OBJECT-TYPE macro as specified in RFC 1212 and NTCIP 8004 v02. The text provided from Sections 5.2 through the end of Section 5.2.4.2 (except the section headings) constitutes the standard NTCIP 1211 v02 Management Information Base (MIB) for a CO.

The sections below present the objects in lexicographical order of their OBJECT IDENTIFIERS, which correspond to their physical location within the global naming tree. Most of the objects defined in NTCIP 1211 v02 reside under the "scp" node of the global naming tree. To aid in object management, the SCP node has been subdivided into logical categories; each defined by a node under the SCP node. The individual objects are then located under the appropriate node.

Conformance requirements for any object is determined by the use of the Requirements Traceability Matrix (RTM) in Annex A. To support any defined Requirement, an implementation shall support all objects to which the Requirement traces in the RTM. The value of the STATUS field for every object in the MIB is "mandatory", and indicates that it is mandatory if any associated Requirement is selected.

The full standardized range of all objects defined within NTCIP 1211 v02 shall be supported, except as otherwise noted. This MIB is managed by the NTCIP SCP Working Group and proprietary features should be defined through vendor-specific nodes in vendor-specific extensions to this MIB. All values not explicitly defined (e.g., enumerated values not listed, bits not defined, etc.) are reserved for future use by the SCP Working Group.

A computer readable format of this MIB is available from NEMA (ntcip@nema.org). The MIB has been verified using SMICng Version 2.2.07 (Book).

5.2.1 Object Definitions—CO

```
CO-MIB1 DEFINITIONS ::= BEGIN

-- The following Management Information Base represents the data
-- elements that would reside in a Coordinator that implements SCP.

IMPORTS

      EntryStatus, OwnerString, OerString,
      Byte, UByte, Short, Ushort, Long, Ulong
      FROM NEMA-SMI

      scp
      FROM NEMA-SMI2

      OBJECT-TYPE
      FROM RFC-1212

      splitNumber, splitPhase
      FROM SCP-MIB1
```



```
      TrueFalse ::= INTEGER (0 | 255)

-- This group of object represents the output from the prioritization
-- process and the parameters that Coordinator acts upon.

priorityStrategyServer ::= { scp 3 }
```

5.2.1.1 Maximum Service Requests to Consider

```
maxStrategyRequestsToConsider OBJECT-TYPE
      SYNTAX      INTEGER (1..10)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            " This object represents the number of rows in the
            priorityStrategyRequestedTable that the CO shall consider when it
            tries to determine which strategies to execute. The number of
            rows in the table is fixed at 10 but an implementation is only
            required to act upon the first row."
::= { priorityStrategyServer 1 }
```

5.2.1.2 Priority Strategy Requested Table

```
priorityStrategyRequestedTable OBJECT-TYPE
      SYNTAX      SEQUENCE OF priorityStrategyRequestedEntry
      ACCESS      not-accessible
      STATUS      mandatory
      DESCRIPTION
            "A static table containing the parameters associated with
            executing a priority strategy. The number of rows in this table
            is equal to 10."
::= { priorityStrategyServer 2 }

priorityStrategyRequestedEntry OBJECT-TYPE
      SYNTAX      PriorityStrategyRequestedEntry
      ACCESS      not-accessible
      STATUS      mandatory
      DESCRIPTION
            "This object defines the parameters that are associated with a
            service request."
      INDEX       { priorityStrategyRequestedNumber }
::= { priorityStrategyRequestedTable 1 }

priorityStrategyRequestedEntry ::= SEQUENCE {
      priorityStrategyRequestNumber                         INTEGER,
      priorityStrategyRequested                             INTEGER,
      priorityStrategyRequestedTimeOfServiceDesired         INTEGER,
      priorityStrategyRequestedTimeOfEstimatedDeparture     INTEGER,
      priorityStrategyRequestStatusInCO                     INTEGER}
```

5.2.1.2.1 Priority Strategy Request Number

```
priorityStrategyRequestNumber OBJECT-TYPE
      SYNTAX      INTEGER (1..10)
      ACCESS      read-only
      STATUS      mandatory
```

```
      DESCRIPTION
            "This object represents the index used to reference rows in the
            priorityStrategyRequestedTable."
::= { priorityStrategyRequestedEntry 1 }
```

5.2.1.2.2 Priority Strategy Requested

```
priorityStrategyRequested OBJECT-TYPE
      SYNTAX      INTEGER (0..255)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object represents a strategy being requested by the PRS. A
            value of zero (0) shall indicate that none is request."
::= { priorityStrategyRequestedEntry 2 }
```

5.2.1.2.3 Priority Strategy Requested Time of Service Desired

```
priorityStrategyRequestedTimeOfServiceDesired OBJECT-TYPE
      SYNTAX      INTEGER -- (0..4294967295)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object represents the estimated time of service desired
            expressed as global time. This value is received from the PRS via
            the prsServiceRequest message's
            priorityRequestTimeOfServiceDesiredInPRS."
::= { priorityStrategyRequestedEntry 3 }
```

5.2.1.2.4 Priority Strategy Requested Time of Estimated Departure

```
priorityStrategyRequestedTimeOfEstimatedDeparture OBJECT-TYPE
      SYNTAX      INTEGER -- (0..4294967295)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object represents the estimated time of departure expressed
            as global time. This value is received from the PRS via the
            prsServiceRequest message's
            priorityRequestTimeOfServiceDesiredInPRS."
::= { priorityStrategyRequestedEntry 4 }
```

5.2.1.2.5 Priority Strategy Request Status (CO)

```
priorityStrategyRequestStatusInCO   OBJECT-TYPE
      SYNTAX      INTEGER { idleNotValid (1),
                  readyQueued (2),
                  readyOverridden (3),
                  activeProcessing (4),
                  activeCancel (5),
                  activeOverride (6),
                  activeNotOverridden (7),
                  closedCanceled (8),
                  reserviceError (9),
                  closedTimeToLiveError (10),
                  closedTimerError (11),
                  closedStrategyError (12),
```



```
            closedCompleted (13),
            activeAdjustNotNeeded (14),
            closedFlash (15) }
ACCESS      read-only
STATUS      mandatory
DESCRIPTION
      "This object provides an image of priorityRequestStatusInPRS that
      was sent by the PRS or a response to readyQueued, activeCancel,
      or activeOverride.
            idleNotValid      PRS determined that the request does not
                              contain valid data
            readyQueued       PRS has validated the request but is
                              waiting for the CO to activate
            readyOverridden   CO has overridden the request
            activeProcessing  CO is processing the requested strategy
            activeCancel      PRS has asked that request be canceled
            activeOverride    PRS has asked that request be overridden
            activeNotOverridden     CO did not process the requested
                              override
            closedCanceled    CO has canceled the request
            reserviceError    PRs determined that the request came too
                              soon after a previous request
            closedTimeToLiveError   PRS determined that TSD exceeds the
                              time to live
            closedTimerError  CO determined that the requested times
                              could not be met
            closedStrategyError CO determined that the requested
                              strategy was not valid.
            ClosedCompleted   CO has completed the requested strategy
                              request
            ActiveAdjustNotNeeded   CO determined that the request
                              could be met with current timing and
                              adjustment was not needed
            ClosedFlash       CO determined that the controller was
                              operating in flash.

      Upon receipt of a readyQueued, the CO may change the status to:
            activeProcessing
            closedTimerError
            closedStrategyError
            activeAdjustNotNeeded
            closedFlash

      Upon receipt of a activeCancel, the CO may change the status to:
            closedCanceled
            closedCompleted

      Upon receipt of a activeOverride, the CO may change the status
      to:
            activeNotOverridden
            readyOverridden
            closedCompleted

      The CO does NOT take any action when one of the following status
      is received from the PRS:
            idleNotValid
            readyOverridden
```

```
                  activeProcessing
                  activeNotOverridden
                  closedCanceled
                  reserviceError
                  closedTimeToLiveError
                  closedTimerError
                  closedStrategyError
                  closedCompleted
                  activeAdjustNotNeeded
                  closedFlash

            Note: A change of status is predicated on the "busy" signal being
            set."
      DEFVAL { idleNotValid }
::= { priorityStrategyRequestedEntry 5}
```

5.2.1.3 Coordinator Busy

```
coBusy OBJECT-TYPE
      SYNTAX      TrueFalse
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object is used to indicate to a PRS that the CO is
            initializing or otherwise processing and when TRUE, the
            information included with it in a prsServiceRequest response is
            not valid. It is used to synchronize exchanges between a PRS and
            CO.

            When the PRS is acting as manager and the CO is acting as an
            agent, coBusy is set TRUE when the PRS performs a SetRequest
            prsServiceRequest.

            When the CO is acting as a manager and the PRS is acting as the
            agent, coBusy is set TRUE when the PRS responds to a GetRequest
            prsServiceRequest from the CO with prsBusy set to FALSE. "
::= { priorityStrategyServer 3 }

-- The following set of objects defines a table of priority strategy
-- parameters and a list of per unit parameters. In the table, each row
-- in the table represents data associated with a particular strategy.
```

5.2.1.4 Priority Strategies Max

```
priorityStrategiesMax OBJECT-TYPE
      SYNTAX      INTEGER (1..255)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "The maximum number of priority strategies supported by this
            entity."
::= { priorityStrategyServer 4}
```

5.2.1.5 Priority Strategies Table

```
priorityStrategyTable OBJECT-TYPE
      SYNTAX      SEQUENCE OF PriorityStrategyEntry
```



```
      ACCESS      not-accessible
      STATUS      mandatory
      DESCRIPTION
            "A static table containing the parameters associated with each
            priority strategy. The number of rows in this table is equal to
            the priorityStrategiesMax object."
::= { priorityStrategyServer 5 }

priorityStrategyEntry OBJECT-TYPE
      SYNTAX      PriorityStrategyEntry
      ACCESS      not-accessible
      STATUS      mandatory
      DESCRIPTION
            "Parameters associated with a specific priority strategy."
      INDEX       { priorityStrategyNumber }
::= { priorityStrategyTable 1 }

priorityStrategyEntry ::= SEQUENCE {
      priorityStrategyNumber                    INTEGER,
      priorityStrategyServicePhases             OCTET STRING,
      priorityStrategyPhaseOmits                OCTET STRING,
      priorityStrategyPedOmits                  OCTET STRING,
      priorityStrategyDescription               OCTET STRING}
```

5.2.1.5.1 Priority Strategy Number

```
priorityStrategyNumber OBJECT-TYPE
      SYNTAX      INTEGER (1..255)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object serves as the index in the priorityStrategyTable."
::= { priorityStrategyEntry 1}
```

5.2.1.5.2 Priority Strategy Service Phases

```
priorityStrategyServicePhases OBJECT-TYPE
      SYNTAX      OCTET STRING
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object defines the phase or phases that represent the
            movement on which priority service is desired. Each octet (binary
            value) within the octet string contains an SCP.phaseNumber.

            There can only be one phase per ring. All defined Service Phases
            shall be able to time concurrently. It is assumed that any
            possible concurrent phase are allowed to be serviced in order
            with no backing up unless otherwise permitted."
      DEFVAL { "" }
::= { priorityStrategyEntry 2}
```

5.2.1.5.3 Priority Strategy Phase Omits

```
priorityStrategyPhaseOmits OBJECT-TYPE
      SYNTAX      OCTET STRING
      ACCESS      read-only
```

```
      STATUS      mandatory
      DESCRIPTION
            "This object represents the phase(s) that are to be omitted
            during the priority service. Each octet (binary value) within the
            octet string contains an SCP.phaseNumber. The CO shall ignore any
            omit applied to the coordinated phase(s). These phase omits are
            logically "OR'ed" with any other phase omits that may be active."
      DEFVAL { "" }
::= { priorityStrategyEntry 3}
```

5.2.1.5.4 Priority Strategy Ped Omits

```
priorityStrategyPedOmits OBJECT-TYPE
      SYNTAX      OCTET STRING
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object represents the pedestrian movement(s) that are to be
            omitted during the priority service. Each octet (binary value)
            within the octet string contains an SCP.phaseNumber. These ped
            omits are logically "OR'ed" with any other ped omits that may be
            active. These ped omits do not apply to non-actuated phases."
      DEFVAL { "" }
::= { priorityStrategyEntry 4}
```

5.2.1.5.5 Priority Strategy Description

```
priorityStrategyDescription OBJECT-TYPE
      SYNTAX      OCTET STRING (SIZE (0..40))
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object represents a text description of the strategy. It is
            intended to convey to priority service requesters what signals or
            movements are given priority. It should be defined in language
            that is domain neutral."
::= { priorityStrategyEntry 5}

-- The following set of objects represent per unit parameters that
-- apply to all strategies. It includes a set of database variables
-- that relate to an extension of the splitTable as defined in NTCIP
-- 1202:2005.
```

5.2.1.6 Priority Strategy Extension to Split Table

```
priorityStrategyExtensionToSplitTable OBJECT-TYPE
      SYNTAX      SEQUENCE OF PriorityStrategyExtensionToSplitEntry
      ACCESS      not-accessible
      STATUS      mandatory
      DESCRIPTION
            "A table containing priority strategy parameters that are
            associated with each split and each phase of that split.

            This is an extension to the splitTable as defined in the NTCIP
            1202:2005."
::= { priorityStrategyServer 6 }
```



```
priorityStrategyExtensionToSplitEntry OBJECT-TYPE
      SYNTAX      PriorityStrategyExtensionToSplitEntry
      ACCESS      not-accessible
      STATUS      mandatory
      DESCRIPTION
            "A table containing priority strategy parameters that are
            associated with each split and each phase of that split."
      INDEX       {splitNumber, splitPhase}
::= { priorityStrategyExtensionToSplitTable 1}

priorityStrategyExtensionToSplitEntry ::= SEQUENCE {
      priorityStrategyMaximumReductionTime INTEGER,
      priorityStrategyMaximumExtensionTime INTEGER}
```

5.2.1.6.1 Priority Strategy Maximum Reduction Time

```
priorityStrategyMaximumReductionTime OBJECT-TYPE
      SYNTAX      INTEGER (0..255)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object represents the maximum number of seconds that the
            split of a phase can be reduced by during a priority strategy.

            This effective value may be reduced by the CO based on the
            operating phase minimum service requirements.

            Note - The definition of a phases minimum service is dependent on
            whether it is actuated or non-actuated, whether the ped clear
            terminates before the yellow or not, whether variable initial is
            active or not, whether it is a coord phase or not, and whether it
            is in a single or multi-ring configuration."
      DEFVAL { 0 }
::= { priorityStrategyExtensionToSplitEntry 1 }
```

5.2.1.6.2 Priority Strategy Maximum Extension Time

```
priorityStrategyMaximumExtensionTime OBJECT-TYPE
      SYNTAX      INTEGER (0..255)
      ACCESS      read-only
      STATUS      mandatory
      DESCRIPTION
            "This object represents the maximum number of seconds that the
            split of a phase can be extended during a priority strategy.

            The effective value may be modified by the CO based on time
            gained from one cycle of reductions to the respective non-
            priority phases."
      DEFVAL { 0 }
::= { priorityStrategyExtensionToSplitEntry 2 }
```

5.2.1.7 Priority Strategy Default Coordination Pattern

```
priorityStrategyDefaultCoordPattern OBJECT-TYPE
      SYNTAX      INTEGER (1..253)
      ACCESS      read-write
      STATUS      mandatory
```

```
      DESCRIPTION
            "This object represents the coordination pattern timing
            parameters to use if the controller is not operating in a
            coordinated mode."
::= { priorityStrategyServer 7 }
```

5.2.2 Priority Service Request and Response Block Objects

```
-- The following block object definition is defined for exchanges
-- between a PRS and a CO.

priorityStrategyMessages ::= { scp 4}
```

5.2.2.1 Service Request

```
prsServiceRequest OBJECT-TYPE
      SYNTAX      OCTET STRING {SIZE(110)}
      ACCESS      read-write
      STATUS      mandatory
      DESCRIPTION
            "<Definition> An OER encoded string of reference parameters for
            exchanges between a CO and a PRS.

            The parameter values of this OID when transferring information
            from a PRS to a CO are:
            priorityRequestServiceStrategyNumber.1    INTEGER (1..255)
            priorityRequestTimeOfServiceDesiredInPRS.1      INTEGER
            (0..4294967295)
            priorityRequestTimeOfEstimatedDepartureInPRS.1  INTEGER
            (0..4294967295)
            priorityRequestStatusInPRS.1              INTEGER (1..255)
            priorityRequestServiceStrategyNumber.2    INTEGER (1..255)
            priorityRequestTimeOfServiceDesiredInPRS.2      INTEGER
            (0..4294967295)
            priorityRequestTimeOfEstimatedDepartureInPRS.2  INTEGER
            (0..4294967295)
            priorityRequestStatusInPRS.2              INTEGER (1..255)
            priorityRequestServiceStrategyNumber.3    INTEGER (1..255)
            priorityRequestTimeOfServiceDesiredInPRS.3      INTEGER
            (0..4294967295)
            priorityRequestTimeOfEstimatedDepartureInPRS.3  INTEGER
            (0..4294967295)
            priorityRequestStatusInPRS.3              INTEGER (1..255)
            priorityRequestServiceStrategyNumber.4    INTEGER (1..255)
            priorityRequestTimeOfServiceDesiredInPRS.4      INTEGER
            (0..4294967295)
            priorityRequestTimeOfEstimatedDepartureInPRS.4  INTEGER
            (0..4294967295)
            priorityRequestStatusInPRS.4              INTEGER (1..255)
            priorityRequestServiceStrategyNumber.5    INTEGER (1..255)
            priorityRequestTimeOfServiceDesiredInPRS.5      INTEGER
            (0..4294967295)
            priorityRequestTimeOfEstimatedDepartureInPRS.5  INTEGER
            (0..4294967295)
            priorityRequestStatusInPRS.5              INTEGER (1..255)
            priorityRequestServiceStrategyNumber.6    INTEGER (1..255)
```



```
priorityRequestTimeOfServiceDesiredInPRS.6      INTEGER
(0..4294967295)
priorityRequestTimeOfEstimatedDepartureInPRS.6  INTEGER
(0..4294967295)
priorityRequestStatusInPRS.6              INTEGER (1..255)
priorityRequestServiceStrategyNumber.7    INTEGER (1..255)
priorityRequestTimeOfServiceDesiredInPRS.7      INTEGER
(0..4294967295)
priorityRequestTimeOfEstimatedDepartureInPRS.7  INTEGER
(0..4294967295)
priorityRequestStatusInPRS.7              INTEGER (1..255)
priorityRequestServiceStrategyNumber.8    INTEGER (1..255)
priorityRequestTimeOfServiceDesiredInPRS.8      INTEGER
(0..4294967295)
priorityRequestTimeOfEstimatedDepartureInPRS.8  INTEGER
(0..4294967295)
priorityRequestStatusInPRS.8              INTEGER (1..255)
priorityRequestServiceStrategyNumber.9    INTEGER (1..255)
priorityRequestTimeOfServiceDesiredInPRS.9      INTEGER
(0..4294967295)
priorityRequestTimeOfEstimatedDepartureInPRS.9  INTEGER
(0..4294967295)
priorityRequestStatusInPRS.9              INTEGER (1..255)
priorityRequestServiceStrategyNumber.10   INTEGER (1..255)
priorityRequestTimeOfServiceDesiredInPRS.10     INTEGER
(0..4294967295)
priorityRequestTimeOfEstimatedDepartureInPRS.10 INTEGER
(0..4294967295)
priorityRequestStatusInPRS.10             INTEGER (1..255)
prsBusy                                   INTEGER (0..1)

The parameter values for this OID when transferring information
from a CO to a PRS are:
priorityRequestServiceStrategyNumber.1    INTEGER (1..255)
priorityStrategyRequestedTimeOfServiceDesired.1 INTEGER
(0..4294967295)
priorityStrategyRequestedTimeOfEstimatedDeparture.1   INTEGER
(0..4294967295)
priorityStrategyRequestStatusInCO.1       INTEGER(1..255)
priorityRequestServiceStrategyNumber.2    INTEGER (1..255)
priorityStrategyRequestedTimeOfServiceDesired.2 INTEGER
(0..4294967295)
priorityStrategyRequestedTimeOfEstimatedDeparture.2   INTEGER
(0..4294967295)
priorityStrategyRequestStatusInCO.2       INTEGER(1..255)
priorityRequestServiceStrategyNumber.3    INTEGER (1..255)
priorityStrategyRequestedTimeOfServiceDesired.3 INTEGER
(0..4294967295)
priorityStrategyRequestedTimeOfEstimatedDeparture.3   INTEGER
(0..4294967295)
priorityStrategyRequestStatusInCO.3       INTEGER(1..255)
priorityRequestServiceStrategyNumber.4    INTEGER (1..255)
priorityStrategyRequestedTimeOfServiceDesired.4 INTEGER
(0..4294967295)
priorityStrategyRequestedTimeOfEstimatedDeparture.4   INTEGER
(0..4294967295)
priorityStrategyRequestStatusInCO.4       INTEGER(1..255)
```

```
            priorityRequestServiceStrategyNumber.5    INTEGER (1..255)
            priorityStrategyRequestedTimeOfServiceDesired.5 INTEGER
            (0..4294967295)
            priorityStrategyRequestedTimeOfEstimatedDeparture.5   INTEGER
            (0..4294967295)
            priorityStrategyRequestStatusInCO.5       INTEGER(1..255)
            priorityRequestServiceStrategyNumber.6    INTEGER (1..255)
            priorityStrategyRequestedTimeOfServiceDesired.6 INTEGER
            (0..4294967295)
            priorityStrategyRequestedTimeOfEstimatedDeparture.6   INTEGER
            (0..4294967295)
            priorityStrategyRequestStatusInCO.6       INTEGER(1..255)
            priorityRequestServiceStrategyNumber.7    INTEGER (1..255)
            priorityStrategyRequestedTimeOfServiceDesired.7 INTEGER
            (0..4294967295)
            priorityStrategyRequestedTimeOfEstimatedDeparture.7   INTEGER
            (0..4294967295)
            priorityStrategyRequestStatusInCO.7       INTEGER(1..255)
            priorityRequestServiceStrategyNumber.8    INTEGER (1..255)
            priorityStrategyRequestedTimeOfServiceDesired.8 INTEGER
            (0..4294967295)
            priorityStrategyRequestedTimeOfEstimatedDeparture.8   INTEGER
            (0..4294967295)
            priorityStrategyRequestStatusInCO.8       INTEGER(1..255)
            priorityRequestServiceStrategyNumber.9    INTEGER (1..255)
            priorityStrategyRequestedTimeOfServiceDesired.9 INTEGER
            (0..4294967295)
            priorityStrategyRequestedTimeOfEstimatedDeparture.9   INTEGER
            (0..4294967295)
            priorityStrategyRequestStatusInCO.9       INTEGER(1..255)
            priorityRequestServiceStrategyNumber.10   INTEGER (1..255)
            priorityStrategyRequestedTimeOfServiceDesired.10      INTEGER
            (0..4294967295)
            priorityStrategyRequestedTimeOfEstimatedDeparture.10  INTEGER
            (0..4294967295)
            priorityStrategyRequestStatusInCO.10      INTEGER(1..255)"
::= { priorityStrategyMessages 1 }
```

5.2.3 SCP Block Objects

```
scpBlock OBJECT IDENTIFIER ::= { scp 5 }

-- This object is an identifier used to group all objects for
-- support of SCP Block Upload and Download activities.
```

5.2.3.1 SCP Block Get Control

```
scpBlockGetControl OBJECT-TYPE
      SYNTAX      OCTET STRING (SIZE(2..12))
      ACCESS      read-write
      STATUS      mandatory
      DESCRIPTION
            "<Definition> An OER encoded string of reference parameters for
            SCP Block Uploads. The parameter values in this string are:
                  scpBlockDataType  INTEGER (0..255)
                  scpBlockDataID    INTEGER (0..255)
```



```
                  scpBlockIndex1    INTEGER (0..255) if needed
                  scpBlockQuantity1 INTEGER (0..255) if needed
                  scpBlockIndex2    INTEGER (0..255) if needed
                  scpBlockQuantity2 INTEGER (0..255) if needed
                  scpBlockIndex3    INTEGER (0..255) if needed
                  scpBlockQuantity3 INTEGER (0..255) if needed
                  scpBlockIndex4    INTEGER (0..255) if needed
                  scpBlockQuantity4 INTEGER (0..255) if needed
                  scpBlockIndex5    INTEGER (0..255) if needed
                  scpBlockQuantity5 INTEGER (0..255) if needed

            A GET of scpBlockData shall utilize values currently in this
            object to define the data to be returned.

            A SET of this object shall be evaluated for validity and Error
            Status of badValue(3) be returned for the following conditions:
                  1) scpBlockDataType is not supported
                  2) scpBlockDataID is not supported
                  3) scpBlockIndex1 is zero or not supported
                  4) scpBlockQuantity1 is zero or scpBlockIndex1 +
                                    scpBlockQuantity1 - 1 is not supported
                  5) scpBlockIndex2 is zero or not supported
                  6) scpBlockQuantity2 is zero or scpBlockIndex2 +
                                    scpBlockQuantity2) - 1 is not supported
                  7) scpBlockIndex3 is zero or not supported
                  8) scpBlockQuantity3 is zero or scpBlockIndex3 +
                                    scpBlockQuantity3) - 1 is not supported
                  9) scpBlockIndex4 is zero or not supported
                  10) scpBlockQuantity4 is zero or scpBlockIndex4 +
                                    scpBlockQuantity4) - 1 is not supported
                  11) scpBlockIndex5 is zero or not supported
                  12) scpBlockQuantity5 is zero or scpBlockIndex5 +
                                    scpBlockQuantity5) - 1 is not supported
                  13) if the SET length is zero or incorrect for
                                    scpBlockDataType & scpBlockDataID
                  14) if the GetResponse length for a GET on scpBlockData
                                    using maximum data field sizes would
                                    exceed a local limitation

            When this validity check fails, scpBlockErrorStatus shall be set
            equal to the Bullet Value above that generated the error.
            <DescriptiveName> NTCIP-1211::SCP.scpBlockGetControl
            <DataConceptType> Data Frame
            <Unit>"
::= { scpBlock 1 }
```

5.2.3.2 SCP Block Data

```
scpBlockData OBJECT-TYPE
      SYNTAX      OCTET STRING (SIZE(2..484))
      ACCESS      read-write
      STATUS      mandatory
      DESCRIPTION
            "<Definition> An OER encoded string used for uploading and
            downloading SCP parameters. See SECTION 5 for encoding and
            decoding the block.
```

```
            A SET on this object shall require the use of
            'dbCreateTransaction' defined in NTCIP 1201 v03 Section 2.3.1.

            A SET of this object shall be evaluated for validity and Error
            Status of badValue(3) be returned for the following conditions:
                  1) scpBlockDataType is not supported
                  2) scpBlockDataID is not supported
                  3) scpBlockIndex1 is zero or not supported
                  4) scpBlockQuantity1 is zero or scpBlockIndex1 +
                                    scpBlockQuantity1 - 1 is not supported
                  5) scpBlockIndex2 is zero or not supported
                  6) scpBlockQuantity2 is zero or scpBlockIndex2 +
                                    scpBlockQuantity2) - 1 is not supported
                  7) scpBlockIndex3 is zero or not supported
                  8) scpBlockQuantity3 is zero or scpBlockIndex3 +
                                    scpBlockQuantity3) - 1 is not supported
                  9) scpBlockIndex4 is zero or not supported
                  10) scpBlockQuantity4 is zero or scpBlockIndex4 +
                                    scpBlockQuantity4) - 1 is not supported
                  11) scpBlockIndex5 is zero or not supported
                  12) scpBlockQuantity5 is zero or scpBlockIndex5 +
                                    scpBlockQuantity5) - 1 is not supported
                  13) if the SET length is zero or incorrect for
                                    scpBlockDataType & scpBlockDataID
                  14) if the SET (SEQUENCE OF) value is incorrect.

            When this validity check fails, scpBlockErrorStatus shall be set
            equal to the Bullet Value above that generated the error.

            A SET that includes an unsupported value for a supported data
            element shall return an Error Status of badValue(3) and
            scpBlockErrorStatus shall be set equal to: (data Sequence # *
            100) + data Element #

            A SET that includes a non-zero or non-null value in the position
            of an unsupported data element shall return an Error Status of
            badValue(3) and scpBlockErrorStatus shall be set equal to: (data
            Sequence # * 100) + data Element #

            A GET on this object shall utilize values currently in
            scpBlockGetControl to define the data to be returned. When
            scpBlockGetControl has invalid data, an Error Status of
            badValue(3) shall be returned.

            A GET shall return a zero or null value in the position of an
            unsupported object.
            <DescriptiveName> NTCIP-1211::SCP.scpBlockData
            <DataConceptType> Data Frame
            <Unit> "
::= { scpBlock 2 }
```

5.2.3.3 SCP Block Error Status

```
scpBlockErrorStatus OBJECT-TYPE
      SYNTAX      INTEGER (0..65535)
      ACCESS      read-only
      STATUS      mandatory
```



```
      DESCRIPTION
            "<Definition> This object defines the data element within
            scpBlockGetControl or scpBlockData that caused a badValue(3)
            ErrorStatus.

            This object should equal zero after any successful SET to
            scpBlockGetControl or scpBlockData.
            <DescriptiveName> NTCIP-1211::SCP.scpBlockErrorStatus
            <DataConceptType> Data Element"
::= { scpBlock 3 }

END -- CO-MIB1
```

5.2.4 Coordination Processor Block Object Definitions

All SCP Block Objects shall begin with two octets that define the Data Type and Data ID.

The Data Type octet (scpBlockDataType) provides for the definition of both NTCIP Standard and Device Proprietary data blocks. NTCIP Standard Data Blocks shall utilize an 'scpBlockDataType' of zero. Device Proprietary Data Blocks shall utilize an 'scpBlockDataType' equal to the Private Node Number (PNN) as assigned by NEMA (1.3.6.1.4.1.1206.3.PNN).

dataType	Description
0x00	Standard Data Block
0XPNN	Device Proprietary Data Block

The Data ID octet (scpBlockDataID) provides for definition of included data parameters. NCTIP Standard Data Blocks shall include an 'scpBlockDataID' as listed in Table 6.

Table 6 scpBlockData-dataID Definitions

DataID	Name	Description
0x00	CoStrategyPlanBlock	Defines Priority Phase, Omits, etc.
0x01	CoStrategySplitsBlock	Defines Max Extensions and Reductions
0x02-0x7F		Reserved For NTCIP SCP Usage

5.2.4.1 Coordinator Strategy Plan Block Definition

```
-- scpBlockData values for standard Block
-- coStrategyPlanBlock Data shall be as follows:

coStrategyPlanBlock ::= SEQUENCE {
      scpBlockDataType        INTEGER (0..255), -- 0x00 standard block
      scpBlockDataID          INTEGER (0..255), -- 0x00 coStrategyPlanBlock
      scpBlockIndex1          INTEGER (0..255), -- priorityStrategyNumber
      scpBlockQuantity1       INTEGER (0..255), -- ## of strategies

      -- for (
      --          x = scpBlockIndex1;
      --          x < (scpBlockIndex1 + scpBlockQuantity1);
      --          x++)

      data        SEQUENCE OF CoStrategyPlanBlockData}

coStrategyPlanBlockData ::= SEQUENCE {
      priorityStrategyServicePhases.x           OCTET STRING,
```

```
      priorityStrategyPhaseOmits.x              OCTET STRING,
      priorityStrategyPedOmits.x                OCTET STRING,
      priorityStrategyDescription.x             OCTET STRING (SIZE 0..40)}
```

5.2.4.2 Coordinator Strategy Splits Block Object Definition

```
-- scpBlockData values for standard Block
-- coStrategySplitsBlock Data shall be as follows:

coStrategySplitsBlock ::= SEQUENCE {
      scpBlockDataType        INTEGER (0..255), -- 0x00 standard block
      scpBlockDataID          INTEGER (0..255), -- 0x01 coStrategySplitsBlock
      scpBlockIndex1          INTEGER (0..255), -- splitPhase
      scpBlockQuantity1       INTEGER (0..255), -- ## of phases
      scpBlockIndex2          INTEGER (0..255), -- splitNumber
      scpBlockQuantity2       INTEGER (0..255), -- ## of splits

      -- for (
      --          y = scpBlockIndex2;
      --          y < (scpBlockIndex2 + scpBlockQuantity2);
      --          y++)
      -- for (
      --          x = scpBlockIndex1;
      --          x < (scpBlockIndex1 + scpBlockQuantity1);
      --          x++)

      data        SEQUENCE OF CoStrategySplitsBlockData }

coStrategySplitsBlockData ::= SEQUENCE {
      priorityStrategyMaximumReductionTime.y.x  INTEGER,
      priorityStrategyMaximumExtensionTime.y.x  INTEGER }
```


Annex A
Requirements Traceability Matrix (RTM) [Normative]

The Requirements Traceability Matrix (RTM) links the Functional Requirements as presented in Section 3 with the corresponding Dialogs (Section 4.2) on the same (gray) line. Each Functional Requirement/Dialog relates/uses one or more groups of Objects. The Objects (also known as Data Elements) are listed to the side; the formal definition of each object is contained within Section 5. Using this table, each Functional Requirement can thus be traced in a standardized way.

> Note: The INDEX objects in any of the tables are not explicitly exchanged but are used as index values for other objects that are exchanged.

The audience for this table is implementers (vendors and central system developers) and conformance testers. Additionally, other interested parties might use this table to determine how particular functions are to be implemented using the standardized dialogs, interfaces, and object definitions.

To conform to a requirement, an SCP system shall implement all objects traced from that requirement; and unless otherwise indicated, shall implement all dialogs traced from the requirement. To be consistent with a requirement, an SCP system shall be able to fulfill the requirement using only objects that a conforming SCP system is required to support.

Section 3 defines Supplemental Requirements, which are refining other functional requirements. These functional requirements in turn are generally traced to design elements (e.g., rather than being directly traced to design elements).

> Note: Visit www.ntcip.org for information on availability of electronic copies of the RTM.

A.1 Notation [Informative]

A.1.1 Functional Requirement Columns

The functional requirements are defined within Section 3 and the RTM is based upon the requirements within that Section. The section number and the functional requirement name are indicated within these columns.

A.1.2 Dialog Column

The standardized dialogs are defined within Section 4 and the RTM references the traces from requirements to this dialog. The section number of the dialog is indicated within this column.

A.1.3 Object Columns

The objects are defined within Section 5 of NTCIP 1211 v02 and Section 2 of NTCIP 1201 v03. The RTM references the data objects that are referenced by the dialog. The section number and object name are indicated within these columns.

A.1.4 Additional Specifications

The "Additional Specifications" column may (and should) be used to provide additional notes and requirements about the dialog or may be used by an implementer to provide any additional details about the implementation.

A.2 Instructions For Completing The RTM [Informative]

To find the standardized design content for a functional requirement, search for the requirement identification number and functional requirement under the functional requirements columns. Next to the functional requirements column is be a dialog identification number, identifying either a generic dialog (found in Annex G) or a specified dialog (found in Section 4.2) to be used to fulfill that requirement. To the right of the dialog identification number are the identification number and name of the data objects that are referenced or used by the dialog to fulfill the functional requirement. Object definitions specific to NTCIP 1211 v02 can be found in Section 5. If an object is defined in a different standard, that standard shall be listed first, followed by the section number where the object definition can be found. The "Additional Specifications" column provides additional notes or details about the design content.

A.3 Requirements Traceability Matrix (RTM) Table

Table 7 Requirements Traceability Matrix (RTM)

Requirements Traceability Matrix (RTM)					
FR ID	**Functional Requirement**	**Dialog ID**	**Object ID**	**Object Name**	**Additional Specifications**
3.4	**Architectural Requirements**				
3.4.1	Support Communications From Multiple Entities				
3.4.1.1	Provide Data				
		G.1			
3.4.1.2	Receive Data				
		G.3			
3.4.1.3	Explore Data				
		G.2			
3.5	**Data Exchange and Operational Environment Requirements**				
3.5.1	Interface – Management Station to PRS				
3.5.1.1	Set Reservice Period				

Requirements Traceability Matrix (RTM)					
FR ID	**Functional Requirement**	**Dialog ID**	**Object ID**	**Object Name**	**Additional Specifications**
		4.2.1.1			
			5.1.2.7	prsProgramData	
3.5.1.2	Set Time To Live Period				
		4.2.1.1			
			5.1.2.7	prsProgramData	
3.5.1.3	Retrieve Priority Request Server Settings				
3.5.1.3.1	Retrieve Priority Request Settings				
		G.1			
			5.1.2.7	prsProgramData	
3.5.1.3.2	Retrieve Reservice Period for a Vehicle Class				
		G.1			
			5.1.1.5	priorityRequestReserviceClass1Time	
			5.1.1.6	priorityRequestReserviceClass2Time	
			5.1.1.7	priorityRequestReserviceClass3Time	
			5.1.1.8	priorityRequestReserviceClass4Time	
			5.1.1.9	priorityRequestReserviceClass5Time	
			5.1.1.10	priorityRequestReserviceClass6Time	
			5.1.1.11	priorityRequestReserviceClass7Time	
			5.1.1.12	priorityRequestReserviceClass8Time	
			5.1.1.13	priorityRequestReserviceClass9Time	
			5.1.1.14	priorityRequestReserviceClass10Time	
3.5.1.3.3	Retrieve Priority Request Time To Live Value				
		G.1			
			5.1.1.3	priorityRequestTimeToLiveValue	
3.5.1.4	Monitor the Status of the PRS				
		G.1			
			5.1.1.1	priorityRequestTable	
			5.1.1.1.6	priorityRequestServiceStrategyNumber	
			5.1.1.1.9	priorityRequestStatusInPRS	

Requirements Traceability Matrix (RTM)					
FR ID	**Functional Requirement**	**Dialog ID**	**Object ID**	**Object Name**	**Additional Specifications**
			5.1.1.1.12	priorityRequestTimeOfServiceDesiredInPRS	
			5.1.1.1.13	priorityRequestTimeOfEstimatedDepartureInPRS	
			5.1.1.2	prsBusy	
3.5.2	Interface – Management Station to CO				
3.5.2.1	Configure the CO				
3.5.2.1.1	Set Priority Strategy Configuration				
		4.2.2.1			
			5.2.4.1	coStrategyPlanBlock	
			5.2.4.2	coStrategySplitsBlock	
			5.2.1.7	priorityStrategyDefaultCoordPattern	
			NTCIP 1201 v03 Sec. 2.3.1	dbCreateTransaction	
3.5.2.1.2	Define Default Coordination Pattern				
		G.3			
			5.2.1.7	priorityStrategyDefaultCoordPattern	
3.5.2.1.3	Define Maximum Priority Strategies Supported				
		G.3			
			5.2.1.4	priorityStrategiesMax	
3.5.2.1.4	Define Maximum Service Requests To Consider				
		G.3			
			5.2.1.1	maxStrategyRequestsToConsider	
3.5.2.2	Retrieve Priority Strategy Configuration				
3.5.2.2.1	Retrieve Priority Strategy Settings				
		4.2.2.2			
			5.2.3.1	scpBlockGetControl	
			5.2.3.2	scpBlockData	
			5.2.3.3	scpBlockErrorStatus	
			5.2.4.1	coStrategyPlanBlock	

Requirements Traceability Matrix (RTM)					
FR ID	**Functional Requirement**	**Dialog ID**	**Object ID**	**Object Name**	**Additional Specifications**
			5.2.4.2	coStrategySplitsBlock	
3.5.2.2.2	Retrieve Priority Strategies				
		G.1			
			5.2.1.5	priorityStrategyTable	
			5.2.1.5.1	priorityStrategyNumber	
			5.2.1.5.2	priorityStrategyServicePhases	
			5.2.1.5.3	priorityStrategyPhaseOmits	
			5.2.1.5.4	priorityStrategyPedOmits	
			5.2.1.5.5	priorityStrategyDescription	
3.5.2.2.3	Retrieve Priority Splits				
		G.1			
			5.2.1.6	priorityStrategyExtensionToSplitTable	
			5.2.1.6.1	priorityStrategyMaximumReductionTime	
			5.2.1.6.2	priorityStrategyMaximumExtensionTime	
3.5.2.2.4	Retrieve Default Coordination Pattern				
		G.1			
			5.2.1.7	priorityStrategyDefaultCoordPattern	
3.5.2.2.5	Retrieve Maximum Priority Strategies Supported				
		G.1			
			5.2.1.4	priorityStrategiesMax	
3.5.2.2.6	Retrieve Maximum Service Requests To Consider				
		G.1			
			5.2.1.1	maxStrategyRequestsToConsider	
3.5.2.3	Monitor the Status of the CO				
		G.1			
			5.2.1.2	priorityStrategyRequestedTable	
			5.2.1.2.2	priorityStrategyRequested	
			5.2.1.2.3	priorityStrategyRequestedTimeOfServiceDesired	

Requirements Traceability Matrix (RTM)					
FR ID	**Functional Requirement**	**Dialog ID**	**Object ID**	**Object Name**	**Additional Specifications**
			5.2.1.2.4	priorityStrategyRequestedTimeOfEstimatedDeparture	
			5.2.1.2.5	priorityStrategyRequestedStatusInCO	
			5.2.1.3	coBusy	
3.5.3	Interface—PRG to PRS				All dialogs and objects between the PRG and PRS must be supported unless the PRG and the PRS are implemented as an integral part of the same physical device.
3.5.3.1	Receive Priority Requests				
3.5.3.1.1	Initiate a Priority Request				
		4.2.3.1			
			5.1.2.8	prgPriorityRequestAbsolute	
			5.1.1.1	priorityRequestTable	
			5.1.1.1.1	priorityRequestEntryNumber	
			5.1.1.1.2	priorityRequestID	
			5.1.1.1.3	priorityRequestVehicleID	
			5.1.1.1.4	priorityRequestVehicleClassType	
			5.1.1.1.5	priorityRequestVehicleClassLevel	
			5.1.1.1.6	priorityRequestServiceStrategyNumber	
			5.1.1.1.7	priorityRequestTimeOfServiceDesired	
			5.1.1.1.8	priorityRequestTimeOfEstimatedDeparture	
			5.1.1.1.9	priorityRequestStatusInPRS	
			5.1.1.1.10	priorityRequestTimeOfMessage	
			5.1.1.1.11	priorityRequestTimeToLive	
			5.1.1.1.12	priorityRequestTimeOfServiceDesiredInPRS	
			5.1.1.1.13	priorityRequestTimeOfEstimatedDepartureInPRS	

Requirements Traceability Matrix (RTM)					
FR ID	**Functional Requirement**	**Dialog ID**	**Object ID**	**Object Name**	**Additional Specifications**
			5.1.1.1.14	priorityRequestTimeOfRequest	
			5.1.1.4	priorityRequestReserviceTimer	
			5.1.1.5	priorityRequestReserviceClass1Time	
			5.1.1.6	priorityRequestReserviceClass2Time	
			5.1.1.7	priorityRequestReserviceClass3Time	
			5.1.1.8	priorityRequestReserviceClass4Time	
			5.1.1.9	priorityRequestReserviceClass5Time	
			5.1.1.10	priorityRequestReserviceClass6Time	
			5.1.1.11	priorityRequestReserviceClass7Time	
			5.1.1.12	priorityRequestReserviceClass8Time	
			5.1.1.13	priorityRequestReserviceClass9Time	
			5.1.1.14	priorityRequestReserviceClass10Time	
3.5.3.1.2	Send a Priority Request Update				
		4.2.3.2			
			5.1.2.9	prgPriorityUpdateAbsolute	
			5.1.1.1	priorityRequestTable	
			5.1.1.1.1	priorityRequestEntryNumber	
			5.1.1.1.2	priorityRequestID	
			5.1.1.1.3	priorityRequestVehicleID	
			5.1.1.1.4	priorityRequestVehicleClassType	
			5.1.1.1.5	priorityRequestVehicleClassLevel	
			5.1.1.1.6	priorityRequestServiceStrategyNumber	
			5.1.1.1.7	priorityRequestTimeOfServiceDesired	
			5.1.1.1.8	priorityRequestTimeOfEstimatedDeparture	
			5.1.1.1.9	priorityRequestStatusInPRS	
			5.1.1.1.11	priorityRequestTimeToLive	
			5.1.1.1.12	priorityRequestTimeOfServiceDesiredInPRS	
			5.1.1.1.13	priorityRequestTimeOfEstimatedDepartureInPRS	

Requirements Traceability Matrix (RTM)					
FR ID	**Functional Requirement**	**Dialog ID**	**Object ID**	**Object Name**	**Additional Specifications**
			5.1.1.1.14	priorityRequestTimeOfRequest	
3.5.3.1.3	Send a Cancel Priority Request				
		4.2.3.3			
			5.1.2.5	prgPriorityCancel	
			5.1.1.1	priorityRequestTable	
			5.1.1.1.1	priorityRequestEntryNumber	
			5.1.1.1.2	priorityRequestID	
			5.1.1.1.3	priorityRequestVehicleID	
			5.1.1.1.4	priorityRequestVehicleClassType	
			5.1.1.1.5	priorityRequestVehicleClassLevel	
			5.1.1.1.6	priorityRequestServiceStrategyNumber	
			5.1.1.1.9	priorityRequestStatusInPRS	
3.5.3.1.4	Send a Clear Priority Request				
		4.2.3.4			
			5.1.2.6	prgPriorityClear	
			5.1.1.1	priorityRequestTable	
			5.1.1.1.1	priorityRequestEntryNumber	
			5.1.1.1.2	priorityRequestID	
			5.1.1.1.3	priorityRequestVehicleID	
			5.1.1.1.4	priorityRequestVehicleClassType	
			5.1.1.1.5	priorityRequestVehicleClassLevel	
			5.1.1.1.6	priorityRequestServiceStrategyNumber	
			5.1.1.1.7	priorityRequestTimeOfServiceDesired	
			5.1.1.1.8	priorityRequestTimeOfEstimatedDeparture	
			5.1.1.1.9	priorityRequestStatusInPRS	
			5.1.1.1.10	priorityRequestTimeOfMessage	
			5.1.1.1.11	priorityRequestTimeToLive	
			5.1.1.1.12	priorityRequestTimeOfServiceDesiredInPRS	

Requirements Traceability Matrix (RTM)					
FR ID	**Functional Requirement**	**Dialog ID**	**Object ID**	**Object Name**	**Additional Specifications**
			5.1.1.1.13	priorityRequestTimeOfEstimatedDepartureInPRS	
3.5.3.1.5	Initiate a Priority Request – NTCIP 1211 v01				
		4.2.3.6			
			5.1.2.1	prgPriorityRequest	
			5.1.1.1	priorityRequestTable	
			5.1.1.1.1	priorityRequestEntryNumber	
			5.1.1.1.2	priorityRequestID	
			5.1.1.1.3	priorityRequestVehicleID	
			5.1.1.1.4	priorityRequestVehicleClassType	
			5.1.1.1.5	priorityRequestVehicleClassLevel	
			5.1.1.1.6	priorityRequestServiceStrategyNumber	
			5.1.1.1.7	priorityRequestTimeOfServiceDesired	
			5.1.1.1.8	priorityRequestTimeOfEstimatedDeparture	
			5.1.1.1.9	priorityRequestStatusInPRS	
			5.1.1.1.10	priorityRequestTimeOfMessage	
			5.1.1.1.11	priorityRequestTimeToLive	
			5.1.1.1.12	priorityRequestTimeOfServiceDesiredInPRS	
			5.1.1.1.13	priorityRequestTimeOfEstimatedDepartureInPRS	
			5.1.1.4	priorityRequestReserviceTimer	
			5.1.1.5	priorityRequestReserviceClass1Time	
			5.1.1.6	priorityRequestReserviceClass2Time	
			5.1.1.7	priorityRequestReserviceClass3Time	
			5.1.1.8	priorityRequestReserviceClass4Time	
			5.1.1.9	priorityRequestReserviceClass5Time	
			5.1.1.10	priorityRequestReserviceClass6Time	
			5.1.1.11	priorityRequestReserviceClass7Time	
			5.1.1.12	priorityRequestReserviceClass8Time	

Requirements Traceability Matrix (RTM)					
FR ID	**Functional Requirement**	**Dialog ID**	**Object ID**	**Object Name**	**Additional Specifications**
			5.1.1.13	priorityRequestReserviceClass9Time	
			5.1.1.14	priorityRequestReserviceClass10Time	
3.5.3.1.6	Send a Priority Request Update – NTCIP 1211 v01				
		4.2.3.7			
			5.1.2.2	prgPriorityUpdate	
			5.1.1.1	priorityRequestTable	
			5.1.1.1.1	priorityRequestEntryNumber	
			5.1.1.1.2	priorityRequestID	
			5.1.1.1.3	priorityRequestVehicleID	
			5.1.1.1.4	priorityRequestVehicleClassType	
			5.1.1.1.5	priorityRequestVehicleClassLevel	
			5.1.1.1.6	priorityRequestServiceStrategyNumber	
			5.1.1.1.7	priorityRequestTimeOfServiceDesired	
			5.1.1.1.8	priorityRequestTimeOfEstimatedDeparture	
			5.1.1.1.9	priorityRequestStatusInPRS	
			5.1.1.1.11	priorityRequestTimeToLive	
			5.1.1.1.12	priorityRequestTimeOfServiceDesiredInPRS	
			5.1.1.1.13	priorityRequestTimeOfEstimatedDepartureInPRS	
3.5.3.2	Receive Priority Request Status				This dialog and objects between the PRG and PRS must be supported unless the PRG and the PRS are implemented as an integral part of the same physical device.
		4.2.3.5			
			5.1.2.3	prgPriorityStatusControl	
			5.1.2.4	prgPriorityStatusBuffer	
			5.1.1.1	priorityRequestTable	

Requirements Traceability Matrix (RTM)					
FR ID	**Functional Requirement**	**Dialog ID**	**Object ID**	**Object Name**	**Additional Specifications**
			5.1.1.1.1	priorityRequestEntryNumber	
			5.1.1.1.2	priorityRequestID	
			5.1.1.1.3	priorityRequestVehicleID	
			5.1.1.1.4	priorityRequestVehicleClassType	
			5.1.1.1.5	priorityRequestVehicleClassLevel	
			5.1.1.1.6	priorityRequestServiceStrategyNumber	
3.5.4	Interface—PRS to CO				
3.5.4.1	Exchange Service Request		The following mapping is only applicable if the CO is the manager.		This dialog and objects between the PRS and CO shall be supported unless the PRS and the CO are implemented as an integral part of the same physical device.
		4.2.4.1.1			
			5.2.2.1	prsServiceRequest	
			5.1.1.1.6	priorityRequestServiceStrategyNumber	
			5.1.1.1.9	priorityRequestStatusInPRS	
			5.1.1.1.12	priorityRequestTimeOfServiceDesiredInPRS	
			5.1.1.1.13	priorityRequestTimeOfEstimatedDepartureInPRS	
			5.1.1.2	prsBusy	
			5.2.1.2.2	priorityStrategyRequested	
			5.2.1.2.3	priorityStrategyRequestedTimeOfServiceDesired	
			5.2.1.2.4	priorityStrategyRequestedTimeOfEstimatedDeparture	
			5.2.1.2.5	priorityStrategyRequestStatusInCO	
			5.2.1.3	coBusy	

Requirements Traceability Matrix (RTM)					
FR ID	**Functional Requirement**	**Dialog ID**	**Object ID**	**Object Name**	**Additional Specifications**
3.5.4.1	Exchange Service Request		The following mapping is only applicable if the PRS is the manager.		This dialog and objects between the PRS and CO shall be supported unless the PRS and the CO are implemented as an integral part of the same physical device.
		4.2.4.1.2			
			5.2.2.1	prsServiceRequest	
			5.1.1.1.6	priorityRequestServiceStrategyNumber	
			5.1.1.1.9	priorityRequestStatusInPRS	
			5.1.1.1.12	priorityRequestTimeOfServiceDesiredInPRS	
			5.1.1.1.13	priorityRequestTimeOfEstimatedDepartureInPRS	
			5.1.1.2	prsBusy	
			5.2.1.2.2	priorityStrategyRequested	
			5.2.1.2.3	priorityStrategyRequestedTimeOfServiceDesired	
			5.2.1.2.4	priorityStrategyRequestedTimeOfEstimatedDeparture	
			5.2.1.2.5	priorityStrategyRequestedStatusInCO	
			5.2.1.3	coBusy	
3.5.4.2	Exchange Service Request Status		The following mapping is only applicable if the CO is the manager.		This dialog and objects between the PRS and CO shall be supported unless the PRS and the CO are implemented as an integral part of the same physical device.
		4.2.4.2.1			
			5.2.2.1	prsServiceRequest	
			5.2.1.3	coBusy	

Requirements Traceability Matrix (RTM)					
FR ID	**Functional Requirement**	**Dialog ID**	**Object ID**	**Object Name**	**Additional Specifications**
3.5.4.2	Exchange Service Request Status		The following mapping is only applicable if the PRS is the manager.		This dialog and objects between the PRS and CO shall be supported unless the PRS and the CO are implemented as an integral part of the same physical device.
		4.2.4.2.2			
			5.2.2.1	prsServiceRequest	
			5.1.1.1.6	priorityRequestServiceStrategyNumber	
			5.1.1.1.12	priorityRequestTimeOfServiceDesiredInPRS	
			5.1.1.1.13	priorityRequestTimeOfEstimatedDepartureInPRS	
			5.1.1.1.9	priorityRequestStatusInPRS	
			5.2.1.2.2	priorityStrategyRequested	
			5.2.1.2.3	priorityStrategyRequestedTimeOfServiceDesired	
			5.2.1.2.4	priorityStrategyRequestedTimeOfEstimatedDeparture	
			5.2.1.2.5	priorityStrategyRequestedStatusInCO	
			5.2.1.3	coBusy	
3.6	Supplemental Requirements				
3.6.1	Response Time for Requests				
				See Requirement 3.6.1 in PRL	
3.6.2	Process Priority Requests				
3.6.2.1	Support Multiple Priority Requests				
		4.2.4.1.4			
			5.1.2.8	prgPriorityRequestAbsolute	
			5.1.1.1	priorityRequestTable	
			5.1.1.1.1	priorityRequestEntryNumber	

Requirements Traceability Matrix (RTM)					
FR ID	**Functional Requirement**	**Dialog ID**	**Object ID**	**Object Name**	**Additional Specifications**
			5.1.1.1.2	priorityRequestID	
			5.1.1.1.3	priorityRequestVehicleID	
			5.1.1.1.4	priorityRequestVehicleClassType	
			5.1.1.1.5	priorityRequestVehicleClassLevel	
			5.1.1.1.6	priorityRequestServiceStrategyNumber	
			5.1.1.1.9	priorityRequestStatusInPRS	
3.6.2.2	Clear Expired Priority Request				
		4.2.4.1.4			
			5.1.1.1	priorityRequestTable	
			5.1.1.1.1	priorityRequestEntryNumber	
			5.1.1.1.2	priorityRequestID	
			5.1.1.1.3	priorityRequestVehicleID	
			5.1.1.1.4	priorityRequestVehicleClassType	
			5.1.1.1.5	priorityRequestVehicleClassLevel	
			5.1.1.1.6	priorityRequestServiceStrategyNumber	
			5.1.1.1.7	priorityRequestTimeOfServiceDesired	
			5.1.1.1.8	priorityRequestTimeOfEstimatedDeparture	
			5.1.1.1.9	priorityRequestStatusInPRS	
			5.1.1.1.10	priorityRequestTimeOfMessage	
			5.1.1.1.11	priorityRequestTimeToLive	
			5.1.1.1.12	priorityRequestTimeOfServiceDesiredInPRS	
			5.1.1.1.13	priorityRequestTimeOfEstimatedDepartureInPRS	
3.6.2.1	Support Multiple Priority Requests—NTCIP 1211 v01				
		4.2.4.1.4			
			5.1.2.1	prgPriorityRequest	
			5.1.1.1	priorityRequestTable	
			5.1.1.1.1	priorityRequestEntryNumber	
			5.1.1.1.2	priorityRequestID	

Requirements Traceability Matrix (RTM)					
FR ID	**Functional Requirement**	**Dialog ID**	**Object ID**	**Object Name**	**Additional Specifications**
			5.1.1.1.3	priorityRequestVehicleID	
			5.1.1.1.4	priorityRequestVehicleClassType	
			5.1.1.1.5	priorityRequestVehicleClassLevel	
			5.1.1.1.6	priorityRequestServiceStrategyNumber	
			5.1.1.1.9	priorityRequestStatusInPRS	
3.6.3	Process Service Requests				
		4.2.4.1.3			
			5.2.1.2	priorityStrategyRequestedTable	
			5.2.1.2.2	priorityStrategyRequested	
			5.2.1.2.3	priorityStrategyRequestedTimeOfServiceDesired	
			5.2.1.2.4	priorityStrategyRequestedTimeOfEstimatedDeparture	
			5.2.1.2.5	priorityStrategyRequestedStatusInCO	
			5.2.1.5	priorityStrategyTable	
			5.2.1.5.1	priorityStrategyNumber	
			5.2.1.5.2	priorityStrategyServicePhases	
			5.2.1.5.3	priorityStrategyPhaseOmits	
			5.2.1.5.4	priorityStrategyPedOmits	
			5.2.1.5.5	priorityStrategyDescription	
			5.2.1.6	priorityStrategyExtensionToSplitTable	
			5.2.1.6.1	priorityStrategyMaximumReductionTime	
			5.2.1.6.2	priorityStrategyMaximumExtensionTime	
H.2	Derived GLOBAL Functional Requirements				
H.2.1	Determine Device Component Information				
		G.1			
			NTCIP 1201 v03 Sec. 2.2.2	globalMaxModules	
			NTCIP 1201 v03 Sec. 2.2.3.1	moduleNumber	

Requirements Traceability Matrix (RTM)					
FR ID	**Functional Requirement**	**Dialog ID**	**Object ID**	**Object Name**	**Additional Specifications**
			NTCIP 1201 v03 Sec. 2.2.3.2	moduleDeviceNode	
			NTCIP 1201 v03 Sec. 2.2.3.3	moduleMake	
			NTCIP 1201 v03 Sec. 2.2.3.4	moduleModel	
			NTCIP 1201 v03 Sec. 2.2.3.5	moduleVersion	
			NTCIP 1201 v03 Sec. 2.2.3.6	moduleType	
H.2.2	Determine Device Configuration Identifier				
		G.1			
			NTCIP 1201 v03 Sec. 2.2.1	globalSetIDParameter	
H.2.3	Determine Supported Standards				
		G.1			
			NTCIP 1201 v03 Sec. 2.2.4	controllerBaseStandards	
H.2.4	Determine System Name				
		G.1			
			RFC1213 Clause 6	sysName	
H.2.5	Manage Time				
H.2.5.1	Set Time				
		G.3			
			NTCIP 1201 v03 Sec. 2.4.1	globalTime	
H.2.5.2	Set Time Zone				
		G.3			
			NTCIP 1201 v03 Sec. 2.4.6	controllerStandardTimeZone	

Requirements Traceability Matrix (RTM)					
FR ID	**Functional Requirement**	**Dialog ID**	**Object ID**	**Object Name**	**Additional Specifications**
H.2.5.3	Set Daylight Savings Mode				
		G.3			
			NTCIP 1201 v03 Sec. 2.4.8.2.1	dstEntryNumber	
			NTCIP 1201 v03 Sec. 2.4.8.2.2	dstBeginMonth	
			NTCIP 1201 v03 Sec. 2.4.8.2.3	dstBeginOccurrences	
			NTCIP 1201 v03 Sec. 2.4.8.2.4	dstBeginDayOfWeek	
			NTCIP 1201 v03 Sec. 2.4.8.2.5	dstBeginDayofMonth	
			NTCIP 1201 v03 Sec. 2.4.8.2.6	dstBeginSecondsToTransition	
			NTCIP 1201 v03 Sec. 2.4.8.2.7	dstEndMonth	
			NTCIP 1201 v03 Sec. 2.4.8.2.8	dstEndOccurrences	
			NTCIP 1201 v03 Sec. 2.4.8.2.9	dstEndDayOfWeek	
			NTCIP 1201 v03 Sec. 2.4.8.2.10	dstEndDayofMonth	
			NTCIP 1201 v03 Sec. 2.4.8.2.11	dstEndSecondsToTransition	
			NTCIP 1201 v03 Sec. 2.4.8.2.12	dstSecondsToAdjust	
H.2.5.4	Verify Current Time				
		G.1			
			NTCIP 1201 v03 Sec. 2.4.1	globalTime	
			NTCIP 1201 v03 Sec. 2.4.6	controllerStandardTimeZone	

Requirements Traceability Matrix (RTM)					
FR ID	**Functional Requirement**	**Dialog ID**	**Object ID**	**Object Name**	**Additional Specifications**
			NTCIP 1201 v03 Sec. 2.4.8.1	maxDaylightSavingEntries	
			NTCIP 1201 v03 Sec. 2.4.8.2.1	dstEntryNumber	
			NTCIP 1201 v03 Sec. 2.4.8.2.2	dstBeginMonth	
			NTCIP 1201 v03 Sec. 2.4.8.2.3	dstBeginOccurrences	
			NTCIP 1201 v03 Sec. 2.4.8.2.4	dstBeginDayOfWeek	
			NTCIP 1201 v03 Sec. 2.4.8.2.5	dstBeginDayofMonth	
			NTCIP 1201 v03 Sec. 2.4.8.2.6	dstBeginSecondsToTransition	
			NTCIP 1201 v03 Sec. 2.4.8.2.7	dstEndMonth	
			NTCIP 1201 v03 Sec. 2.4.8.2.8	dstEndOccurrences	
			NTCIP 1201 v03 Sec. 2.4.8.2.9	dstEndDayOfWeek	
			NTCIP 1201 v03 Sec. 2.4.8.2.10	dstEndDayofMonth	
			NTCIP 1201 v03 Sec. 2.4.8.2.11	dstEndSecondsToTransition	
			NTCIP 1201 v03 Sec. 2.4.8.2.12	dstSecondsToAdjust	
			NTCIP 1201 v03 Sec. 2.4.7	controllerLocalTime	
H.2.6	Support Logged Data				
H.2.6.1	Retrieve Current Configuration of Logging Service				
		H.3.1.1			
			NTCIP 1103 v02 Sec. A.7.3.1	eventClassNumber	

Requirements Traceability Matrix (RTM)					
FR ID	**Functional Requirement**	**Dialog ID**	**Object ID**	**Object Name**	**Additional Specifications**
			NTCIP 1103 v02 Sec. A.7.3.2	eventClassLimit	
			NTCIP 1103 v02 Sec. A.7.3.3	eventClassClearTime	
			NTCIP 1103 v02 Sec. A.7.3.4	eventClassDescription	
			NTCIP 1103 v02 Sec. A.7.3.6	eventClassNumEvents	
			NTCIP 1103 v02 Sec. A.7.5.1	eventConfigID	
			NTCIP 1103 v02 Sec. A.7.5.2	eventConfigClass	
			NTCIP 1103 v02 Sec. A.7.5.3	eventConfigMode	
			NTCIP 1103 v02 Sec. A.7.5.4	eventConfigCompareValue	
			NTCIP 1103 v02 Sec. A.7.5.5	eventConfigCompareValue2	
			NTCIP 1103 v02 Sec. A.7.5.6	eventConfigCompareOID	
			NTCIP 1103 v02 Sec. A.7.5.7	eventConfigLogOID	
			NTCIP 1103 v02 Sec. A.7.5.8	eventConfigAction	
			NTCIP 1103 v02 Sec. A.7.5.9	eventConfigStatus	
H.2.6.2	Configure Logging Service				
		H.3.1.2			
			NTCIP 1103 v02 Sec. A.7.3.1	eventClassNumber	
			NTCIP 1103 v02 Sec. A.7.3.2	eventClassLimit	

Requirements Traceability Matrix (RTM)					
FR ID	**Functional Requirement**	**Dialog ID**	**Object ID**	**Object Name**	**Additional Specifications**
			NTCIP 1103 v02 Sec. A.7.3.3	eventClassClearTime	
			NTCIP 1103 v02 Sec. A.7.3.4	eventClassDescription	
			NTCIP 1103 Sec. A.7.5.1	eventConfigID	
			NTCIP 1103 v02 Sec. A.7.5.2	eventConfigClass	
			NTCIP 1103 v02 Sec.Clause A.7.4.3	eventConfigMode	
			NTCIP 1103 v02 Sec. A.7.5.4	eventConfigCompareValue	
			NTCIP 1103 v02 Sec. A.7.5.5	eventConfigCompareValue2	
			NTCIP 1103 v02 Sec. A.7.5.6	eventConfigCompareOID	
			NTCIP 1103 v02 Sec. A.7.5.7	eventConfigLogOID	
			NTCIP 1103 v02 Sec. A.7.5.8	eventConfigAction	
			NTCIP 1103 v02 Sec. A.7.5.9	eventConfigStatus	
H.2.6.3	Retrieve Logged Data				
		H.3.1.3			
			NTCIP 1103 v02 Sec. A.7.3.1	eventClassNumber	
			NTCIP 1103 v02 Sec. A.7.3.5	eventClassNumRowsInLog	
			NTCIP 1103 v02 Sec. A.7.3.6	eventClassNumEvents	

Requirements Traceability Matrix (RTM)					
FR ID	**Functional Requirement**	**Dialog ID**	**Object ID**	**Object Name**	**Additional Specifications**
			NTCIP 1103 v02 Sec. A.7.7.1	eventLogClass	
			NTCIP 1103 v02 Sec. A.7.7.2	eventLogNumber	
			NTCIP 1103 v02 Sec. A.7.7.3	eventLogID	
			NTCIP 1103 v02 Sec. A.7.7.4	eventLogTime	
			NTCIP 1103 v02 Sec. A.7.7.5	eventLogValue	
			NTCIP 1103 v02 Sec. A.7.3	numEvents	
H.2.6.4	Clear Log				
		G.3			
			NTCIP 1103 v02 Sec. A.7.3.1	eventClassNumber	
			NTCIP 1103 v02 Sec. A.7.3.3	eventClassClearTime	
H.2.6.5	Determine Capabilities of Event Logging Service				
		G.1			
			NTCIP 1103 v02 Sec. A.7.2	maxEventClasses	
			NTCIP 1103 v02 Sec. A.7.4	maxEventLogConfigs	
			NTCIP 1103 v02 Sec. A.7.6	maxEventLogSize	
H.2.6.6	Retrieve Total Number of Logged Events				
		G.1			
			NTCIP 1103 v02 Sec. A.7.3.1	eventClassNumber	
			NTCIP 1103 v02 Sec. A.7.3.5	eventClassNumRowsInLog	

Requirements Traceability Matrix (RTM)					
FR ID	**Functional Requirement**	**Dialog ID**	**Object ID**	**Object Name**	**Additional Specifications**
			NTCIP 1103 v02 Sec. A.7.3.6	eventClassNumEvents	
			NTCIP 1103 v02 Sec. A.7.8	numEvents	
H.2.7	Supplemental Requirements for Event Monitoring				
H.2.7.1	Record and Timestamp Events				
H.2.7.2	Support a Number of Event Classes			See Requirement H.2.7.2 in PRL	
H.2.7.3	Support a Number of Event Types to Monitor			See Requirement H.2.7.3 in PRL	
H.2.7.4	Support Monitoring of Event Types				
H.2.7.4.1	Support On-Change Events				
H.2.7.4.2	Support Greater Than Events				
H.2.7.4.3	Support Less Than Events				
H.2.7.4.4	Support Hysteresis Events				
H.2.7.4.5	Support Periodic Events				
H.2.7.4.6	Support Bit-flag Events				
H.2.7.4.7	Support Event Monitoring on Any Data				
H.2.8	Support a Number of Events to Store in Log			See H.2.8 in PRL	

Annex B
Object Tree [Informative]

The Object Definitions for Signal Control and Prioritization are defined under the nema.transportation node of the ISO global naming tree to uniquely identify all management information (object definitions). It also references numerous objects in the SCP and Global subtrees of the global naming tree.

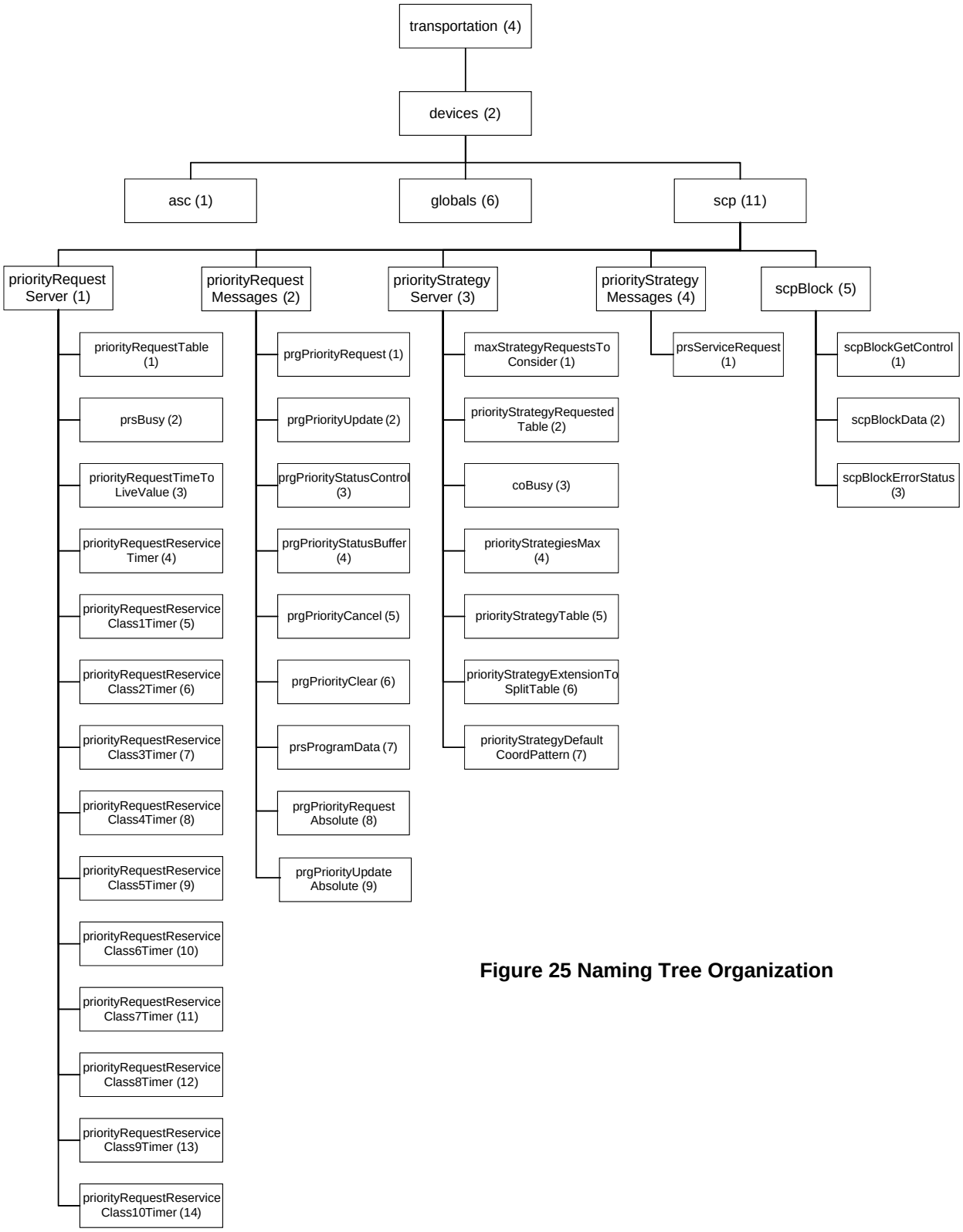

Figure 25 Naming Tree Organization

Annex C
Test Procedures [Normative]

It is anticipated that Test Procedures may be developed as part of a future revision of NTCIP 1211 v02. Annex C is a placeholder, at present.

Annex D
Documentation of Revisions [Informative]

This annex identifies significant changes that have been made to the NTCIP 1211 standard. NTCIP makes reasonable efforts to ensure that the standards are as backward compatible as possible, but the primary purpose of the standard is to provide interoperability by developing standards in a consensus environment. When changes are required to meet these objectives, the problematic objects are refined (if the issue is primarily editorial) or deprecated and, in most cases, replaced with new objects. This annex identifies why each of these changes have been made. New implementations should support the new/replacement objects; they may also support deprecated objects.

D.1 Version 1 to Version 2

NTCIP 1211 v02 is a major enhancement to NTCIP 1211 v01. NTCIP 1211 v02 has been restructured to follow an established ‘systems engineering’ approach. Several new sections were added to relate user needs identified in a concept of operations, functional requirements, interface specifications and a requirements traceability matrix to the existing sections

D.1.1 Conformance

Annex A (Priority Request Server Information Profile – Requirements List) and Annex B (Coordinator Information Profile – Requirements List) as defined in Version 1 have been deleted. The intention and functions of the Conformance Groups has been replaced by the User Needs (see Section 2), Functional Requirements (see Section 3), and particularly the Protocol Requirement List (PRL) (see Section 3), and the Requirements Traceability Matrix (RTM) (see Annex A).

D.1.2 Support for Absolute Time

The original concept for the SCP time relationship was based on the time of receipt (of the SCP request), since most of the mechanisms for SCP used a direct connection from the vehicle to the traffic controller (PRG to PRS) such that the time of receipt was exactly the time the message was sent. However, with some newer communications media (such as cellular telephone and IP network communications), variable transmission latency can result in a delay of several seconds between the transmission of the request (from the PRG) and the receipt of the message (by the PRS). This latency can have a negative impact on traffic signal response to the SCP request. However, in those cases where the PRG is known to have an accurate time source (such as GNSS), this precise time reference can be used to mark the time that the PRG generates the request.

To address this issue, three objects were added (priorityRequestTimeOfRequest, prgPriorityRequestAbsolute, prgPriorityUpdateAbsolute) to allow a PRG with an accurate time source to transmit a precise time reference; thus, allowing the PRS to compute and set the priorityRequestTimeOfMessage. By using the value in priorityRequestTimeOfRequest instead of the time of receipt, the effects of moderate but variable network delays can be minimized.

The priorityRequestTimeOfRequest can be flagged so that the time of message receipt is still used, as originally noted in the definition of priorityRequestTimeOfMessage.

D.1.3 MIB – Object Status

The STATUS of all objects was changed to "mandatory" to reflect the fact that conformance is now measured through the use of the PRL as contained in Section 3 and the RTM contained in Annex A A.

D.1.4 MIB - References

References to Global Objects are now made through the RTM rather than through comments in the MIB.

D.1.5 MIB – Corrections

Some minor corrections were made to the MIB, including spelling errors. Other corrections include:

priorityRequestStatusInPRS – Corrected description.
priorityRequestStatusInCO – Corrected description.
priorityRequestTable – Corrected
prsServiceRequest – Corrected

D.1.6 Busy State

There were some inconsistencies and ambiguities in the dialogs and the object descriptions in Version 01 for the “busy” state (coBusy and prsBusy). The dialogs and objects have been changed for clarification and for consistency.

D.1.7 Clarifications

Some dialogs have been updated to include device behavior, such as prioritization processing and service processing. In NTCIP 1211 v01, it was unclear when certain processes occur.

D.1.8 State Transition Diagram

Some corrections were made to the State Transition Diagram, including spelling errors. Other corrections included adding some flows from one state to another, and deleting some flows from one state to another, to be consistent with other corrections or clarifications made in NTCIP 1211 v02 from NTCIP 1211 v01.

Annex E
User Requests [Informative]

This annex identifies features that were suggested for this standard but either are supported by mechanisms that may not be readily obvious or are not supported by this version of the standard.

E.1 Features Not Supported by This Version

The following clauses identify certain features that were considered for this version, but were excluded from NTCIP 1211 v02.

E.1.1 Exception Reporting

There is a user need to have an entity automatically transmit data to a management station when certain conditions occur. Under this scenario, the transportation system operator can define under what conditions s/he wishes to be notified and the entity automatically notifies the management station when the condition occurs. An example may be the transportation system manager who wants to know when a priority request has been granted.

This user need is proposed to be supported in a future NTCIP 1103 v03 (to be developed) and is anticipated to standardize a method that allows any implementation to utilize a trap mechanism. However, at the time that NTCIP 1211 v02 was written, NTCIP 1103 v03 is under development.

E.1.2 Vehicle Approach

There is a user need for a traffic signal controller to provide preferential treatment for a fleet vehicle by servicing a specific phase, dependent on what direction the fleet vehicle is approaching the signalized intersection.

The implementation of NTCIP 1211 v01 and NTCIP 1211 v02 satisfies this user need, but assumes that the priority strategy information has been provided to and uploaded to the fleet vehicle or the fleet management center, depending on how the SCP system is implemented. With the priority strategy information uploaded, the fleet vehicle or fleet management center may request the appropriate priority strategy via the PRG, based on the direction of travel of the fleet vehicle approaching the signalized intersection.

For example, the traffic management agency and the fleet management agency or company may agree to configure priority strategy number 5 at a signalized intersection to provide preferential treatment for northbound through vehicles. As the fleet vehicle approaches the intersection from the northbound direction, the PRG would send a priority request asking for priority strategy number 5.

However, if the priority strategy is not uploaded to the fleet vehicle or fleet management center, the PRG is unable to indicate, as the fleet vehicle approaches the signalized intersection, which phase the traffic signal controller should service in the priority request sent to the PRS.

Thus, the user need, for the traffic signal controller to service a specific phase dependent on the direction that the fleet vehicle is approaching the signalized intersection, is only partially satisfied by NTCIP 1211 v01 and NTCIP 1211 v02. For situations where the priority strategy information is not uploaded to the fleet vehicle or the fleet management center, this user need is not satisfied.

The SCP WG discussed this and concluded that it would:

a) Result in a standard that was not backward compatible with NTCIP 1211 v01; and
b) Result in a standard that was complex.

The SCP WG revisited this issue in NTCIP 1211 V02 and concludes that the PRG may not know the specific strategy or even the phase that would serve the desired movement, but should know the entering approach, and possibly the exit approach, directions that specify the desired movement of the requesting vehicle at the intersection. The SCP WG is considering proposals for addressing this need in NTCIP 1211 v03 while maintaining backward compatibility.

Annex F
SCP Tutorial [Informative]

Annex F describes how a typical SCP system works.

In a SCP System, a Fleet Vehicle, Fleet Management, or Traffic Management initiates priority service by instructing the Priority Request Generator to send a Priority Request to a Priority Request Server. The Priority Request Server prioritizes and sorts all requests based upon Class Type and Class Level. It then sends its queue of requests to the Coordinator that resides in a Traffic Signal Controller. Based upon the strategy (or strategies based upon the implementation), the Coordinator attempts to adjust the split timing in the Coordinator to accommodate the service request(s).

F.1 Coordinator

Based upon pre-programmed entries, the Coordinator applies phase and pedestrian omits, and increases or decreases non-priority phase splits to extend the priority split time up to some maximum. Figure 26 and Figure 27 illustrate the operation.

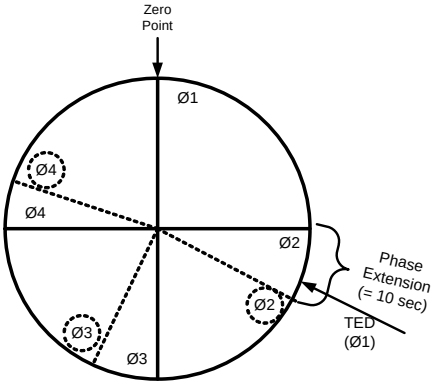

Figure 26 Early Return to Coordinated Phase

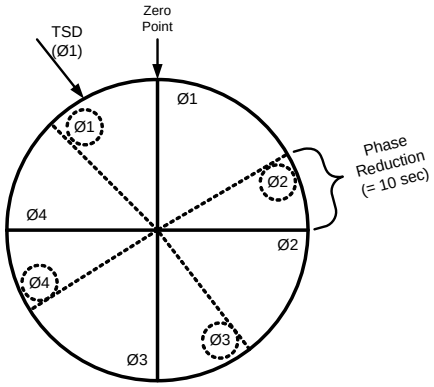

Figure 27 Late Departure from Coordinated Phase

In Figure 26, a priority request message for priority service on Phase 1 is received at some point prior to the "zero point". Projecting the priorityRequestTimeOfServiceDesired (TSD) into the anticipated timing sequence, the phase 1, 2, 3 and 4 split times are reduced by the priorityStrategyMaximumReductionTime to ensure an early return to Phase 1 green at TSD. [The split times for phases 2, 3, and 4 could have also been reduced to zero (0) by application of priorityStrategyPhaseOmits and/or priorityStrategyPedOmits.]

In Figure 27, the priority request message's priorityRequestTimeOfEstimatedDeparture (TED) would normally occur after Phase 1 would have terminated. In this case, the Phase 1 split time is extended so that Phase 1 remains green so that the fleet vehicle can make its way through the intersection. [The split times for phases 2, 3, and 4 could have also been reduced to zero (0) by application of priorityStrategyPhaseOmits and/or priorityStrategyPedOmits.]

All coordinator calculations that project TSD and TED into a future timing position shall be based upon the current pattern. The calculations are not required to take into consideration a future change in pattern and any transitioning to that new pattern.

F.2 Typical Sequence Diagram

The following figure shows a typical sequence of events for an SCP system. The sequence of events depicted assumes that the CO is the manager and the PRS is the agent for communications between the CO and the PRS.

The figure represents a situation where a priority request is initially queued by the PRS and a priority request update is sent prior to the initial priority request being highest in the queue. A service request is then retrieved by the CO and it returns the status of the service request back to the PRS. The status buffer for the priority request is then set up between the PRG and the PRS, and the PRG reads the contents of the status buffer. The CO, again, retrieves a service , and the CO returns a status of complete. The status buffer is set up again to return this status to the PRG, and the status is read. Since the priority request is now complete, the priority request is cleared from the PRS.

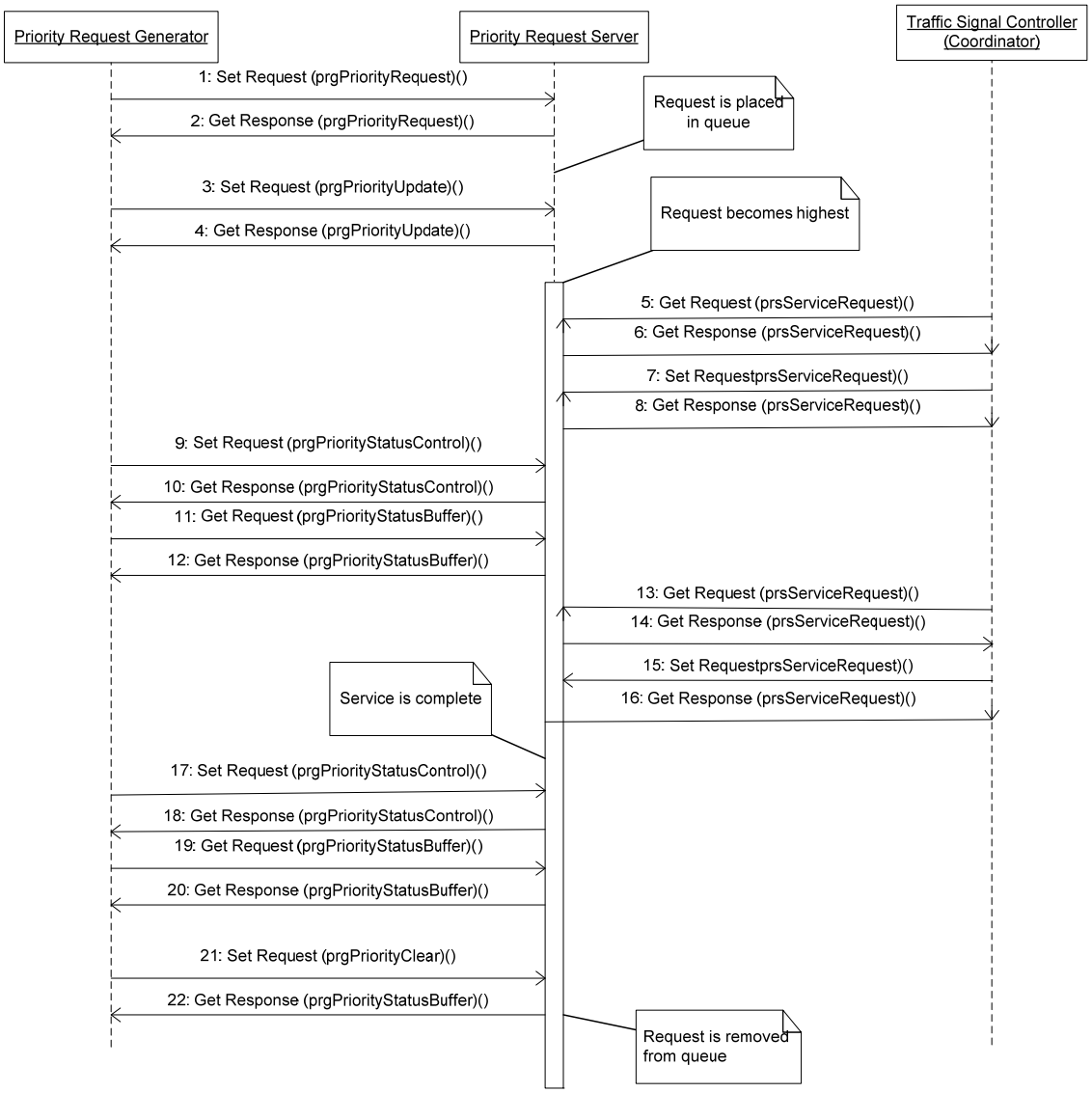

Figure 28 Priority Request Sequence Diagram (Typical)

Annex G
SNMP Interface [Normative]

The SCP system shall conform to the requirements for the Simple Network Management Protocol (SNMP) as defined in NTCIP 1103 v02. Annexes G.1 through G.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 G.5 extends the requirements of NTCIP 1103 v02 by providing additional requirements that supplement, but do not replace any requirements of NTCIP 1103 v02.

> Note: To promote interoperability and to reflect marketplace realities, NTCIP requires support for SNMP. Use of other protocols defined in NTCIP 1103 v02 (e.g., the Simple Transportation Management Protocol and the Simple Fixed Message Protocol) is discouraged for an SCP system as these have not been widely implemented and thus would likely result in decreased interoperability, limited competition, and increased resources for testing, integration, and maintenance.

G.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 29. Both the Get request and the GetResponse messages contain a list of objects as defined by the varBindingList structure (see Annex G.4).

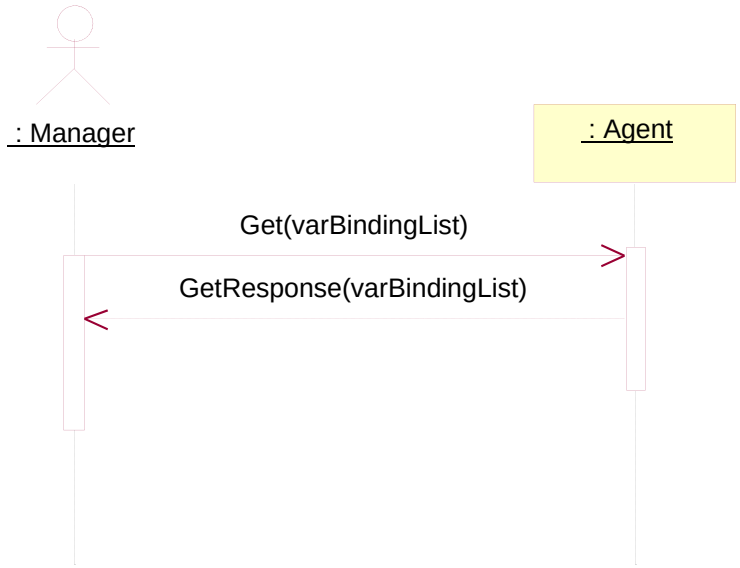

Figure 29 SNMP Get Interface

The RTM (Annex A) customizes this generic process by calling out the appropriate objects to fulfill specific requirements defined in Section 3.

G.2 Generic SNMP GET-NEXT Interface

SNMP defines a process by which a management station can explore data within a device to fulfill the requirement as defined in Section 3.4.1.3. This process consists of a GetNext request and a GetResponse as depicted in Figure 30. Both the GetNext request and the GetResponse messages contain a list of objects as defined by the varBindingList structure (see Annex G.4).

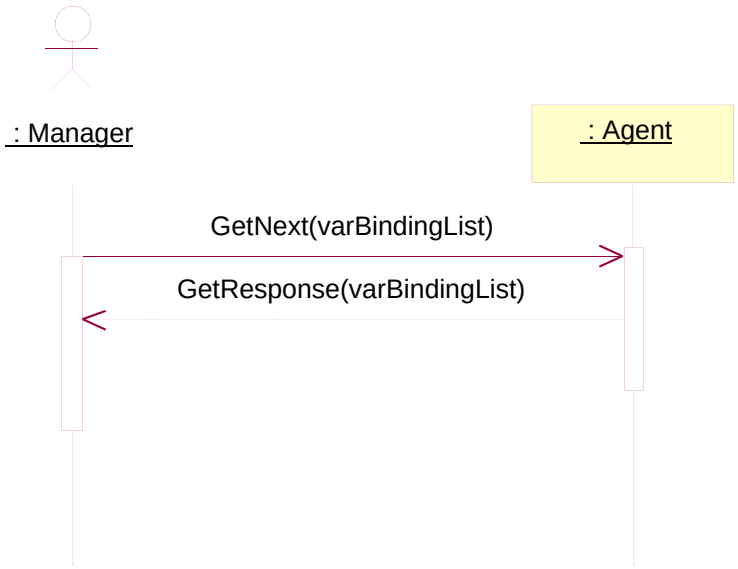

Figure 30 SNMP GetNext Interface

G.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 (sic) as depicted in Figure 31. Both the Set request and the GetResponse messages contain a list of objects as defined by the varBindingList structure (see Annex G.4).

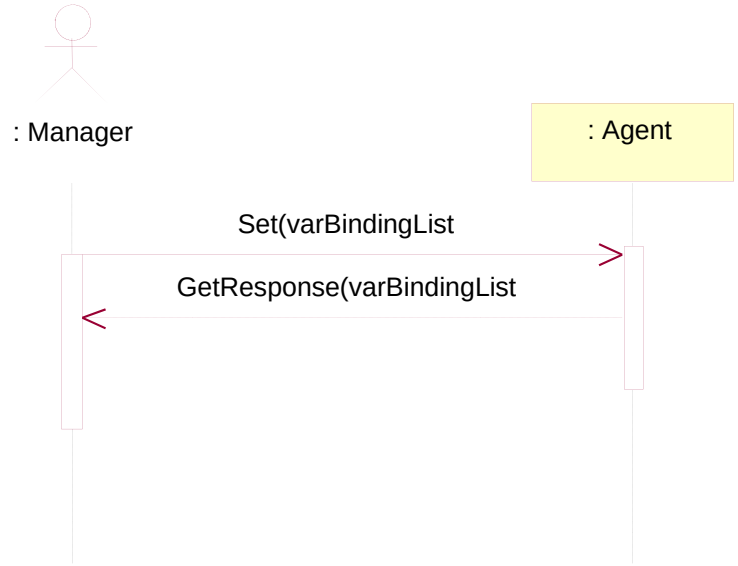

Figure 31 SNMP Set Interface

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.

This generic process is customized by subsequent sections of this standard, by referencing the 'SET' operation, and directly by the RTM, by section number, to fulfill a wide range of the requirements defined in Section 3.

G.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 is relationship is depicted in Figure 32.

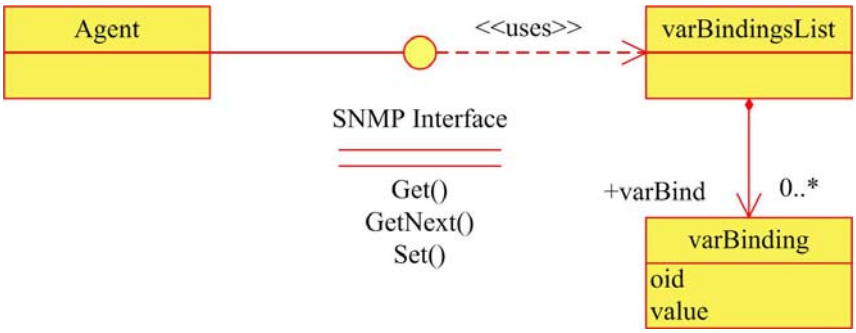

Figure 32 SNMP Interface—View of Participating Classes

G.5 Additional Requirements

G.5.1 Grouping of Objects in a Request

An agent shall allow the manager 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 NTCIP 1211 v02.

The agent 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."

G.5.2 Support of Get

An agent shall allow the manager to perform the Get operation on any supported object for which support for the Get Operation is indicated in Annex G.4.

G.5.3 Support of Get-Next

An agent shall allow the manager to perform the GetNext operation on any OBJECT IDENTIFIER.

G.5.4 Support of Set

An agent shall allow the manager to perform the Set operation on any supported object for which support for the Set Operation is indicated in Annex G.4.

G.5.5 Performance

An agent shall process the Get, GetNext, or Set request in accordance with all of the rules of NTCIP 1103 v02, including updating the value in the database and initiating the transmission of the appropriate response (assuming that the agent has permission to transmit) within 1 second of receiving the last byte of the request.

Note: If a user desires a shorter response time, then a shorter response time should be identified in the agency (procurement) specification. .

Annex H
NTCIP 1201 v03 Derived User Needs, Functional Requirements, and Dialogs [Informative]

The annex content serves as a reference for NTCIP 1211 v02. Eventually this reference information may be moved to successors of NTCIP 1201 v03.

> Note: At the time, the SCP WG needed to reference certain information from NTCIP 1201 (Global Object Definitions) such as user needs, functional requirements, and dialogs, NTCIP 1201 did not contain this type of information to the extend necessary NTCIP 1201 v3 does contain a Concept of Operations for 3 relevant functions that might be used in conjunction with SCP). The SCP WG, with support from the NTCIP Globals WG and from NEMA, decided to develop and provide the following temporary references within an annex in this standard (NTCIP 1211 v02). Annex H is scheduled for deletion when NTCIP 1201 supports the information contained.

H.1 Introduction

Content within this annex exists to serve as a reference for NTCIP 1211 v02. Eventually this information needed for reference may exist within the standard referenced.

H.2 Derived GLOBAL Functional Requirements

The following functional requirements address features defined in NTCIP 1201 v03.

H.2.1 Determine Device Component Information

The device shall allow a management station to determine identification information for each module contained in the device including:

a) An indication of the type of device
b) The manufacturer of the module
c) The model number or firmware reference of the module
d) The version of the module
e) An indication of whether it is a software or hardware module

H.2.2 Determine Device Configuration Identifier

The device shall allow a management station to determine an identifier when changes are made to the configuration.

H.2.3 Determine Supported Standards

The device shall allow a management station to determine the NTCIP standards which it supports.

H.2.4 Determine System Name

The device shall allow a management station to determine the system name of the device.

H.2.5 Manage Time

Requirements for managing the sign controller's clock follow.

H.2.5.1 Set Time

The device shall allow a management station to set the coordinated universal time to the nearest second.

H.2.5.2 Set Time Zone

The device shall allow a management station to configure the time zone in which the device is located.

H.2.5.3 Set Daylight Savings Mode

The device shall allow a management station to indicate whether or not day light savings time adjustments should be performed when determining local time.

H.2.5.4 Verify Current Time

The device shall allow a management station to determine the current time settings within the controller.

H.2.6 Support Logged Data

Requirements for managing the logged data follow.

H.2.6.1 Retrieve Current Configuration of Logging Service

Upon request from a management station, the device shall return the current configuration of the event logging service, including the classes and types of events that are currently configured

H.2.6.2 Configure Logging Service

Upon request from a management station, the device shall configure the event logging service as requested, including configuration of the event classes and event types to log.

H.2.6.3 Retrieve Logged Data

Upon request from a management station, the device shall return the event log.

H.2.6.4 Clear Log

Upon request from a management station, the device shall clear the indicated log entries of a given event class.

H.2.6.5 Determine Capabilities of Event Logging Service

Upon request from a management station, the device shall return the capabilities of the event logging service, including the number of classes, number of event types, and number of events that can be supported by the device.

H.2.6.6 Determine Total Number of Logged Events

Upon request from a management station, the device shall return the total number of events that the device has detected.

H.2.7 Supplemental Requirements for Event Monitoring

Supplemental requirements for monitoring for the occurrence of certain events follow.

H.2.7.1 Record and Timestamp Events

The device shall support the capability to record configured event types with timestamps, in a local log (log contained in the device controller), upon request by the user and/or the management station.

H.2.7.2 Support a Number of Event Classes

The device shall support the number of event classes as defined by the specification. If the specification does not define the number of event classes, the device shall support at least one event class.

H.2.7.3 Support a Number of Event Types to Monitor

The device shall support the number of event types as defined by the specification. If the specification does not define the number of event types, the device shall support at least one event type.

H.2.7.4 Support Monitoring of Event Types

Supplemental requirements for monitoring types of events follow.

H.2.7.4.1 Support On-Change Events

The device shall allow any event type configuration to monitor data for changes in value.

H.2.7.4.2 Support Greater Than Events

The device shall allow any event type configuration to monitor data for values exceeding a defined threshold for a period of time.

H.2.7.4.3 Support Less Than Events

The device shall allow any event type configuration to monitor data for values falling below a defined threshold for a period of time.

H.2.7.4.4 Support Hysteresis Events

The device shall allow any event type configuration to monitor data for values exceeding an upper limit or dropping below a lower limit.

H.2.7.4.5 Support Periodic Events

The device shall allow any event type configuration to monitor data on a periodic basis.

H.2.7.4.6 Support Bit-flag Events

The device shall allow any event type configuration to monitor one or more bits of a value becoming true (e.g., obtaining a value of one).

H.2.7.4.7 Support Event Monitoring on Any Data

The device shall allow a management station to configure any event type to monitor any piece of data supported by the device within the logical rules of the type of event (e.g., ASCII strings should not be monitored with greater than or less than conditions).

Note: This allows a user to monitor an event based on the value of any data.

H.2.8 Support a Number of Events to Store in Log

The device event log shall support the number of events as defined by the specification. If the specification does not define the number of events for the log, the device shall support at least one event in the log.

H.3 Derived GLOBAL Dialogs

H.3.1 Manage Communications Environment

Standardized dialogs for managing the communications environment that are more complex than simple GETs or SETs follow.

H.3.1.1 Determining Current Configuration of Event Reporting/Logging Service

The standardized dialog for a management station to determine the current configuration of the logging service and/or exception reporting events shall be as follows:

a) (Precondition) The management station shall be aware of the number of classes and event configurations supported by the SCP. (See Annex A for Requirement 3.4.2.5)
b) For each row of the event class table, the management station shall GET the following data:
   1) eventClassLimit.x
   2) eventClassClearTime.x
   3) eventClassDescription.x
c) For each row of the event configuration table, the management station shall GET the following data:
   1) eventConfigClass.y
   2) eventConfigMode.y
   3) eventConfigCompareValue.y
   4) eventConfigCompareValue2.y
   5) eventConfigCompareOID.y
   6) eventConfigLogOID.y
   7) eventConfigAction.y
   8) eventConfigStatus.y

   Where:
   x = event class number
   y = event configuration identifier

H.3.1.2 Configuring Reporting/Logging Service

The standardized dialog for a management station to configure the logging service or events to be reported shall be as follows:

a) (Precondition) The management station shall determine that there are sufficient rows in the event configuration and event class tables to download the proposed configuration.
b) The management station shall SET the following data to the desired values to configure each desired event class:
   1) eventClassLimit.x
   2) eventClassClearTime.x
   3) eventClassDescription.x

   Note: Each event type to be monitored is classified into one event class. For example, critical events may be grouped into Class 1 events and warnings may be grouped into Class 2 events. This step, defines the structure of each class of events.
c) The management station shall SET the following data to the desired values to configure each desired event to be monitored:
   1) eventConfigClass.y
   2) eventConfigMode.y
   3) eventConfigCompareValue.y
   4) eventConfigCompareValue2.y
   5) eventConfigCompareOID.y
   6) eventConfigLogOID.y
   7) eventConfigAction.y

Note: Depending on the value of eventConfigMode, not all other objects may be necessary for the event to be defined, however, they shall always be SET as a part of the standardized dialog.

d) The management station shall GET eventConfigStatus.y to check that there is not an error in the configuration.

Where:
x = event class number
y = event configuration identifier

H.3.1.3 Retrieving Logged Data

The standardized dialog for a management station to retrieve logged data shall be as follows:

a) (Precondition) The management station shall be aware of the number of events that had previously been reported for the device for the subject event class (e.g., from the previous performance of this operation).
b) The management station shall GET the following data:
   1) eventClassNumRowsInLog.x
   2) eventClassNumEvents.x
c) If eventClassNumEvents.x has not changed since the previous reading, the management station shall exit the process. Otherwise, the management station shall determine the additional number of events that have occurred since the last read.
   Note: This is generally determined by subtracting the previous number of events from eventClassNumEvents; however, since this object wraps at 65535, the management station should be prepared to determine the differential if eventClassNumEvents is less than the previous number.
d) The management station shall determine the lesser of eventClassNumRowsInLog and the additional number of events that have occurred since the last read. This number shall be termed the Events to Read.
e) Starting with y = eventClassNumRowsInLog and working down until y = (eventClassNumRowsInLog - Events to Read), the management station shall GET the following data:
   1) eventLogID.x.y
   2) eventLogTime.x.y
   3) eventLogValue.x.y
f) Repeat the same GET operation with y decremented by one (1) for each set of duplicated values (until y reaches a value of zero (0)).
   Note: If the event class is full and another event occurs, the new event is recorded in the last entry and all previously logged data is moved to one index lower with index 1 being deleted from the table. Thus, if a duplicate row is detected (e.g., same event at same time), it is likely an indication that the same event is being read and that a new event was added to the log.
   Note: The management station may wish to clear the event log after the read to minimize the above problem.

   Where:
   x = event log class
   y = event log number

H.3.2 Automatic Reporting of Events (SNMP Traps)

Note: Ultimately, the NTCIP-specific handling of traps is anticipated to be defined in NTCIP 1103 v03 (under development at the time of NTCIP 1211 v02 publication). However, the current version (NTCIP 1103 v02) does not contain any trap definitions. Therefore, NTCIP 1211 v02 does not address traps.

H.3.3 Determining Device Component Information

The standardized dialog for a management station to identify the hardware and software configuration of

a NTCIP device shall be as follows:

a) The management station shall GET the object globalMaxModules.0.
b) For each row in the module table, the management station shall GET the following objects:
   1) moduleDeviceNode.x,
   2) moduleMake.x,
   3) moduleModel.x,
   4) moduleVersion.x,
   5) moduleType.x.

   Where:
   x = module number.

H.3.4 Global Time Data

The following subsection identifies the interface to a field device to obtain and manage time related information.

H.3.4.1 Graphical Depiction of Global Time Data

See Figure 33.

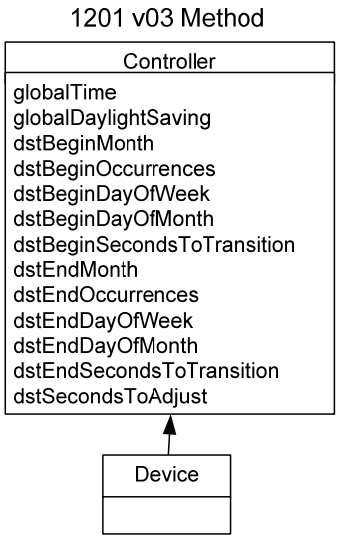

Figure 33 Global Time Data

H.4 EXTERNAL DATA ELEMENTS

NTCIP 1211 v02 references data elements in Annex H that are physically defined in NTCIP 1201 v03. See NTCIP 1201 v03.

§

